AIX バージョン 7.2

プリンターおよび印刷

AIX バージョン 7.2

プリンターおよび印刷

IBM

お願い

本書および本書で紹介する製品をご使用になる前に、 281 ページの『特記事項』に記載されている情報をお読みください。

本書は AIX バージョン 7.2 および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： AIX Version 7.2
Printers and printing

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

目次

本書について

本書は、ユーザーとシステム管理者を対象に、ファイルの印刷、印刷要求の進行状況の管理、プリンターの構成などの作業を実施する方法について、詳細な情報を提供します。上級者とプログラマーのために、本書では印刷スプーリングに関する情報、およびプリンター・バックエンドなどのトピックについても説明します。本書は、オペレーティング・システムに付属のドキュメンテーション CD にも収録されています。

強調表示

本書では、以下の強調表示規則を使用します。

太字	名前がシステムによって事前定義されているコマンド、サブルーチン、キーワード、ファイル、構造体、ディレクトリー、および他の項目を示します。さらに、ユーザーが選択するボタン、ラベル、およびアイコンなどのグラフィカル・オブジェクトも示します。
イタリック	実際の名前または値をユーザーが提供する必要があるパラメーターを示します。
モノスペース	具体的なデータ値の例、表示される可能性があるテキストの例、プログラマーとして作成する可能性があるプログラム・コードの一部の例、システムからのメッセージ、またはユーザーが実際に入力する必要がある情報を示します。

AIX

AIX® オペレーティング・システムでは、すべてケース・センシティブとなっています。これは、英大文字と小文字を区別するということです。例えば、**ls** コマンドを使用するとファイルをリストすることができます。 `LS` と入力すると、システムはそのコマンドが「`is not found`」(見付からない) と応答します。同様に、**FILEA**、**FiLea**、および **filea** は、同じディレクトリーにある場合でも、3 つの異なるファイル名です。想定外の処理が実行されないように、常に正しい大/小文字を使用するようにしてください。

ISO 9000

当製品の開発および製造には、ISO 9000 登録品質システムが使用されました。

プリンターおよび印刷

本書は、ユーザーとシステム管理者を対象に、ファイルの印刷、印刷要求の進行状況の管理、プリンターの構成などの作業を実施する方法について、概念と手順を説明します。上級者とプログラマーのために、本書では印刷スプーリングに関する概念と手順、およびプリンター・バックエンドなどのトピックについても説明します。本書は、オペレーティング・システムに付属のドキュメンテーション CD にも収録されています。

印刷および印刷ジョブ

AIX での印刷には、数多くの構成とセットアップ・オプションを利用できます。

使用するプリンターに応じて、AIX オペレーティング・システムは最終出力の外観と特性を制御します。プリンターをシステム装置およびシステム・コンソールと同じエリアに配置する必要はありません。プリンターをローカル・システムに直接接続できますが、状況によっては印刷ジョブをネットワーク経由でリモート・システムに送信する必要が生じることもあります。

印刷ジョブを最も効率良く処理するために、AIX オペレーティング・システムはそれぞれのジョブをキューに入れて、プリンターが使用可能になるまで待機します。システムは、1 つ以上のファイルからの出力をキューに保管できます。プリンターが 1 つのファイルから出力を生成すると、システムはキュー内にある次のジョブを処理します。この処理は、キュー内のそれぞれのジョブが印刷されるまで続きます。

印刷ジョブの開始

印刷ジョブを要求するには、**qprt** コマンドまたは **smit** コマンドを使用します。

- ローカル印刷ジョブの場合は、プリンターがシステムに物理的に接続されている必要があり、ネットワーク・プリンターの場合は、プリンターがネットワーク上で接続および構成されている必要があります。
- リモート印刷ジョブの場合は、リモート印刷サーバーと通信するようにシステムを構成する必要があります。
- ファイルを印刷するには、あらかじめそのファイルの *読み取り* 権限が必要です。印刷後にファイルを除去するには、そのファイルを含むディレクトリーへの *書き込み* 権限が必要です。

印刷ジョブを要求するには、次の情報を指定します。

- 印刷するファイルの名前
- 印刷キュー名
- 印刷部数
- リモート・ホスト上にファイルのコピーを作成するかどうか
- 印刷後にファイルを消去するかどうか
- ジョブ状況の通知を送信するかどうか
- ジョブ状況の通知をシステム・メールによって送信するかどうか
- バースト状況
- 「送り先」ラベルのユーザー名
- リモート印刷のコンソール肯定応答メッセージ

- リモート印刷のファイル肯定応答メッセージ
- 優先順位レベル

指定したファイルを印刷するための印刷ジョブを作成し、キューに入れるには、**qprt** コマンドを使用します。複数のファイルを指定すると、すべてのファイルをまとめて 1 つの印刷ジョブが構成されます。これらのファイルは、コマンド・ラインに指定した順序で印刷されます。

qprt コマンドの基本的なフォーマットは、次のとおりです。

```
qprt -PQueueName FileName
```

有用な **qprt** コマンド・フラグを次に示します。

項目	説明
-b *Number*	下部マージンを指定します。下部マージンは、各ページの下部に残すブランク行の数です。
-B *Value*	バースト・ページ (ミシン目で分離された連続用紙のページ) を印刷するかどうかを指定します。*Value* 変数は 2 文字からなる文字列です。先頭文字はヘッダー・ページに適用され、2 番目の文字はトレーラー・ページに適用されます。これら 2 つの文字はそれぞれ、次のいずれかです。 **a** それぞれの印刷ジョブにあるそれぞれのファイルごとに、(ヘッダーまたはトレーラー) ページを常に印刷します。 **n** (ヘッダーまたはトレーラー) ページを印刷しません。 **g** それぞれの印刷ジョブ (ファイルのグループ) ごとに、(ヘッダーまたはトレーラー) ページを 1 回ずつ印刷します。例えば、**-B ga** フラグは、ヘッダー・ページをそれぞれの印刷ジョブの最初に印刷し、トレーラー・ページをそれぞれの印刷ジョブ内でそれぞれのファイルの後に印刷するように指定します。 リモート印刷環境では、デフォルトはサーバー上のリモート・キューによって決まります。
-e *Option*	強調印刷を行うかどうかを指定します。 **+** 強調印刷を行うように指定します。 **!** 強調印刷を行わないように指定します。
-E *Option*	縦倍角印刷を行うかどうかを指定します。 **+** 縦倍角印刷を行うように指定します。 **!** 縦倍角印刷を行わないように指定します。
-f *FilterType*	印刷ファイルをプリンターに送信する前に通すフィルターを指定する、1 文字の ID。使用可能なフィルター ID は **p** (**pr** フィルターを呼び出す) と、**n** (**troff** コマンドからの出力を処理する) です。
-i *Number*	それぞれの行が指定した数のスペースだけインデントされます。 *Number* 変数は、**-w** フラグによって指定したページ幅の範囲内であることが必要です。
-K *Option*	縮刷機能を使用するかどうかを指定します。 **+** 縮刷機能を使用するように指定します。 **!** 縮刷機能を使用しないように指定します。
-l *Number*	ページ長を指定した行数に設定します。 *Number* 変数が 0 の場合、ページ長は無視され、出力は連続した 1 つのページと見なされます。ページ長は上部マージンと下部マージンを含み、用紙の印刷可能な長さを示します。
-L *Option*	ページ幅より広い行を次の行に折り返すか、右マージンで切り捨てるかを指定します。 **+** 長い行を次の行に折り返すように指定します。 **!** 長い行を折り返さず、右マージンで切り捨てるように指定します。

項目	説明
-N *Number*	印刷部数を指定します。このフラグを指定しない場合は、1 部が印刷されます。
-p *Number*	ピッチを *Number* 字/インチに設定します。*Number* の標準的な値は 10 および 12 です。印刷される文字の実際のピッチは、**-K** (縮刷) フラグと **-W** (横倍角) フラグの値にも影響されます。
-P *Queue*[:*QueueDevice*]	印刷キュー名とオプションのキュー・デバイス名を指定します。このフラグを指定しない場合は、デフォルト・プリンターが想定されます。
-Q *Value*	印刷ジョブの用紙サイズを指定します。用紙サイズの *Value* はプリンターによって異なります。標準的な値は、レターサイズ用紙の場合は **1**、リーガルの場合は **2** などです。特定の用紙サイズに割り当てられた値については、ご使用のプリンターのマニュアルを参照してください。
-t *Number*	上部マージンを指定します。上部マージンは、各ページの上部に残すブランク行の数です。
-w *Number*	ページ幅を *Number* 変数によって指定される文字数に設定します。ページ幅には、**-i** フラグによって指定されるインデント・スペースの数を含める必要があります。
-W *Option*	倍角印刷を行うかどうかを指定します。 **+** 倍角印刷を行うように指定します。 **!** 倍角印刷を行わないように指定します。
-z *Value*	右回りに 4 分の 1 回転の単位で、*Value* 変数の指定に従ってページ・プリンターの出力を回転します。長さ (**-l**) と幅 (**-w**) の値は、自動的に正しく調整されます。 **0** 縦長 **1** 右向き横長 **2** 下向き縦長 **3** 左向き横長
-# Value	特殊機能を指定します。 **j** 指定した印刷ジョブのジョブ番号を表示します。 **h** 印刷ジョブをキューに入れますが、再び解放されるまで **HELD** 状態にします。 **v** 指定したプリンター・バックエンド・フラグ値を妥当性検査します。この妥当性検査は、印刷ジョブを実行依頼する時点で、正しくないフラグ値を検査する際に便利です。妥当性検査を指定しない場合、誤ったフラグ値があると、後でジョブが実際に処理されるときに印刷ジョブが停止します。

次のリストに、**qprt** コマンド・フラグの使用法の例を示します。

- デフォルト値を使用して、デフォルト印刷キュー用に構成された最初の使用可能なプリンター上で `myfile` ファイルを印刷するように要求するには、次のように入力します。

  ```
  qprt myfile
  ```

- 特定のフラグ値を使用して、特定のキュー上で `myfile` ファイルを印刷するように要求し、印刷ジョブを実行依頼する時点でフラグ値を妥当性検査するには、次のように入力します。

  ```
  qprt -f p -e + -Pfastest -# v myfile
  ```

  この例では、`myfile` ファイルは **pr** フィルター・コマンド (**-f p** フラグ) によって処理され、強調モード (**-e +** フラグ) を使用して、**fastest** という名前のキュー (**-Pfastest** フラグ) 用に構成された最初の使用可能なプリンターで印刷されます。

- `myfile` ファイルをリーガル・サイズの用紙で印刷するには、次のように入力します。

  ```
  qprt -Q2 myfile
  ```

- new.index.c、print.index.c、および more.c の各ファイルを印刷キュー Msp1 に入れて、それぞれ 3 部印刷するには、次のように入力します。

  ```
  qprt -PMsp1 -N 3 new.index.c print.index.c more.c
  ```

- 3 つのファイル new.index.c、print.index.c、および more.c を連結して 3 部印刷するには、次のように入力します。

  ```
  cat new.index.c print.index.c more.c | qprt -PMsp1 -N 3
  ```

注: 基本オペレーティング・システムは、BSD UNIX 印刷コマンド (**lpr**)、および System V UNIX 印刷コマンド (**lp**) もサポートしています。完全な構文については、「*Commands Reference, Volume 3*」の **lpr** コマンド、および **lp** コマンドを参照してください。

完全な構文については、「*Commands Reference, Volume 4*」の **qprt** コマンドを参照してください。

SMIT を使用して印刷ジョブを要求することもできます。SMIT を使用して印刷ジョブを開始するには、次のように入力します。

```
smit qprt
```

印刷ジョブの取り消し (qcan コマンド)

qcan コマンドを使用して印刷ジョブを取り消すことができます。

- ローカル印刷ジョブの場合は、プリンターがシステムに物理的に接続されている必要があり、ネットワーク・プリンターの場合は、プリンターがネットワーク上で接続および構成されている必要があります。
- リモート印刷ジョブの場合は、リモート印刷サーバーと通信するようにシステムを構成する必要があります。

印刷ジョブを取り消す際には、ジョブが入っている印刷キューの名前、および取り消すジョブ番号を指定するようにプロンプトが出されます。

この手順は、ローカルおよびリモートの両方の印刷ジョブに適用されます。

ローカルまたはリモートの印刷キュー内にある特定のジョブ番号を取り消す、またはローカル印刷キュー内にあるすべてのジョブを取り消すには、**qcan** コマンドを使用します。ジョブ番号を判別するには、**qchk** コマンドを使用します。

qcan コマンドの基本的なフォーマットは、次のとおりです。

```
qcan -P QueueName -x JobNumber
```

完全な構文については、「*Commands Reference, Volume 4*」の **qcan** コマンドを参照してください。

次のリストに、**qcan** コマンドの使用法の例を示します。

- ジョブが送られているすべてのプリンター上でジョブ番号 **123** を取り消すには、qcan -x 123 と入力します。
- プリンター **lp0** のキューに入れられたジョブをすべて取り消すには、qcan -X -Plp0 と入力します。

注: 基本オペレーティング・システムは、BSD UNIX 印刷取り消しコマンド (**lprm**)、および System V UNIX 印刷取り消しコマンド (**cancel**) もサポートしています。完全な構文については、**lprm**コマンドおよび **cancel**コマンド (「 コマンド・リファレンス の」) を参照してください。

印刷ジョブの取り消し (SMIT)

SMIT を使用して印刷ジョブを取り消すことができます。

- ローカル印刷ジョブの場合は、プリンターがシステムに物理的に接続されている必要があり、ネットワーク・プリンターの場合は、プリンターがネットワーク上で接続および構成されている必要があります。
- リモート印刷ジョブの場合は、リモート印刷サーバーと通信するようにシステムを構成する必要があります。

SMIT を使用して印刷ジョブを取り消すには、次のように入力します。

```
smit qcan
```

その後、印刷ジョブ番号またはプリンターを選択できます。

印刷ジョブの優先順位付け (qpri コマンド)

ローカル・キュー上でのみ、**qpri** コマンドを使用してジョブ優先順位を割り当てることができます。

プリンターはシステムに物理的に接続されている必要があります。

値が大きいほど、印刷ジョブの優先順位が高いことを示しています。デフォルトの優先順位は **15** です。最高の優先順位は、大部分のユーザーの場合は **20**、root ユーザー特権のあるユーザーと **printq** グループ (グループ **9**) のメンバーの場合は **30** です。

注: リモート印刷ジョブに優先順位を割り当てることはできません。

実行依頼した印刷ジョブの優先順位を再度割り当てるには、**qpri** コマンドを使用します。root ユーザー権限がある場合、または **printq** グループに所属している場合は、ジョブが印刷キュー内にある間、任意のジョブに優先順位を割り当てることができます。

qpri コマンドの基本的なフォーマットは、次のとおりです。

```
qpri -# JobNumber -a PriorityLevel
```

完全な構文については、「*Commands Reference, Volume 4*」の **qpri** コマンドを参照してください。

次のリストに、**qpri** コマンドの使用法の例を示します。

- ジョブ番号 123 の優先順位番号を 18 に変更するには、次のように入力します。

  ```
  qpri -# 123 -a 18
  ```

- ローカル印刷ジョブを実行依頼するときに優先順位付けするには、次のように入力します。

  ```
  qprt -PQueueName -R PriorityLevel FileName
  ```

印刷ジョブの優先順位付け (SMIT)

ローカル・キュー上でのみ、ジョブ優先順位を割り当てることができます。

プリンターはシステムに物理的に接続されている必要があります。

値が大きいほど、印刷ジョブの優先順位が高いことを示しています。デフォルトの優先順位は **15** です。最高の優先順位は、大部分のユーザーの場合は **20**、root ユーザー特権のあるユーザーと **printq** グループ (グループ **9**) のメンバーの場合は **30** です。

注: リモート印刷ジョブに優先順位を割り当てることはできません。

SMIT を使用して印刷ジョブの優先順位を変更するには、次のように入力します。

```
smit qpri
```

別の印刷キューへの印刷ジョブの移動 (qmov コマンド)

qmov コマンドを使用して、キューの間で印刷ジョブを移動できます。

このタスクを実行するには、以下の前提条件が満たされている必要があります。

- プリンターはシステムに物理的に接続されている必要があります。
- 印刷ジョブの所有者であることが必要です。
- root 権限が必要です。
- **printq** グループのメンバーであることが必要です。

注: リモート印刷ジョブを別の印刷キューに移動することはできません。

印刷ジョブを別の印刷キューに移動するには、**qmov** コマンドを使用します。特定の印刷ジョブを移動することも、指定した印刷キューにあるすべての印刷ジョブを移動することもできます。また、特定のユーザーによって送信された印刷ジョブをすべて移動することもできます。印刷ジョブ番号を判別するには、**qchk** コマンドを使用します。詳しくは、**qchk** を参照してください。

qmov コマンドの基本的なフォーマットは、次のとおりです。

```
qmov  -mNewQueue {[ -#JobNumber ] [ -PQueue ] [ -uUser ]}
```

次のいずれかのコマンドを使用して、印刷ジョブを移動できます。

- `qmov -m DestinationQueue -# JobNumber`
- `qmov -m DestinationQueue -P Queue`
- `qmov -m DestinationQueue -u User`

完全な構文については、**qmov** コマンド コマンド・リファレンス のを参照してください。

次のリストに、**qmov** コマンドの使用法の例を示します。

- ジョブ番号 `280` を印刷キュー `hp2` に移動するには、次のように入力します。

  ```
  qmov -mhp2 -#280
  ```

- 印刷キュー `hp4D` にあるすべての印刷ジョブを印刷キュー `hp2` に移動するには、次のように入力します。

  ```
  qmov -mhp2 -Php4D
  ```

別の印刷キューへの印刷ジョブの移動 (SMIT)

プリンターがご使用のシステムに接続されている場合は、SMIT を使用して印刷ジョブを別の印刷キューに移動できます。

プリンターがご使用のシステムに物理的に接続されている場合は、System Management Interface Tool (SMIT) を使用して印刷ジョブを別の印刷キューに移動できます。

このタスクを実行するには、以下の前提条件が満たされている必要があります。

- プリンターはシステムに物理的に接続されている必要があります。
- 印刷ジョブの所有者であることが必要です。
- root 権限が必要です。

- **printq** グループのメンバーであることが必要です。

注: リモート印刷ジョブを別の印刷キューに移動することはできません。

次のコマンドを入力します。

```
smit qmov
```

印刷ジョブの保留と解放 (qhld コマンド(

qhld コマンドを使用して、印刷ジョブを保留および解放できます。

注: リモート印刷ジョブを保留および解放することはできません。

このタスクを実行するには、以下の前提条件が満たされている必要があります。

- プリンターはシステムに物理的に接続されている必要があります。
- 印刷ジョブの所有者であることが必要です。
- root 権限が必要です。
- **printq** グループのメンバーであることが必要です。

qhld コマンドを使用して、印刷ジョブを送信した後で保留にします。特定の印刷ジョブを保留にすることも、指定した印刷キューにあるすべての印刷ジョブを保留することもできます。印刷ジョブ番号を判別するには、**qchk** コマンドを入力します。**qhld** コマンドの基本的なフォーマットは、次のとおりです。

```
qhld [ -r ] {[ -#JobNumber ] [ -PQueue ] [ -uUser ]}
```

次のいずれかのコマンドを使用して、印刷ジョブを保留できます。

- `qhld -# JobNumber`
- `qhld -P Queue`
- `qhld -u User`

次のいずれかのコマンドを使用して、印刷ジョブを解放できます。

- `qhld -r -# Jobnumber`
- `qhld -r -P Queue`
- `qhld -r -u User`

次のリストに、**qhld** コマンドの使用法の例を示します。

1. ジョブがどの印刷キューにあってもジョブ番号 452 を保留するには、次のコマンドを入力します。
   ```
   qhld -#452
   ```
2. 印刷キュー hp2 に入っているすべてのジョブを保留するには、次のコマンドを入力します。
   ```
   qhld -Php2
   ```
3. ジョブがどの印刷キューにあってもジョブ番号 452 を解放するには、次のコマンドを入力します。
   ```
   qhld -#452 -r
   ```
4. 印刷キュー hp2 に入っているすべてのジョブを解放するには、次のコマンドを入力します。
   ```
   qhld -Php2 -r
   ```

印刷ジョブの保留と解放 (SMIT)

SMIT を使用して、印刷ジョブを保留および解放できます。

印刷ジョブを保留または解放するには、次のいずれかのユーザーであることが必要です。

- 印刷ジョブ所有者
- root 権限のあるユーザー
- **printq** グループのメンバー

印刷ジョブを保留または解放するには、次のように入力します。

- `smit qhld`

印刷ジョブの状況の検査 (qchk コマンド)

qchk コマンドを使用して、印刷ジョブの状況を検査できます。

- ローカル印刷ジョブの場合は、プリンターがシステムに物理的に接続されている必要があり、ネットワーク・プリンターの場合は、プリンターがネットワーク上で接続および構成されている必要があります。
- リモート印刷ジョブの場合は、リモート印刷サーバーと通信するようにシステムを構成する必要があります。

指定した印刷ジョブ、印刷キュー、またはユーザーに関する現在の状況情報を表示するには、**qchk** コマンドを使用します。

qchk コマンドの基本的なフォーマットは、次のとおりです。

```
qchk -P QueueName -# JobNumber -u OwnerName
```

注: 基本オペレーティング・システムは、BSD UNIX 印刷キュー検査コマンド (**lpq**)、および System V UNIX 印刷キュー検査コマンド (**lpstat**) もサポートしています。完全な構文については、**lpq**コマンドおよび **lpstat**コマンド (「 コマンド・リファレンス の」) を参照してください。

次のリストに、**qchk** コマンドの使用法の例を示します。

- デフォルト印刷キューを表示するには、次のように入力します。

  ```
  qchk -q
  ```

- 画面を **5** 秒ごとに最新表示しながら、すべてのキューの詳細な状況を空になるまで表示するには、次のように入力します。

  ```
  qchk -A -L -w 5
  ```

- 印刷キュー **lp0** の状況を表示するには、次のように入力します。

  ```
  qchk -P lp0
  ```

- ジョブ番号 **123** の状況を表示するには、次のように入力します。

  ```
  qchk -# 123
  ```

- すべてのキューにあるすべてのジョブの状況を検査するには、次のように入力します。

  ```
  qchk -A
  ```

印刷キューの状況条件

印刷キューの状況条件には、次のものがあります。

項目	説明
DEV_BUSY	次のことを示します。 • プリンター・デバイス (**lp0**) に対して複数のキューが定義されており、別のキューがプリンター・デバイスを現在使用しています。 • **qdaemon** はプリンター・ポート・デバイス (**lp0**) の使用を試みましたが、別のアプリケーションがそのプリンター・デバイスを現在使用しています。 **DEV_BUSY** からリカバリーするには、キューまたはアプリケーションがプリンター・デバイスを解放するまで待つか、プリンター・ポートを使用しているジョブまたはプロセスを取り消してください。
DEV_WAIT	プリンターがオフラインになっているか、用紙切れまたはジャム、ケーブルのゆるみ、不良、または配線の誤りが起きているために、キューがプリンターを待機していることを示します。 **DEV_WAIT** からリカバリーするには、待機の原因になっている問題を訂正してください。診断テストを行うには、**enq** コマンドを使用して、キューに入れられた印刷中または **DOWN** 状態のジョブすべてを、**DEV_WAIT** キューから別のキューに移動すると便利です。問題を訂正した後、まだ印刷されていないジョブを元のキューに戻すことができます。
DOWN	キューは通常、**DEV_WAIT** 状態になった後で **DOWN** 状態に移行します。この状態は、正しいシグナリングが行われていないために、プリンターのデバイス・ドライバーがプリンターの存在を判別できないときに発生します。ただし、一部のプリンターはオフラインであることを示すシグナルをキューイング・システムに送る機能を備えておらず、代わりにオフであることを示すシグナルを送信します。プリンター・デバイスがオフであることを示すシグナルを出している場合、またはオフであると見なされる場合に、キューは **DOWN** 状態になります。 **DOWN** 状態からリカバリーするには、キューをダウン状態にした問題を訂正し、システム管理者にキューの再起動を依頼してください。キューを再び使用できるようにするには、キューを手動で起動する必要があります。
HELD	印刷ジョブが保留中であることを示します。解放されるまで、印刷ジョブはスプーラーによって処理できません。
QUEUED	印刷ファイルがキューに入れられており、印刷を順次待機していることを示します。
READY	キューに関連した準備がすべてできており、ジョブをキューに入れて印刷できることを示しています。
RUNNING	印刷ファイルが印刷中であることを示します。

印刷ジョブの状況の検査 (SMIT)

smit コマンドを使用して、印刷ジョブの状況を検査できます。

- ローカル印刷ジョブの場合は、プリンターがシステムに物理的に接続されている必要があり、ネットワーク・プリンターの場合は、プリンターがネットワーク上で接続および構成されている必要があります。
- リモート印刷ジョブの場合は、リモート印刷サーバーと通信するようにシステムを構成する必要があります。

指定したジョブ番号、キュー、プリンター、またはユーザーに関する現在の状況情報を表示できます。
SMIT を使用して印刷ジョブの状況を検査するには、次のように入力します。

```
smit qchk
```

印刷キューの状況条件

印刷キューの状況条件には、次のものがあります。

項目	説明
DEV_BUSY	次のことを示します。 • プリンター・デバイス (**lp0**) に対して複数のキューが定義されており、別のキューがプリンター・デバイスを現在使用しています。 • **qdaemon** はプリンター・ポート・デバイス (**lp0**) の使用を試みましたが、別のアプリケーションがそのプリンター・デバイスを現在使用しています。 **DEV_BUSY** からリカバリーするには、キューまたはアプリケーションがプリンター・デバイスを解放するまで待つか、プリンター・ポートを使用しているジョブまたはプロセスを取り消してください。
DEV_WAIT	プリンターがオフラインになっているか、用紙切れまたはジャム、ケーブルのゆるみ、不良、または配線の誤りが起きているために、キューがプリンターを待機していることを示します。 **DEV_WAIT** からリカバリーするには、待機の原因になっている問題を訂正してください。診断テストを行うには、**enq** コマンドを使用して、キューに入れられた印刷中または **DOWN** 状態のジョブすべてを、**DEV_WAIT** キューから別のキューに移動すると便利です。問題を訂正した後、まだ印刷されていないジョブを元のキューに戻すことができます。
DOWN	キューは通常、**DEV_WAIT** 状態になった後で **DOWN** 状態に移行します。この状態は、正しいシグナリングが行われていないために、プリンターのデバイス・ドライバーがプリンターの存在を判別できないときに発生します。ただし、一部のプリンターはオフラインであることを示すシグナルをキューイング・システムに送る機能を備えておらず、代わりにオフであることを示すシグナルを送信します。プリンター・デバイスがオフであることを示すシグナルを出している場合、またはオフであると見なされる場合に、キューは **DOWN** 状態になります。 **DOWN** 状態からリカバリーするには、キューをダウン状態にした問題を訂正し、システム管理者にキューの再起動を依頼してください。キューを再び使用できるようにするには、キューを手動で起動する必要があります。
HELD	印刷ジョブが保留中であることを示します。解放されるまで、印刷ジョブはスプーラーによって処理できません。
QUEUED	印刷ファイルがキューに入れられており、印刷を順次待機していることを示します。
READY	キューに関連した準備がすべてできており、ジョブをキューに入れて印刷できることを示しています。
RUNNING	印刷ファイルが印刷中であることを示します。

印刷するファイルのフォーマット設定 (pr コマンド)

pr コマンドを使用して、プリンターに送るファイルの簡単なフォーマット設定を行うことができます。

pr コマンドの出力を **qprt** コマンドにパイピングして、テキストのフォーマットを設定できます。

pr コマンドの有用なフラグには、次のものがあります。

項目	説明
-d	出力をダブルスペースにします。
-h " *String*"	ページ・ヘッダーとして、ファイル名の代わりに、指定した文字列を引用符 (" ") で囲んで表示します。フラグと文字列はスペースで区切ります。
-l *Lines*	66 行のデフォルトを指定変更し、ページ長を *Lines* 変数に指定した行数に再設定します。 *Lines* の値がヘッダーとトレーラーの縦の長さの合計 (行数) より小さい場合、ヘッダーとトレーラーは抑止されます (**-t** フラグを有効にした場合と同様に)。
-m	ファイルをマージします。*File* 変数によって指定された各ファイルから 1 行ずつ、列位置数に応じて等しい固定幅のテキスト列に横並びに書き込むように、**pr** コマンドの標準出力のフォーマットが設定されます。このフラグは、**-Column** フラグと一緒に使用することはできません。
-n [*Width*][*Character*]	*Width* 変数によって指定された桁数に基づいて、行番号を付けます。デフォルトは 5 桁です。 *Character* (数字以外の任意の文字) 変数を指定すると、その文字が行番号に付加されて、その行にある後続の文字と行番号を区切ります。デフォルトの分離文字は、ASCII タブ文字です。
-o *Offset*	*Offset* 変数によって指定された数の文字位置だけ、それぞれの行をインデントします。1 行あたりの文字位置の合計数は、幅とオフセットの合計です。*Offset* のデフォルト値は 0 です。
-s *Character*	自動的に決まる数のスペースではなく、*Character* 変数に指定した単一文字によって列を区切ります。*Character* のデフォルト値は、ASCII タブ文字です。
-t	5 行の識別ヘッダー、および 5 行のフッターを表示しません。各ファイルの最終行の後、ページの末尾でスペーシングを行わずに停止します。

項目	説明
-w *Width*	1 行あたりの列位置数を、*Width* 変数に指定した値に設定します。デフォルト値は、等幅の複数列出力の場合は 72 です。それ以外の場合、限界はありません。**-w** フラグを指定せず、**-s** フラグを指定した場合は、デフォルトの幅は 512 列位置になります。
-*Column*	列数を *Column* 変数に指定した値に設定します。デフォルト値は 1 です。このオプションを **-m** フラグと一緒に使用しないでください。複数列出力の場合は、**-e** フラグと **-i** フラグが前提になります。テキスト列はページ長を超えてはなりません (**-l** フラグを参照)。このフラグを **-t** フラグと一緒に使用すると、出力の書き込みに最小限の行数が使用されます。
+*Page*	*Page* 変数によって指定されたページ番号から表示を開始します。デフォルト値は 1 です。

完全な構文については、「 コマンド・リファレンス の」の **pr** コマンドを参照してください。

次に、**pr** コマンド・フラグの使用例のリストを示します。

- プリンターで prog.c という名前のファイルを見出しとページ番号付きで印刷するには、次のように入力します。

  ```
  pr prog.c | qprt
  ```

  このコマンドは、ページ見出しを prog.c に追加して、これを **qprt** コマンドに送ります。見出しは、ファイルの最終変更日、ファイル名、およびページ番号で構成されます。
- prog.c という名前のファイルの表題を指定するには、次のように入力します。

  ```
  pr -h "MAIN PROGRAM" prog.c | qprt
  ```

  これにより、ファイル名の代わりに MAIN PROGRAM という表題を付けて、prog.c が印刷されます。この場合も、変更日とページ番号は印刷されます。
- word.lst という名前のファイルを複数列に印刷するには、次のように入力します。

  ```
  pr -3 word.lst | qprt
  ```

  これにより、word.lst ファイルが垂直方向 3 列に印刷されます。
- 複数のファイルを用紙上で横並びに印刷するには、次のように入力します。

  ```
  pr -m -h "Members and Visitors" member.lst visitor.lst | qprt
  ```

  これにより、member.lst および visitor.lst が、**Members and Visitors** という表題とともに横並びに印刷されます。
- prog.c という名前のファイルを後で使用するために変更するには、次のように入力します。

  ```
  pr -t -e prog.c > prog.notab.c
  ```

  このコマンドにより、prog.c 内のタブ文字がスペースに置換され、結果が prog.notab.c に書き込まれます。タブ位置は、列 9、17、25、33 のように続きます。**-e** フラグの指示により **pr** コマンドはタブ文字を置き換え、**-t** フラグはページ見出しを抑止します。
- myfile という名前のファイルを 7 ポイントのテキストで横長に 2 列に印刷するには、次のように入力します。

  ```
  pr -l66 -w172 -2 myfile | qprt -z1 -p7
  ```

PostScript プリンターでの ASCII ファイルの印刷

テキスト・フォーマット設定システムには **enscript** フィルターが組み込まれており、ASCII 印刷ファイルを PostScript プリンターで印刷するための PostScript ファイルに変換できます。

- プリンターはシステムに物理的に接続されている必要があります。

- プリンターの構成および定義が済んでいる必要があります。
- テキスト・フォーマット設定サービスのトランスクリプト部分をインストールする必要があります。

enscript フィルターは、印刷ジョブを PostScript 印刷キューに実行依頼するときに、**qprt -da** コマンドによって呼び出されます。**qprt** コマンドにいくつかのフラグを指定して、ASCII ファイルを PostScript 印刷キューに実行依頼する際の出力をカスタマイズできます。

項目	説明
-1+	ページ見出しを追加します。
-2+	出力のフォーマットを 2 列に設定します。
-3+	ページ見出し、日付、およびページ番号を飾り付きのスタイルで印刷します。これは「装飾」モードとも呼ばれます。
-4+	印刷不能文字を含む場合であっても、ファイルを印刷します。
-5+	フォントに含まれない文字をリストします。
-h *string*	ページ見出しに使用する文字列を指定します。このフラグを指定しない場合、見出しはファイル名、変更日、およびページ番号で構成されます。
-l *value*	1 ページあたりの最大印刷行数を指定します。ポイント・サイズによっては、実際に印刷される 1 ページあたりの行数がこれより少なくなることがあります。
-L!	ページ幅より長い行を切り捨てます。
-p	ポイント・サイズを指定します。このフラグを指定しない場合は、ポイント・サイズ 10 が想定されます。ただし、2 列回転モード (**-2+ -z1**) を指定した場合は異なり、この場合は値 7 が使用されます。
-s	フォント・スタイルを指定します。このフラグを指定しない場合は、Courier フォントが使用されます。PostScript プリンターが、指定したフォントにアクセスできることが必要です。受け入れ可能な値は次のとおりです。 Courier-Oblique Helvetica Helvetica-Oblique Helvetica-Narrow Helvetica-Narrow-Oblique NewCenturySchlbk-Italic Optima Optima-Oblique Palatino-Roman Palatino-Italic Times-Roman Times-Italic
-z1	出力を 90 度回転します (横長モード)。

次のリストに、**qrpt** コマンド・フラグの使用法の例を示します。

- ASCII ファイル `myfile.ascii` を **Msps1** という名前の PostScript プリンターに送るには、次のように入力します。

  ```
  qprt -da -PMsps1 myfile.ascii
  ```

- ASCII ファイル `myfile.ascii` を **Msps1** という名前の PostScript プリンターに送り、Helvetica フォントで印刷するには、次のように入力します。

  ```
  qprt -da -PMsps1 -sHelvetica myfile.ascii
  ```

- ASCII ファイル `myfile.ascii` を **Msps1** という名前の PostScript プリンターに送り、ポイント・サイズ 9 で印刷するには、次のように入力します。

  ```
  qprt -da -PMsps1 -p9 myfile.ascii
  ```

ASCII から PostScript への変換の自動化

PostScript 印刷キューに対して実行依頼された ASCII 印刷ファイルを検出し、これらの ASCII ファイルを PostScript プリンター用の PostScript に自動的に変換するように、システムを構成できます。

PostScript ファイルを生成するアプリケーションの多くは、 PostScript ファイルの先頭 2 文字を **%!** にするという規則に従っています。これは、印刷ファイルが PostScript 印刷ファイルであることを示しています。PostScript 印刷キューに対して実行依頼された ASCII 印刷ファイルを検出し、これらの ASCII ファイルを PostScript プリンターに送る前に PostScript ファイルに自動的に変換するようにシステムを構成するには、次の手順で行います。

1. プロンプトで、`smit chpq` と入力します。
2. PostScript キュー名を入力するか、「**List (リスト)**」機能を使用してキューのリストから選択します。
3. 「**Printer Setup (プリンターの設定)**」メニュー・オプションを選択します。
4. 「**AUTOMATIC detection of print file TYPE to be done? (印刷ファイル・タイプの自動検出を行いますか)**」フィールドの値を「**yes (はい)**」に変更します。

これで、次のコマンドのいずれを実行しても、ASCII ファイルが PostScript ファイルに変換され、PostScript プリンターで印刷されるようになります。`myfile.ascii` を変換するには、コマンド・ラインで次のいずれかを入力します。

```
qprt -Pps myfile.ps myfile.ascii
lpr -Pps myfile.ps myfile.ascii
lp -dps myfile.ps myfile.acsii
```

ここで、*ps* は PostScript 印刷キューです。

印刷ファイル・タイプの自動判別の指定変更

場合によっては、印刷ファイル・タイプの自動判別を指定変更する必要が生じることがあります。

-d フラグと **-s** フラグを使用して、PostScript 印刷の印刷ファイル・タイプの自動判別を指定変更できます。 **-d** フラグはデフォルトの印刷ファイル・タイプを指定変更し、**-s** フラグは PostScript 印刷を指定します。

次のような状況で、PostScript 印刷の印刷ファイル・タイプの自動判別を指定変更する必要が生じることがあります。

- **%!** から始まらない `myfile.ps` という名前の PostScript ファイルを印刷するには、次のコマンドを入力します。

  ```
  qprt -ds -Pps myfile.ps
  ```

- **%!** から始まる `myfile.ps` という名前の PostScript ファイルのソース・リストを印刷するには、次のコマンドを入力します。

  ```
  qprt -da -Pps myfile.ps
  ```

印刷用コマンドの要約

いくつかのコマンドが、印刷および印刷キューの管理に使用されます。

項目	説明
cancel	ライン・プリンターへの要求を取り消します。
lp	ライン・プリンターに要求を送信します。
lpq	スプール・キューを検査します。
lpr	印刷ジョブをキューに入れます。
lprm	ライン・プリンターのスプール・キューからジョブを除去します。
lpstat	ライン・プリンターの状況情報を表示します。
pr	ファイルを標準出力に書き込みます。
qcan	印刷ジョブを取り消します。
qchk	印刷キューの状況を表示します。
qhld	印刷ジョブを保留または解放します。
qmov	印刷ジョブを別の印刷キューに移動します。
qpri	印刷キュー内のジョブを優先順位付けします。
qprt	印刷ジョブを開始します。

印刷の管理

プリンターを操作する際には、システム管理者はスプーラー、実プリンター、仮想プリンター、バックエンド、およびキューを管理する必要があります。これらはすべて、プリンター・サブシステムを構成する部分です。

プリンターに関連したシステム管理情報には、次のものがあります。

印刷プロセス

ファイルを印刷する際に、システムはプリンターにコードを送信します。一部のコードは、特定の英字や数字などの文字を印刷します。その他のコードは、特定の特性を持つ文字に下線を付けたり、ページ長を調整したりして、文字やファイルの印刷方法を制御します。

異なる文字コードをプリンターに送信したい場合、例えば **that** という単語を **this** に変更する場合などは、基礎のコードについての知識は不要で、単にファイルを編集するだけです。

ただし、プリンターの作動方法を変更するには、ファイルを印刷するときに行われる処理、プリンターに制御情報を送信するために使用できるオプション、および制御できるプリンター特性を理解している必要があります。

プリンターにファイルを送信するには、System Manager Interface Tool (SMIT)、または **qprt** コマンドを使用します。また、SMIT を使用して、印刷ジョブの取り消しまたは優先順位付けを行うことができます。

どちらの印刷方法を使用する場合も、ファイルがプリンターに直接送られることはありません。これら 3 つの方式ではすべて、まず **enq** コマンドを呼び出して、印刷要求をキューに入れる必要があります。プリンターが使用可能になるまで、印刷要求はキューに残ります。プリンターが使用可能になった時点で、**qdaemon** コマンドは (プリンター入出力バックエンド) **piobe** コマンドを実行します。**piobe** コマンドは、ファイルを処理して制御情報とともにプリンターに送信します。その後プリンターは、**qprt** コマンドによって指定されたファイルの内容と制御情報を含むデータ・ストリームを受信します。

印刷プロセス制御

次の方法で、プリンター制御情報をプリンター・データ・ストリームに追加できます。

- ファイルにプリンター制御コードを組み込む。

このためには、印刷キュー・データ・ストリームを**パススルー**に設定します (つまり、**d=p**)。詳しくは、79 ページの『プリンター・コロン・ファイルの規則』を参照してください。

そのファイルに固有のプリンター制御情報をすべて組み込みます。例えば、資料の表題に下線を付けたり、段落を太字の書体で印刷したりするには、プリンター制御情報を開始および停止するコードを適切な位置に挿入します。

ワード・プロセッサーなどのアプリケーション・プログラムを使用すると、特定のプリンター制御情報をファイルに挿入できます。ただし、プリンターをアプリケーション・プログラムから構成できない場合は、システム・エディターを使用してプリンター制御コードを挿入する必要があります。プリンター制御コードは、プリンターに付属しているか、プリンターを購入した販売業者、またはプリンターの製造メーカーから入手できます。

- **qprt** コマンドを使用して、コマンド・フラグを指定する。

  qprt コマンド、または SMIT の「**Start a Print Job (印刷ジョブの開始)**」オプションは、次のように、プリンターの操作を制御するいくつかのフラグを認識します。

  – 縮刷、強調、横倍角、および二重印刷の指定
  – 指定した色での印刷
  – マージンの設定
  – 縦方向の行/インチ数の設定
  – 改行または垂直タブ制御に対する印刷行の水平位置の維持

  単一の印刷ジョブに対して、特定の印刷特性を指定できます。例えば、ピッチを設定するための **qprt** コマンド・フラグは **-p** *Number* です (ここで、*Number* は字/インチ数)。**qprt** コマンドの標準設定が 10 字/インチで、ただし `printtest` ファイルに対しては 12 字/インチにする必要がある場合は、次のコマンドを入力します。

  ```
  qprt -p 12 printtest
  ```

  コマンド・ラインのフラグは、このジョブに対する標準の **qprt** コマンドの設定を指定変更します。標準の **qprt** コマンドのピッチ設定は 10 のままです。

- 標準の **qprt** コマンドの設定を変更する。

  SMIT または **lsvirprt** コマンドを使用することができます。

  注: root 権限を持っているか、**printq** グループのメンバーであることが必要です。

  例えば、標準ピッチを 12 字/インチに変更するには、**chvirprt** コマンド、または SMIT を実行します。表示されたリストからプリンターを選択し、属性名と値を等号 (=) で区切って入力します。

  qprt コマンド・フラグの属性名は、フラグ文字です。 `p=12` を指定して、標準ピッチを 12 に変更できます。

プリンターの初期構成

プリンターを構成するプロセスを行ってから、別のプロセスによって印刷キューを追加できます。使用するタスクは、システムへのプリンターの接続方法によって異なります。

また、印刷キューを追加せずにプリンターを構成することもできます。

注: AIX オペレーティング・システムは、USB (Universal Serial Bus) を経由でシステムに接続するプリンターをサポートしていません。

次に、これらのタスクを行う方法を説明します。

構成ファイルの変更

`/etc/qconfig` ファイルを編集できます。

enq コマンドと **qdaemon** コマンドの両方とも、開始時に `/etc/qconfig` ファイルを読み取ります。**qdaemon** コマンドはシステムの始動時に開始され、**enq** コマンドはいずれかのユーザーが印刷ジョブを要求するたびに開始されます。このため、`/etc/qconfig` ファイルを変更すると、**enq** コマンドは次回実行されるときに構成ファイルの新しいバージョンを読み取ります。

いずれかのキューにアクティブ・ジョブが存在するときは、`/etc/qconfig` ファイルを編集しないでください。編集には、手動での編集と、**mkque**、**rmque**、**chque**、**mkquedev**、**rmquedev**、または **chquedev** の各コマンドを使用した編集の両方が含まれます。手動で編集する必要がなければ、`/etc/qconfig` ファイルを変更する際にはこれらのコマンドを使用してください。手動で編集するには、まず `enq -G` コマンドを発行して、すべてのジョブが完了した後でキューイング・システムと **qdaemon** コマンドを停止します。その後、`/etc/qconfig` ファイルを編集し、新しい構成を使用して **qdaemon** コマンドを再始動します。

仮想プリンターと印刷キュー

仮想プリンターは、印刷キューに関連付けられます。

プリンターがサポートするそれぞれのデータ・ストリームごとに、印刷キューを定義できます。複数の印刷キューが同じ実プリンターを使用できます。

印刷キューを追加するには、SMIT の「**Add a Print Queue (印刷キューの追加)**」オプション、または **mkque**、**mkquedev**、および **mkvirprt** の各コマンドを使用します。

印刷ジョブを実行依頼する際には、印刷キューを直接または間接的に指定する必要があります。印刷ジョブに対して特定のプリンターを指定するには、印刷キュー名にコロンとプリンター・デバイス名を追加します。印刷ジョブに対してプリンターを指定しない場合、スプーラーは印刷キューに関連付けられた最初の使用可能なプリンターを選択します。複数のプリンターが印刷キューに関連付けられている場合は、いずれかのプリンターが使用されます。

例えば、IBM® プロプリンターの場合は、それぞれの実プリンターに対して定義する必要がある印刷キューはただ 1 つです。これは、プロプリンターがただ 1 つのデータ・ストリーム (IBM 拡張 ASCII) をサポートするからです。 IBM 4216 モデル 031 パーソナル・ページ・プリンターの場合は、複数の印刷キューを定義する必要があります。プリンターがサポートするそれぞれのデータ・ストリームごとに、印刷キューを 1 つずつ定義できます。PostScript、プロプリンター、HP LaserJet、および Diablo 630 の各エミュレーションに対して印刷キューを定義できます。 4 つの印刷キューの出力先は、すべて同じ実プリンター、つまり 4216 モデル 031 です。

ローカル・プリンターの構成とキューの追加

SMIT を使用して、ローカル・プリンターを構成し、キューを追加できます。

- ご使用のプリンターの資料をお読みください。プリンターを接続して構成するには、プリンター固有の情報が必要になる場合があります。
- システムの構成を検討します。プリンターを接続するパラレル・ポートまたはシリアル・ポートを判別します。

- root 権限が必要です。

注: AIX オペレーティング・システムは、USB (Universal Serial Bus) を経由でシステムに接続するプリンターをサポートしていません。

ローカル・プリンターを構成し、さらに印刷ジョブのスプーリングを行う必要がある場合は、次の手順を使用します。

1. ローカル・ホストのシリアル・ポートまたはパラレル・ポートに、プリンターを直接接続します。
   a. システム・プロンプトで **shutdown** コマンドを使用して、システムを停止します。
   b. システムと外部デバイスの電源を切ります。
   c. プリンターを適切なシリアル・ポートまたはパラレル・ポートに接続します。
   d. プリンターの資料の説明に従ってプリンターをセットアップします。
   e. システムを再始動します。
2. システム・プロンプトで「**smit chpq**」と入力し、指示に従ってこの作業を完了します。

リモート・プリンターの構成とキューの追加

リモート・プリンターを構成し、キューを追加できます。

リモート・ホストは印刷サーバーとして構成する必要があります。

リモート・プリンターを構成し、さらに印刷ジョブのスプーリングを行う必要がある場合は、次の手順を使用します。

注: 印刷キューを追加せずにプリンターを構成する場合は、18 ページの『キューを追加しない場合のプリンターの構成 (SMIT)』、および 18 ページの『キューを追加しない場合のプリンターの構成 (qprt コマンド)』を参照してください。

1. システム・プロンプトで「**smit mkpq**」と入力し、指示に従ってこの作業を完了します。
2. 印刷キューが正常に作成された後、その名前が表示されます。終了する前に、エラー・メッセージがある場合は必ず記録しておいてください。
3. 「**smit chpq**」を使用して、新規印刷キューをカスタマイズします。

ネットワーク・プリンターの構成とキューの追加

ネットワーク・プリンターを構成し、キューを追加できます。

- ご使用のプリンターの資料をお読みください。プリンターを接続して構成するには、プリンター固有の情報が必要になる場合があります。
- Hewlett-Packard JetDirect カードの資料をお読みください。
- root 権限が必要です。

ネットワーク・プリンターを構成し、さらに印刷ジョブのスプーリングを行う必要がある場合は、次の手順を使用します。

1. システム・プロンプトで「**smit mkpq**」と入力し、指示に従ってこの作業を完了します。
2. 印刷キューが正常に作成された後、その名前が表示されます。終了する前に、エラー・メッセージがある場合は必ず記録しておいてください。
3. 「**smit chpq**」を使用して、新規印刷キューをカスタマイズします。

/dev ディレクトリー内のファイルに対する印刷キューの構成

/dev ディレクトリー内のファイルに対して、印刷キューを構成できます。

- ご使用のプリンターの資料をお読みください。プリンターを接続して構成するには、プリンター固有の情報が必要になる場合があります。
- システムの構成を検討します。
- root 権限が必要です。

/dev ディレクトリー内のファイルに対して印刷キューを構成する必要がある場合は、次の手順を使用します。

1. システム・プロンプトで「**smit mkpq**」と入力し、指示に従ってこの作業を完了します。
2. 印刷キューが正常に作成された後、その名前が表示されます。終了する前に、エラー・メッセージがある場合は必ず記録しておいてください。
3. 「**smit chpq**」コマンドを使用して、新規印刷キューをカスタマイズします。

キューを追加しない場合のプリンターの構成 (qprt コマンド)

プリンターまたはプロッターの印刷ジョブをスプールしない場合に、qprt コマンドを使用して、キューを追加せずにプリンターまたはプロッターを追加できます。

プリンター・ポートを構成する前に、プリンターまたはプロッターがシステムに物理的に接続されている必要があります。

プリンターまたはプロッターの追加のみを行い、印刷ジョブをスプールしない場合は、次の手順を使用します。

注: プリンターを構成する際に印刷キューの追加も行う場合は、 15 ページの『プリンターの初期構成』を参照してください。

次の例では、**qprt** コマンド、**enq** コマンド、**lp** コマンド、または **lpr** コマンドを使用して、印刷ジョブをキューに入れる方法を説明します。 **lp** コマンドには **-d** フラグを指定する (**-P** フラグの代わりに) 必要がある点を除いて、3 つのキューイング・コマンドの構文はすべて同じです。

```
Command -P QueueName FileName
```

ここで

項目	説明
QueueName	印刷キューの名前。
FileName	印刷するファイルの名前。

次の例では、**qprt** コマンドの使用法を示します。

```
qprt -Pfastest myfile
```

指定できるその他のフラグについては、個別のキューイング・コマンドを参照してください。

キューを追加しない場合のプリンターの構成 (SMIT)

プリンターまたはプロッターの印刷ジョブをスプールしない場合に、SMIT を使用して、キューを追加せずにプリンターまたはプロッターを追加できます。

プリンター・ポートを構成する前に、プリンターまたはプロッターがシステムに物理的に接続されている必要があります。

プリンターまたはプロッターの追加のみを行い、印刷ジョブをスプールしない場合は、次の手順を使用します。

注: プリンターを構成する際に印刷キューの追加も行う場合は、 15 ページの『プリンターの初期構成』を参照してください。

1. システム・プロンプトで、次のように入力します。

   ```
   smit pdp
   ```

2. 「**Add a Printer/Plotter (プリンター/プロッターの追加)**」を選択します。
3. プロンプトに応じて追加情報を入力します。

印刷キューの操作

印刷キューの操作に関連した手順がいくつかあります。

ここでは、次の手順について説明します。

印刷キュー・デバイスの追加

印刷キュー・デバイスを追加できます。

この作業を行うには、root 権限が必要です。

印刷キュー・デバイスを追加するには、次の手順を使用します。

システム・プロンプトで「**smit mkquedev**」と入力し、指示に従ってこの作業を完了します。

- **mkquedev** コマンドを使用して、次のように印刷キュー・デバイスを追加することもできます。

  ```
  mkquedev -d QueueName -q QueueName -a Attribute = Value
  ```

  印刷キュー・デバイスを完全に構成するには、**-a** フラグを複数回使用する必要が生じることがあります。

5080 によるプロッター・サポートの追加

プロッター・サポートを追加するための手順について説明します。

- プロッターはシステムに物理的に接続されている必要があります。
- プロッター・デバイスが既に追加済みであることが必要です。

プロッター・サポートを追加するには、次の手順を使用します。この手順を使用してプロッターを識別した後、**enq** コマンドを使用して、5080 接続アダプターのプロッター・バックエンドにアクセスできます。

1. システム・プロンプトで、`smit pq_mklque` と入力します。
2. 「**NAME of Queue to Add (追加するキュー名)**」プロンプトで、`plta` と入力します。 これにより、シリアル・ポート **a** が定義されます。
3. 「**NAME of Device to Add (追加するデバイス名)**」プロンプトで、`plota` と入力します。 これにより、シリアル・ポート **a** が定義されます。
4. 「**BACKEND PROGRAM Pathname (バックエンド・プログラムのパス名)**」に、`/usr/lib/lpd/plotgbe -gswa 9600` と入力します。
5. 「**NAME of Queue to Add (追加するキュー名)**」プロンプトで、次のように入力します。

   ```
   pltb
   ```

   これにより、シリアル・ポート **b** が定義されます。

6. 「**NAME of Queue to Add (追加するキュー名)**」プロンプトで、`pltb` と入力します。 これにより、シリアル・ポート **b** が定義されます。
7. 「**BACKEND PROGRAM Pathname (バックエンド・プログラムのパス名)**」に、`/usr/lib/lpd/plotgbe -gswa 9600` と入力します。
8. ポート **a** またはポート **b** のどちらかにプロッターを接続します。

mkque コマンドと **mkquedev** コマンドを使用して、この作業を実行することもできます。プロッター・サポートを追加するには、他のフラグが必要です。詳しくは、**mkque** コマンド、および **mkquedev** コマンドを参照してください。

プロッター・セットアップ・ファイルの作成

プロッターにプロット・ファイルを送るには、使用するペーシング・プロトコルのタイプを示す命令を含むスペシャル・ファイルが必要です。

次の命令は、Xon/Xoff ペーシング・プロトコルと、Data Transmit Rate (DTR) ペーシング・プロトコルを設定します。

Xon/Xoff ペーシングの場合	**DTR ペーシングの場合**
ESC.R:	ESC.R:
ESC.M2:	ESC.M2:
ESC.N2:	ESC.N2:
ESC.P1:	ESC.P3:

各行はスペースを入れずに入力する必要があります。ESC の ASCII 値は 27 です。. (ピリオド) はコマンドの一部です。

既存キューへのローカル・プリンターの追加

既存キューにローカル・プリンターを追加できます。

この作業を行うには、次のいずれかのユーザーであることが必要です。

- root 権限のあるユーザー
- **printq** グループのメンバー

既存キューにローカル・プリンターを追加するには、次の手順を使用します。

システム・プロンプトで「**smit mkpqprt**」と入力し、指示に従ってこの作業を完了します。

既存キューへの ASCII 端末プリンターの追加

既存キューに ASCII 端末プリンターを追加できます。

この作業を行うには、次のいずれかのユーザーであることが必要です。

- root 権限のあるユーザー
- **printq** グループのメンバー

既存キューに ASCII 端末プリンターを追加するには、次の手順を使用します。

1. システム・プロンプトで、`smit mkpqprt` と入力します。
2. 「**ascii**」接続タイプ、製造メーカー、プリンター・モデル、および TTY 名を選択します。
3. プロンプトに応じて追加情報を入力します。

既存キューへの HP JetDirect プリンターの追加

既存キューに HP JetDirect プリンターを追加できます。

この作業を行うには、次のいずれかのユーザーであることが必要です。

- root 権限のあるユーザー
- **printq** グループのメンバー

既存キューに HP JetDirect プリンターを追加するには、次の手順を使用します。

1. システム・プロンプトで「**smit mkpqprt**」と入力し、指示に従ってプリンターを追加します。
2. プロンプトに応じて追加情報を入力します。

既存キューへのファイルの追加

既存キューにファイルを追加できます。

この作業を行うには、次のいずれかのユーザーであることが必要です。

- root 権限のあるユーザー
- **printq** グループのメンバー

既存キューにファイルを追加するには、次の手順を使用します。

1. システム・プロンプトで、`smit mkpqprt` と入力します。
2. 「**file (ファイル)**」接続タイプ、製造メーカー、およびモデルを選択します。
3. 「**Name of existing FILE** in the `/dev` directory (`/dev` ディレクトリーに存在するファイル名)」にファイル名を入力します。これは、印刷ジョブ出力を保管するファイルです。このファイルは既に存在していて、`/dev` ディレクトリーに置かれている必要があります。
4. プロンプトに応じて追加情報を入力します。

サポートされないプリンターの構成

サポートされないプリンターは、対応するプリンター・ドライバーがオペレーティング・システムに付属して提供されないプリンターです。ユーザーが、サポートされないプリンターを構成できます。

サポートされないプリンターを構成する方法はいくつかあります。サポートされないプリンターと印刷スプーリング・サブシステムの機能をサポートするために、仮想プリンターを定義する必要があります。また、基本オペレーティング・システムのシリアル・プリンター・デバイス・ドライバーと組み合わせて正しく機能するように、サポートされないプリンターの調整が必要になることもあります。

サポートされないプリンターの構成オプション

ご使用のプリンターを構成し、そのプリンターに対応するように仮想プリンターの特性を変更できます。

サポートされないプリンターを構成して使用するには、次のいずれかの方式を選択できます。

- サポートされないプリンターが、サポートされるプリンターと同じハードウェア・インターフェース (シリアルまたはパラレル) を使用し、そのプリンターにきわめて近い機能を備えている場合は、そのデバイスをサポートされるプリンターとして構成できます。
- ご使用のプリンターに類似したサポートされるプリンターが存在しない場合は、ご使用のプリンターをサポートされるプリンターとして構成できます。ご使用のプリンターに対応するように、仮想プリンターの特性を変更します。

- ご使用のプリンターがサポートされるデバイスをエミュレートしているかどうか分からない場合は、プリンター・タイプとして「**generic (汎用)**」を使用し、適切なインターフェース・タイプを使用します。オペレーティング・システムは、「other parallel printer (他のパラレル・ポート接続プリンター) (**opp**)」、および「other serial printer (他のシリアル接続プリンター) (**osp**)」の 2 つの汎用デバイスを提供しています。インターフェース・タイプを選択してこれらのデバイスのいずれかを指定し (例: **parallel rs232**)、ご使用のプリンターのマニュアルに記載されている仕様に従って特性を変更してください。
- ユーザー間でプリンターを共用するために印刷スプーリング・サブシステムを使用可能にしたいが、データ・ストリームのフォーマット設定に仮想プリンター・システムを使用する必要がない場合は、プリンター・デバイス・ドライバーと印刷キューを構成し、一方で印刷サブシステムはすべての印刷要求を透過的にプリンターに渡すように設定します。この構成では、アプリケーションがプリンター・データ・ストリームを正しく組み立てる必要があります。
- ラスター・グラフィックスとしての入力を必要とする静電プロッターなど、出力デバイスに特殊なフォーマット設定要件がある場合は、プリンター・フォーマッターまたはプリンター・バックエンド・プログラムの代わりに、フォーマット設定ソフトウェアを使用してください。

サポートされない仮想プリンターのカスタマイズ

サポートされないプリンターと印刷スプーリング・サブシステムの機能をサポートするために、仮想プリンターを定義する必要があります。

仮想プリンターを定義するには、次の手順で行います。

1. サポートされないプリンターをカスタマイズするために、ご使用のプリンターに最もよく一致するプリンター・データ・ストリームを識別します。オペレーティング・システムに定義済みの仮想プリンターによって、次のデータ・ストリームがサポートされます。

項目	説明
asc	拡張 ASCII
pcl	Hewlett-Packard LaserJet
gl	プロッター
ps	PostScript
630	Diablo 630
855	Texas Instruments 855 ドット・マトリックス・プリンター (dp モード)

2. プリンターが使用するデータ・ストリームを識別した後、同じデータ・ストリームを使用するサポートされるプリンター、または汎用プリンターの 1 つを選択して、ご使用のプリンターに合うように定義をカスタマイズします。

サポートされないプリンターの配線

基本オペレーティング・システムのシリアル・プリンター・デバイス・ドライバーと組み合わせて正しく機能するように、サポートされないプリンターの調整が必要になることがあります。

次の表で、シリアル・プリンター・デバイス・ドライバーに対する RS-232 シグナルの意味を詳しく説明します。

RS-232 シグナル	シリアル・プリンター・デバイス・ドライバーによる使用法
FG	フレーム接地。シールドとして使われることがよくあります。
TxD —>	プリンターへのデータ送信に使用されます。
RxD <—	プリンターからのデータ受信に使用されます。
RTS <—	プリンター・ポートが開かれた後は高電圧に保たれます。ホスト状況をプリンターに提供します。データ・ペーシングには使用されません。
CTS <—	プリンター・ポートが開かれた後は高電圧になる必要があります。プリンターがオンになっていることの検出に使用されます。
DSR	使用されません。通常は DCD に結合されます。
SG	シグナルの基準電圧。
DCD <—	DTR が「**yes (はい)**」に設定されている場合に、データ・ペーシングに使用されます。
DTR —>	プリンター・ポートが開かれた後は高電圧に保たれます。ホスト状況をプリンターに提供します。

1. ケーブル・シールドとして FG を使用する場合は、片方の端のみが接続されていることを確認してください。どちらの端が接続されていても違いはありません。これは、電子ノイズからの効果的な保護になります。
2. RTS シグナルを使用してプリンター・ポート上の CTS に電圧を供給する場合は、ご使用のプリンターが RTS シグナルによって行う動作を調べてください。

   RTS および CTS のデータ・ペーシングはシリアル・プリンターではサポートされないので、デバイス・ドライバーは CTS が高電圧になるまでプリンター・ポートが開かないようにブロックします。CTS シグナルは、通常はプリンターからの RTS シグナルによって供給されます。ただし、一部のプリンターはデータ・ペーシングに RTS シグナルを使用します。これらのプリンターは、システムにデータの送信を停止させるときに RTS をドロップします。キューイング・システムが状況を検査するためにはポートが常に開いている必要があるので、プリンターが RTS シグナルをドロップすると、ポートは閉じてキューが停止します。
3. プリンターによっては、DCD と DSR または CTS の電圧をプリンター側で上げる必要があります。これらの電圧を上げる必要があるプリンターを使用している場合は、次のいずれかの方法を使用して電圧を上げてください。
   a. コンピューター側の DTR または RTS を使用して電圧を供給する。

      または

   b. プリンター側から電圧を得る。

端末接続プリンターによる印刷

多くの非同期 ASCII 端末には補助 (AUX) ポートが備わっており、このポートを使用してプリンターを接続できます。端末接続による印刷は、ホスト・マシンに直接接続された端末装置、またはホスト・マシンにモデム経由でリモート側から接続された端末装置でサポートされています。

ここでは、端末接続プリンターの構成、保守、および問題判別について、さらに次に示すトピックについて説明します。

端末接続プリンターの取り付け

端末接続プリンターを使用するには、その前にプリンターを取り付け、印刷スプーリング・サブシステムに対してプリンターを構成する必要があります。

端末接続プリンターを新規に取り付け、印刷スプーリング・サブシステムに対して構成するには、次の手順で行います。

1. 物理 ASCII 端末 (tty) デバイスを取り付け、システムに接続します。 24 ページの『物理 ASCII 端末の取り付け』を参照してください。

2. ASCII 端末用の TTY デバイス・ドライバーを構成します。『端末デバイス・ドライバーの構成 (TTY)』を参照してください。
3. 端末装置の出力を検査します。『端末装置の出力の検査』を参照してください。
4. ASCII 端末の AUX ポートまたは PRINT ポートにシリアル・プリンターを接続します。『物理プリンターの取り付け』を参照してください。
5. 仮想プリンターと印刷キューを構成します。 25 ページの『仮想プリンターと印刷キューの構成』を参照してください。
6. モデム回線のキューを設定します。 25 ページの『モデム接続へのキューの構成』を参照してください。

物理 ASCII 端末の取り付け:

物理 ASCII 端末を使用するには、その前に端末装置を取り付ける必要があります。

物理 ASCII 端末を取り付けるには、次の手順で行います。

1. 取り付けの計画に関するすべての関連情報と端末装置の資料を検討して、取り付けに必要なコンポーネントがすべて揃っていることを確認します。
2. ご使用のシステムの構成を検討し、シリアル・ポートを選択します。
3. 通信ポートが使用中でないことを確認します。
4. 端末装置をシリアル通信ポートに接続します。必ず適切な配線を使用してください。配線手順に関する資料を参照してください。
5. 端末装置に付属の資料に従って、端末装置を構成します。ボー・レート、ストップ・ビット、1 文字あたりのビット数、およびフロー制御について、選択した設定値を必ず記録しておいてください。基本オペレーティング・システムの TTY デバイス・ドライバーを構成するために、この情報が必要になります。

端末デバイス・ドライバーの構成 (TTY):

関連した端末装置を使用する前に、端末デバイス・ドライバー (TTY) を構成する必要があります。

TTY を構成するには、次の手順で行います。

1. root ユーザーとしてログインします。
2. システム・プロンプトで「**smit tty**」コマンドを入力し、指示に従ってこの作業を完了します。

端末装置の出力の検査:

端末デバイスを構成した後、端末が作動しているかどうか検査する必要があります。

端末装置が作動しているかどうか検査し、出力を端末装置の画面に直接送るには、次のコマンドを使用します。

```
cat /etc/qconfig > /dev/tty nn
```

ただし、*nn* は該当する TTY デバイス番号です。 `/etc/qconfig` ファイルの内容が端末装置の画面に表示されます。

物理プリンターの取り付け:

物理プリンターを使用するには、その前にプリンターを取り付ける必要があります。

物理プリンターを取り付けるには、次の手順で行います。

1. 取り付けの計画に関するすべての関連情報とプリンターの資料を検討して、プリンターの取り付けに必要なコンポーネントと情報が揃っていることを確認します。
2. 補助 (AUX) ポートへのプリンターの接続について、端末装置の資料を確認します。
3. 端末装置の AUX ポートが、ご使用のプリンターと同じ設定値を使用して構成されていることを確認します (ボー・レート、パリティー、データ・ビット、ストップ・ビット、XON/XOFF など)。
   - AUX ポートの値の設定については、ご使用の端末装置の資料を参照してください。
   - プリンターのシリアル・インターフェースの構成については、ご使用のプリンターの資料を参照してください。
4. 端末装置の AUX ポートにプリンターを接続します。必ず適切な配線を使用してください。配線手順に関する資料を参照してください。

仮想プリンターと印刷キューの構成:

仮想プリンターと印刷キューを使用するには、その前に仮想プリンターと印刷キューを構成する必要があります。

印刷スプーリング・サブシステムに対して端末接続プリンターを構成するには、次の手順で行います。

システム・プロンプトで「**smit mkpq**」と入力し、指示に従ってこの作業を完了します。

「**piomkpq**」コマンドを使用して、この手順を実行することもできます。

モデム接続へのキューの構成:

モデム接続へのキューを構成できます。

端末接続による印刷をサポートするために、特定の端末装置へのキューを作成する代わりに、モデム回線へのキューを設定することもできます。ダイヤルイン端末装置のターミナル・タイプを保証することはできないので、次のコマンドを入力して、**PIOTERM** 環境変数をダイヤルイン端末装置のターミナル・タイプに設定してください。

```
export PIOTERM=Dialin-Terminal-Type
```

ASCII ディスプレイ端末に対するプリンターの構成

ASCII ディスプレイ端末に対してプリンターを構成できます。

ASCII ディスプレイ端末に対してプリンターを構成するには、次の前提条件を満たしている必要があります。

- ASCII 端末の AUX ポートまたは PRINT ポートにシリアル・プリンターを接続する必要があります。配線手順については、端末装置の資料を参照してください。
- ASCII 端末に対して TTY デバイスを定義する必要があります。
- プリンターをオンラインにする必要があります。
- 端末装置の AUX ポートが、ご使用のプリンターと同じ設定値を使用して構成されていることを確認します。このためには、AUX ポートの値の設定について、ご使用の端末装置の資料を参照してください。プリンターのシリアル・インターフェースの構成については、ご使用のプリンターの資料を参照してください。
- root ユーザー権限が必要です。

ASCII ディスプレイ端末に対してプリンターを構成するには、次の手順で行います。

1. システム・プロンプトで、次のように入力します。

```
smit mkpq
```

2. 「**ascii**」接続タイプ、製造メーカー、およびプリンター・モデルを選択します。
3. プロンプトに応じて追加情報を入力します。

piomkpq コマンドを使用してこの作業を実行することもできます。

端末接続による印刷の制限

端末接続プリンターを構成する際には、次の制限を考慮に入れてください。

1. ASCII データのみをプリンターに送信します。バイナリー・データによって端末装置が予期せずロックされたり、印刷が途中で停止したりする可能性があります。
2. 用紙切れやプリンター・オフラインなどの条件を示すプリンター状況メッセージはサポートされません。

端末接続プリンター用にサポートされるハードウェア

端末接続プリンター用にサポートされるハードウェアのリストを示します。

端末接続による印刷を行うためにサポートされるハードウェアは、次のとおりです。

- ケーブル
  - RS-232
  - RS-422
- 端末デバイス
  - IBM 3151、3161、3162、3163、3164
  - DEC VT100、VT220、VT320、VT330
  - WYSE 30、50、60、350
- プリンター
  - IBM 2380 パーソナル・プリンター II
  - IBM 2381 パーソナル・プリンター II
  - IBM 2390 パーソナル・プリンター II
  - IBM 2391 パーソナル・プリンター II
  - IBM 2380 パーソナル・プリンター II (モデル 2)
  - IBM 2381 パーソナル・プリンター II (モデル 2)
  - IBM 2390 パーソナル・プリンター II (モデル 2)
  - IBM 2391 パーソナル・プリンター II (モデル 2)
  - IBM 3112 ページ印刷装置
  - IBM 3116 ページ印刷装置
  - IBM 3130 レーザー・プリンター
  - IBM 4019 レーザー・プリンター
  - IBM 4029 レーザー・プリンター
  - IBM 4037 レーザー・プリンター
  - IBM 4039 レーザー・プリンター
  - IBM 4076 インクジェット・プリンター
  - IBM 4201 モデル 3 プロプリンター III

- IBM 4202 モデル 3 プロプリンター III XL
- IBM 4207 モデル 2 プロプリンター X24E
- IBM 4208 モデル 2 プロプリンター XL24E
- IBM 4247 印刷装置
- IBM 5204 印刷装置
- IBM 6400 印刷装置
- IBM InfoPrint 40 プリンター
- IBM ネットワーク・カラー・プリンター
- IBM ネットワーク・プリンター 12
- IBM ネットワーク・プリンター 17
- IBM ネットワーク・プリンター 24
- Hewlett-Packard 2500C カラー・プリンター
- Hewlett-Packard LaserJet II
- Hewlett-Packard LaserJet III
- Hewlett Packard LaserJet IIISi
- Hewlett-Packard LaserJet 4
- Hewlett Packard LaserJet 4Si
- Hewlett Packard LaserJet 4 Plus
- Hewlett Packard LaserJet 4V
- Hewlett-Packard LaserJet 5000 D640 プリンター
- Hewlett Packard LaserJet 5Si/5Si MX
- Hewlett Packard LaserJet 5Si Mopier
- Hewlett-Packard LaserJet 8000 プリンター
- Hewlett-Packard LaserJet 8100 プリンター
- Hewlett Packard LaserJet Color
- Hewlett-Packard Color LaserJet 4500
- Hewlett-Packard Color LaserJet 8500
- Lexmark Optra レーザー・プリンター
- Lexmark Optra E310 レーザー・プリンター
- Lexmark Optra M410 レーザー・プリンター
- Lexmark Optra Se レーザー・プリンター
- Lexmark Optra T レーザー・プリンター・ファミリー
- Lexmark Optra W810 レーザー・プリンター
- Lexmark Optra Plus レーザー・プリンター
- Lexmark Optra C カラー・レーザー・プリンター
- Lexmark Optra E レーザー・プリンター
- Lexmark Optra N レーザー・プリンター
- Lexmark ExecJet IIc
- Lexmark ValueWriter 600
- Lexmark 2380 Plus プリンター (モデル 3)

- Lexmark 2381 Plus プリンター (モデル 3)
- Lexmark 2390 Plus プリンター (モデル 3)
- Lexmark 2391 Plus プリンター (モデル 3)
- Lexmark 4039 Plus レーザー・プリンター
- Lexmark 4079 Color JetPrinter Plus
- Lexmark 4227 フォーム・プリンター

- 非同期通信アダプター
  - ネイティブ・シリアル・ポート・コントローラー
  - 8 ポート・コントローラー
  - 16 ポート・コントローラー
  - 64 ポート・コントローラー
  - 128 ポート・コントローラー
  - サード・パーティー・コントローラー

    注: サード・パーティーの非同期コントローラーもサポートされます。ASCII 端末がサード・パーティーのコントローラーと組み合わせて構成されていることを基本オペレーティング・システムが検出した場合、端末接続プリンターは、ネイティブ・ポート・コントローラーに接続されている場合と同様に構成されます。詳しくは、 29 ページの『ネイティブ、8 ポート、16 ポート、およびサード・パーティーのコントローラー』を参照してください。

前記のリストにある一部の Lexmark プリンターについて詳しくは、次に示すセクションを参照してください。

- 164 ページの『IBM InfoPrint 40 プリンター』
- 192 ページの『Lexmark Optra E310 レーザー・プリンター』
- 194 ページの『Lexmark Optra M410 レーザー・プリンター』
- 196 ページの『Lexmark Optra Se レーザー・プリンター』
- 199 ページの『Lexmark Optra T レーザー・プリンター・ファミリー』
- 202 ページの『Lexmark Optra W810 レーザー・プリンター』

terminfo データベース

印刷サービスは、標準インターフェース・スクリプトと **terminfo** データベースを使用して、それぞれのプリンターの初期設定、および選択されたページ・サイズ、文字ピッチ、行ピッチ、および文字セットのセットアップを行います。

このため、印刷サービスに新規プリンターを追加するには、通常は **terminfo** データベース (`/usr/lib/terminfo/terminfo.lp`) に正しいエントリーが存在していれば十分です。

terminfo データベースは、それぞれのプリンターをショート・ネームによって識別します。これは、**TERM** シェル変数の設定に使用される名前と同じ種類のものです。例えば、AT&T モデル 455 プリンターを示す **terminfo** データベース内での名前は **455** です。

ご使用のプリンターの **terminfo** タイプを指定するには、**lpadmin** コマンドの **-T** オプションを使用します。デフォルトで、**terminfo** データベースには多くの一般的なプリンターに対応するエントリーが含まれています。ご使用のプリンターに対応する **terminfo** タイプを選択してください。

terminfo にご使用のプリンターに対応するエントリーが含まれていない場合でも、印刷サービスに対してそのプリンターを使用できることがあります。ただし、ページ・サイズ、ピッチ、および文字セットの自動選択は使用できず、また印刷要求のたびにプリンターが正しいモードに設定されるように維持したり、プリンターに対してプリンター・フォームを使用したりする際に問題が生じることがあります。この場合は、ご使用のプリンターに対応するエントリーを *terminfo* に追加するか (220 ページの『terminfo データベースへのプリンター・エントリーの追加』)、カスタマイズしたインターフェース・プログラムを作成して (217 ページの『プリンター・インターフェース・スクリプトの作成』)、プリンターに対して使用できます。

それぞれの端末装置またはプリンターごとに、多数の項目を **terminfo** データベースに定義できます。ただし、印刷サービスが使用するものはこれらのうち 50 に満たない数で、ほとんどのプリンターはさらに少数しか必要としません。次のコマンドを入力すれば、特定の **terminfo** エントリーに対して定義された項目を確認できます。

```
infocmp terminfo_name
```

サポートされない端末装置のサポートの追加

サポートされない端末装置用の制御シーケンスを **terminfo** データベースに追加できます。

制御シーケンスは、`/usr/share/lib/terminfo` ディレクトリー内の **terminfo** データベースに追加する必要があります。ご使用の端末装置用の制御シーケンス値を追加するには、次の手順で行います。

1. 該当する `*.ti` ファイルを編集します。
2. **tic** コマンドを使用してファイルをコンパイルします。制御シーケンス値について詳しくは、ご使用の端末装置に付属の資料を参照してください。

   仮想プリンター・データベース は、プリンターに送信するデータ・ストリームなど、印刷要求を処理する方法を記述する一連のファイルです。端末接続プリンターに固有のユーザー構成可能な属性は、仮想プリンター・データベース内で定義され、使用される非同期通信アダプターによって異なります。

   仮想プリンターを構成する際には、*仮想プリンター属性* を定義します。端末接続プリンターに固有の属性の命名規則は、**y***N* です。ここで、*N* は 0 以上の整数です。値 **y0** は予約済みです。この属性は、端末接続プリンターに対して仮想プリンター・キューが構成されていることを指定し、端末ポートに対するハードウェア伝送制御手順を含んでいます。以降では、端末接続プリンターに対するアダプター固有の仮想プリンター属性について詳しく説明します。

   既存の仮想プリンターの属性値を変更するには、「**smit ps_lsvirprt**」高速パス・コマンドを使用します。

ネイティブ、8 ポート、16 ポート、およびサード・パーティーのコントローラー

ネイティブ・ポート (S1 または S2)、8 ポート、および 16 ポートのコントローラーは、端末接続プリンターに対してハードウェア・サポートを提供せず、サード・パーティー・コントローラーのハードウェア・サポートは不明です。このため、印刷ファイルは小さなデータ・ブロックに分割する必要があります。

それぞれのデータ・ブロックの前には **mc5** 制御シーケンスがあり、データ・ブロックの後には **mc4** 制御シーケンスが続きます。端末装置が **mc5** 制御シーケンスを受信すると、**mc4** 制御シーケンスを受信するまで、後続のデータはすべて AUX ポートに送付されます。

端末装置に送信されるデータ・ブロックは、比較的小さくする必要があります。TTY に一度に送信する文字数が多すぎると、プリンターへの出力が、送信操作中に入力された内容のエコーと混ざる可能性があります。また、データ受信エラーが最小限になるように、データ・ブロック伝送間隔の遅延時間も設定する必要があります。

ネイティブ・ポート、8 ポート、16 ポート、およびサード・パーティーのコントローラーには、ブロック・サイズと遅延値を指定するための次に示す仮想プリンター属性があります。これらの属性は、ファイル内で指定されます。

項目	説明
y1	データ・ブロックの最大文字数を示します。
y2	データ・ブロックを伝送する間隔を遅らせる時間をマイクロ秒数で示します。

64 ポート・コントローラー

64 ポート・コントローラーは、端末接続プリンターのハードウェア・サポートを提供します。

64 ポート・コントローラーには、次の仮想プリンター属性があります。

項目	説明
y1	印刷処理が端末装置アクティビティーよりも優先される度合いを設定します。数が大きいほど、プリンターが端末装置よりも優先される度合いが高くなります。

128 ポート・コントローラー

128 ポート・コントローラーも、端末接続プリンターのハードウェア・サポートを提供します。

128 ポート・コントローラーには、次の仮想プリンター属性があります。

項目	説明
y1	文字を印刷装置に送信する最大の字/秒 (CPS) 速度を設定します。この速度は、ご使用のプリンターの平均印刷速度よりわずかに低くする必要があります。印刷速度については、ご使用のプリンターの資料を参照してください。
y2	デバイス・ドライバーが出力キューに入れる印刷文字の最大数を設定します。この数を減らすと、システムのオーバーヘッドが増えます。この数を増やすと、端末接続プリンターが使用中のときに、オペレーターのキー・ストロークのエコー時間に遅延が生じます。
y3	端末接続プリンターの入力バッファー・サイズのデバイス・ドライバーによる見積もりを設定します。一定期間アクティビティーがないと、ドライバーは指定された数の文字をプリンターにバースト送信します。入力バッファー・サイズについては、ご使用のプリンターの資料を参照してください。

プリンター・バックエンド・コマンド

piobe コマンドは、ローカル接続プリンター・デバイスへの印刷時に、印刷スプーリング・サブシステムによって実行される標準のバックエンド・プログラムです。

piobe コマンドは、**qdaemon** プロセスによって開始されます。このコマンドは、フラグを読み取るか仮想プリンター・データベースを照会して、作成するデータ・ストリームを決定します。その後、**piobe** プロセスは、正しいデータ・ストリームが生成されるように、適切なフィルターのパイプラインを経由して印刷ファイルを渡します。このパイプラインの最後に、フィルター済みファイルは **pioout** デバイス・ドライバー・インターフェース・プログラムに渡されます。

pioout コマンドは、パイプライン内で **piobe** コマンドによって呼び出されます。ローカル接続プリンターの場合、**pioout** コマンドは印刷ファイルを適切なプリンター・デバイス・ドライバーに送信します (例: `/dev/lp1`)。一方、端末接続プリンターの場合は、印刷ファイルは terminfo データベースおよび仮想プリン

ター・データベースから収集したデータによって変更された後、TTY デバイス・ドライバー (例: /dev/tty0) を経由してプリンターに送信されます。**mc5** および **mc4** 端末管理属性については、**terminfo** データベースが照会されます。非同期コントローラーに固有の属性については、仮想プリンター・データベースが照会されます。

印刷キューのリスト表示

「**lsallq**」コマンド、または SMIT を使用して、印刷キューのリストを表示できます。

- ローカル印刷キューの場合は、プリンター・デバイスがシステムに接続されている必要があります。
- リモート印刷キューの場合は、リモート・ホストと通信するようにシステムを構成する必要があります。

次の手順は、ローカルとリモートの両方の印刷キューに適用されます。

システム・プロンプトで「**smit lspq**」と入力し、指示に従ってこの作業を完了します。

「**lsallq**」コマンドを使用して、この作業を実行することもできます。

印刷キューの状況の表示

enq コマンド、または SMIT を使用して、印刷キューの状況を表示できます。

SMIT インターフェースを使用して、印刷キューの状況を表示できます。

システム・プロンプトで、「**smit qstatus**」と入力します。

この作業は、「`enq -e "$@"`」コマンドを使用して実行することもできます。

印刷キューの開始および停止

root 権限がある場合は、印刷キューを開始および停止できます。

これらの作業を行うには、root 権限が必要です。

キューを開始するには、次の手順で行います。

```
smit qstart
```

または

```
qadm -U QueueName
```

キューを停止するには、次の手順で行います。

```
smit qstop
```

または

```
qadm -D QueueName
```

デフォルト印刷キューの設定

SMIT を使用して、デフォルト印刷キューを設定できます。

この作業を行うには、次のいずれかのユーザーであることが必要です。

- root 権限のあるユーザー
- printq グループのメンバー

デフォルト印刷キューを設定するには、次の手順で行います。

システム・プロンプトで「**smit qdefault**」と入力し、指示に従ってこの作業を完了します。

印刷ジョブのスケジューリング

SMIT 高速パスを使用して、印刷ジョブをスケジュールに入れることができます。

印刷ジョブをスケジュールに入れるには、root ユーザーのログイン名を /var/adm/cron/at.allow ファイルに組み入れるか、自身が root ユーザー権限を持っている必要があります。

- スケジュールに入っている印刷ジョブをすべてリストするには、システム・プロンプトで次のコマンドを入力します。

  ```
  smit lsat
  ```

  このコマンドは、ユーザーがスケジュールに入れたすべての印刷ジョブのリストを表示します。root ユーザー権限がある場合は、現在スケジュールに入っているすべての印刷ジョブがこのコマンドによってリストされます。
- 印刷ジョブをスケジュールに入れるには、次の手順で行います。
  1. システム・プロンプトで次のコマンドを入力します。

     ```
     smit sjat
     ```
  2. 適切な日付/時刻フィールドを選択するか、フィールドに入力します。
  3. プロンプトに応じて追加情報を入力します。
- スケジュールに入れたジョブを削除するには、システム・プロンプトで次のコマンドを入力します。
  1. 次のように入力します。

     ```
     smit rmat
     ```
  2. 「**List (リスト)**」を選択して、ジョブ番号を削除します。

キュー特性の変更または表示

ローカルまたはリモートの印刷キュー、および印刷キュー・デバイスのキュー特性を表示したり、変更したりすることができます。

キューの属性を変更または表示するには、次の前提条件が満たされている必要があります。

- ローカル印刷キューの場合は、プリンターがシステムに物理的に接続されている必要があります。
- リモート印刷キューの場合は、リモート印刷サーバーと通信するようにシステムを構成する必要があります。
- キューまたはキュー・デバイスの特性を変更するには、root 権限が必要です。

次の手順は、ローカルとリモートの両方の印刷キューと印刷キュー・デバイスに適用されます。

システム・プロンプトで「**smit chpq**」と入力し、指示に従ってこの作業を完了します。

この手順は、**chque**、**chquedev**、**lsvirprt**、および **chvirprt** の各コマンドを使用して行うこともできます。

印刷キューの削除

ローカルおよびリモートの印刷キューを削除できます。

- ローカル印刷キューの場合は、プリンターがシステムに物理的に接続されている必要があります。

- リモート印刷キューの場合は、リモート印刷サーバーと通信するようにシステムを構成する必要があります。
- キューまたはキュー・デバイスを削除するには、root 権限が必要です。

ローカルまたはリモートの印刷キューを削除するには、次の手順で行います。

システム・プロンプトで「**smit rmpq**」と入力し、指示に従ってこの作業を完了します。

注: 選択したキューにあるプリンターがただ 1 つの場合は、キューとそのプリンターが除去されます。キューに複数のプリンターがある場合は、選択したプリンターのみが除去されます。

rmque、**rmquedev**、および **rmvirprt** の各コマンドを使用して、この作業を行うこともできます。

他のプリンター管理作業の実行

プリンターの属性を管理および変更できます。

ここでは、次のトピックについて説明します。

サポートされるプリンターすべてのリスト表示

SMIT を使用して、サポートされるプリンターをすべてリストできます。

システム・プロンプトで、「**smit lssprt**」高速パス・コマンドを入力します。

次の例のような出力が表示されます。

```
bull1021    parallel Bull Compuprint Page Master 1021
.
.
.
ibm2380     parallel IBM 2380 Personal Printer II
ibm2380     rs232    IBM 2380 Personal Printer II
ibm2380     rs422    IBM 2380 Personal Printer II
.
.
.
opp      parallel Other parallel printer
osp      rs232    Other serial printer
osp      rs422    Other serial printer
```

定義済みのプリンターすべてのリスト表示

SMIT を使用して、定義済みのプリンターをすべてリストできます。

システム・プロンプトで、「**smit lsdprt**」高速パス・コマンドを入力します。

次の例のような出力が表示されます。

```
lp0  Available  00-04-01-06  Other serial printer
lp1  Available  00-04-01-07  Other serial printer
lp2  Available  00-00-0P-00  Other parallel printer
```

プリンターの削除

システムからプリンターを削除できます。

- プリンターが既に追加済みであることが必要です。 18 ページの『キューを追加しない場合のプリンターの構成 (SMIT)』、および 18 ページの『キューを追加しない場合のプリンターの構成 (qprt コマンド)』を参照してください。

- root 権限が必要です。

プリンターを削除しても、印刷ジョブをそのプリンターに送る印刷キューは削除されません。印刷キューも削除する必要がある場合は、32 ページの『印刷キューの削除』を参照してください。

SMIT を使用して、プリンターを削除することができます。

システム・プロンプトで「**smit rmprt**」と入力し、指示に従ってこの作業を完了します。

プリンター・サーバー・サブシステムの状況の表示

SMIT を使用して、印刷サーバー・サブシステムの状況を表示できます。

1. システム・プロンプトで、次のように入力します。

   ```
   smit server
   ```

2. 「**Show Status of the Print Server Subsystem (印刷サーバー・サブシステムの状況の表示)**」を選択します。

プリンター・キューイング・システムの状況条件:

プリンターまたはデバイスが TTY デバイスとして追加されている場合、キューイング・システムはキャリア検出 (CD) を探してプリンターを認識します。デバイスが LP デバイスの場合、キューイング・システムは CTS を使用してプリンターを検出します。

次のリストに、印刷キューの状況条件を示します。

項目	説明
DEV_BUSY	次の条件を示します。 • プリンター・デバイス (**lp0**) に対して複数のキューが定義されており、別のキューがプリンター・デバイスを現在使用しています。 • **qdaemon** はプリンター・ポート・デバイス (**lp0**) の使用を試みましたが、別のアプリケーションがそのプリンター・デバイスを現在使用しています。詳しくは、**qdaemon** コマンドを参照してください。 **通常のリカバリー: DEV_BUSY** からリカバリーするには、キューまたはアプリケーションがプリンター・デバイスを解放するまで待つか、プリンター・ポートを使用しているジョブまたはプロセスを強制終了してください。
DEV_WAIT	プリンターがオフラインになっているか、用紙切れまたはジャム、ケーブルのゆるみ、不良、または配線の誤りが起きているために、キューがプリンターを待機していることを示します。 **通常のリカバリー: DEV_WAIT** からリカバリーするには、待機の原因になっている問題を訂正する必要があります。プリンターがオフラインになっていないか、用紙切れまたはジャムを起こしていないか、ケーブルがゆるんでいないか確認してください。診断テストを行うには、**enq** コマンドを使用して、キューに入れられた印刷中または **DOWN** 状態のジョブすべてを、**DEV_WAIT** キューから別のキューに移動すると便利です。問題を訂正した後、まだ印刷されていないジョブを元のキューに戻すことができます。 **DEV_WAIT** は、プリンターに対する不適切なフロー制御が原因で発生することもあります (特に、XON/XOFF ソフトウェア制御を使用している場合)。SMIT を使用して、適切なフロー制御を使用しているかどうか確認してください (XON/XOFF または DTR ペーシング)。 ケーブル接続の不良または配線の誤りによって **DEV_WAIT** 状態が発生することもあります。通常、ケーブルを取り替えない限りこの状態からはリカバリーできません。

項目	説明
DOWN	デバイス・ドライバーが **TIMEOUT** 秒を過ぎてもプリンターと通信できないことを示します (CD または CTS がドロップされた、または低電圧)。**TIMEOUT** 値は、キューイング・システムがプリンター操作の完了を待つ時間の長さ (秒) を示します。この値は、SMIT を使用して設定できます。 キューは通常、**DEV_WAIT** 状態になった後で **DOWN** 状態に移行します。キューが直接 **DOWN** 状態になる場合は、**TIMEOUT** 値が小さすぎるか、ケーブル接続に問題があります。通常、この状態は、正しいシグナリングが行われていないために、プリンターのデバイス・ドライバーがプリンターの存在を判別できないときに発生します。ただし、プリンターによっては、単にオフラインであるというシグナルをキューイング・システムに送ることができないものがあります。これらのプリンターは、オフであるというシグナルを送ります。つまり、CTS をドロップするか (lp の場合)、CD をドロップします (TTY の場合)。 プリンター・デバイスがオフの場合、キューは **DOWN** 状態になります。システム管理者は、キューイング・コマンド (**qadm**、**disable**、**enq**、およびその他) を使用して、保守のためにキューを **DOWN** 状態にすることができます。 **通常のリカバリー:** キューをダウン状態にした問題を訂正し、適切なフラグを指定した **qadm**、**enable**、または **enq** の各コマンドを使用して、キューを再起動します。キューを再び使用できるようにするには、キューを手動で起動する必要があります。
HELD	ジョブが保留中であり、**qhld** コマンドまたは **enq** コマンドを使用して解放されるまで、キューに入れられないことを示します。
OPR_WAIT	バックエンド・プログラムが、用紙の給紙などの作業をオペレーターが行うまで待っていることを示します。これは通常はソフトウェアに関連しています。 **通常のリカバリー: OPR_WAIT** 状態からリカバリーするには、キューイング・システムから出された要求に適切に対処してください。
QUEUED	印刷ファイルがキューに入れられており、印刷を順次待機していることを示します。
READY	キューに関連した準備がすべてできており、ジョブをキューに入れて印刷できることを示しています。
RUNNING	印刷ファイルが印刷中であることを示します。
UNKNOWN	別のキューによって使用されているデバイス・ファイルに対してユーザーがキューを作成し、状態が **DEV_WAIT** であることを示します。キューが保留中 (**DEV_WAIT**) のときは、キューはプリンター・デバイス (`lp0`) から状況を取得できません。 **通常のリカバリー:** この問題を訂正するには、他のキューをダウン状態にするか、プリンターの問題を修正します。新規キューをダウン状態にし、再起動すると、キューは **READY** として登録されます。

リモート・キューには、次の状況条件が適用されます。

項目	説明
CONNECT	バックエンドがリモート・ホストへの接続を試みていることを示します。
GET_HOST	バックエンドが、印刷ジョブの送信先であるホストを取得していることを示します。
INITING	バックエンドがネットワークへの接続の確立処理中であることを示します。
SENDING	バックエンドが印刷ジョブをリモート・ホストに送信していることを示します。

リモート印刷

リモート印刷により、さまざまなコンピューターがプリンターを共用できます。

リモート印刷機能を使用するには、コンピューターが TCP/IP 経由で接続されていて、必要な TCP/IP アプリケーション (**lpd** デーモンなど) をサポートしている必要があります。

リモート印刷要求は、ローカル印刷要求と同じ方法でキューに入れられます。

- **qprt**、**lpr**、**enq** などのフロントエンド印刷コマンドは、ローカル・システム上の適切なキューに対して要求を開始します。
- ローカル・システム上の **qdaemon** は、ローカル・キューに入れられたジョブを処理するときと同様に要求を処理します。ただし例外が 1 つあり、**qdaemon** は要求を **piobe** バックエンドではなく **rembak** バックエンド・プログラムに渡します。
- **rembak** プログラムは、TCP/IP ネットワークを経由して印刷ジョブをリモート・サーバーに送信します。
- リモート・サーバー上では、**lpd** デーモンはリモート印刷要求があるかどうかポート 515 をモニターします。
- **lpd** はリモート印刷要求を受け取ると、ジョブを適切なローカル・キューに入れます。
- 印刷要求は、印刷サーバー上の **qdaemon** によって処理されます。
- **qdaemon** は、印刷サーバー上の **piobe** バックエンドに要求を渡します。
- **piobe** バックエンドは、指定されたプリンター上での印刷のためにデータ・ストリームのフォーマットを設定します。

次に、リモート印刷環境を構成、使用、および管理する方法について説明します。

rembak プログラム

リモート印刷要求にサービスを提供するようにセットアップされるローカル・キューは、リモート印刷バックエンド・コマンド **rembak** を使用するように構成する必要があります。

キューをセットアップする際に、バックエンド・プログラムのパス名を入力するようにシステムからプロンプトが出されます。このプロンプトへの入力によって、印刷要求の処理に使用するバックエンド・プログラムを **qdaemon** コマンドに指示します。リモート印刷要求を処理するためにキューをセットアップするには、`/usr/lpd/rembak` と入力します。

また **rembak** コマンドは、状況要求、ジョブ取り消し要求、およびリモート・キューイング・システムの強制終了要求も処理します。 **qchk -A** や **lpstat** などの状況要求は、`qconfig` ファイルとローカル印刷スプーリング・サブシステム状況ファイルを分析して、ローカル印刷キューとデバイスの状況を照会します。

リモート印刷環境では、**qchk -A** コマンドと **lpstat** コマンドは **rembak** プログラムを使用して、キュー状況情報を印刷サーバーに要求します。キュー状況コマンドの出力では、それぞれのリモート・キューごとに 2 つのエントリーが表示されます。最初のエントリーは、リモート・ジョブが送られる先のローカル・キュ

ーの状況です。2 番目のエントリーは、ジョブが印刷されるリモート印刷サーバー上のキューの状況を示します。次の例では、ローカル・システム上のキューとリモート印刷サーバー上のキューの両方に、キュー名 **rq** が使用されています。

```
Queue Dev   Status  Job Files             User       PP  %  Blks Cp
----- ---   ------  --- ---------------   ---------- --- -- ---- --
Iago  Iago  RUNNING 284 mileaf            ann@arctur 15  13  1   1
Pro   asc   READY
bsh   bshde READY
ps    ps    READY
rq    rqd   READY
rq    ps1   RUNNING 297 .deskprint/dsktop sarah@alde 60  22  1   1
            QUEUED  298 .deskprint/howtol sarah@alde     60  1   2
```

前記の例が示すように、現在稼働中またはキューに入れられている印刷ジョブは、キューのリモート印刷サーバー・エントリーに表示されます。

rembak プログラムは、印刷ジョブを取り消す要求もリモート印刷サーバーに送ります。それぞれの印刷ジョブには番号が割り当てられています。前記の例に示したとおり、印刷キュー状況要求を行うと、現在キューに入れられているか印刷要求を実行しているジョブ番号が表示されます。リモート・キュー上のジョブを取り消すには、ローカル印刷ジョブの取り消しに使用されるものと同じコマンドを使用します。例えば、キュー **rq** のジョブ **298** を取り消すには、次のいずれかのコマンドを使用します。

```
qcan -Prq -x298
```

または

```
lprm -Prq 298
```

lpd デーモン

ローカルおよびリモートの印刷ジョブは同じコマンドを使用して実行依頼されますが、異なる方法で処理されます。印刷ジョブがリモート・ホストに送信された後、印刷ジョブはローカル印刷スプーリング・サブシステムによって管理されなくなります。

lpd デーモンは TCP/IP システム・グループの一部です。TCP/IP ネットワーク上にあるすべてのホストが **lpd** デーモンを実行でき、すべてのホストがネットワーク上にある他のホストに印刷要求を送信できます (そのホストが現在 **lpd** を実行していれば)。セキュリティー対策として、**lpd** デーモンは、それぞれのリモート印刷要求を 2 つのデータベース・ファイル (/etc/hosts.equiv ファイルと /etc/hosts.lpd ファイル) と照らし合わせて検査する子プロセスを fork します。印刷要求を実行依頼しているホストの名前が /etc/hosts.lpd ファイルにない場合、印刷要求はリジェクトされます。

注: /etc/hosts.equiv ファイルは、パスワードを指定せずにローカル・ホスト上の特定のコマンドを実行することが許可されている、ネットワーク上のコンピューターを定義します。 /etc/hosts.lpd ファイルは、パスワードを指定せずにローカル・ホスト上の印刷コマンドを実行することが許可されている、ネットワーク上のコンピューターを定義します。

リモート印刷サーバー上の **lpd** デーモンは、印刷要求があるかどうかポート 515 をモニターします。**lpd** デーモンが有効なホストから印刷要求を受信すると、指定されたキューに要求を入れます。 **lpd** デーモンは、印刷要求に指定されたファイルをディレクトリー /var/spool/lpd に置きます。その後、印刷要求は、リモート・サーバー上の **qdaemon** および適切なバックエンド (通常は **piobe**) によって管理されます。

/etc/locks/lpd ファイルには、**lpd** デーモンの現在稼働しているインスタンスのプロセス ID が含まれています。**lpd** デーモンを実行するマシンが作動不能になった場合は、システムを再始動する前に、**lpd** デーモンの ID を除去する必要が生じることがあります。エラー・メッセージ「**lpd: lock file**」または「**duplicate daemon**」は、ID を除去する必要があることを示しています。

lpd デーモンの制御:

lpd デーモンの制御は、**lpd** サブシステムを始動および停止したり、**lpd** サブシステムの特性を変更したりする作業です。

lpd デーモンを開始するには、3 つの方法があります。現在実行されていない場合は、デーモンをいつでも開始できます。また、システム再起動時に **lpd** デーモンが開始されるようにすることも、現在時刻とシステム再起動時の両時点で開始されるようにすることもできます。**lpd** デーモンを停止する場合にも同じオプションを使用できます。つまり、即停止、システム再起動時の停止、または即時およびシステム再起動時の両時点での停止が可能です。**lpd** デーモンを実行する際には、DEBUG を指定するか、SYSLOG を指定するか、DEBUG と SYSLOG の両方を指定するか、またはどちらも指定しないことができます。

SMIT を使用して **lpd** デーモンを制御するには、`smit lpd` と入力し、SMIT メニューから必要なオプションを選択します。 SRC を使用して **lpd** デーモンを制御するには、次の SRC コマンドを使用します。

項目	説明
startsrc	サブシステム、サブシステムのグループ、またはサブサーバーを開始します。
stopsrc	サブシステム、サブシステムのグループ、またはサブサーバーを停止します。
lssrc	サブシステム、サブシステムのグループ、またはサブサーバーの状況を取得します。
refresh	サブシステムまたはサブシステムのグループが、該当する構成ファイルを再読み取りするようにします。
traceson	サブシステム、サブシステムのグループ、またはサブサーバーのトレースを使用可能にします。
tracesoff	サブシステム、サブシステムのグループ、またはサブサーバーのトレースを使用不可にします。

リモート・プリンターおよびキューの管理と使用

リモート・システムを出力先として印刷を行うには、ローカル・システム上でリモート・キューをセットアップする必要があります。

リモート・キューをセットアップするには、ローカル・ホスト上のキューとキュー・デバイスの名前を指定したり、印刷ジョブが送信される先のリモート・ホストの名前、およびリモート・ホスト上のキューを指示したりする作業を行う必要があります。

リモート印刷キューのセットアップ

SMIT を使用して、リモート印刷キューをセットアップできます。

リモート印刷要求を受信するように指定されたリモート・ホスト上のキューは、アクティブ・キューであることが必要です。

- リモート・キューをセットアップするには、「**smit mkrque**」コマンドを使用します。
- 詳しくは、 19 ページの『印刷キュー・デバイスの追加』を参照してください。

リモート印刷と qconfig ファイル

qconfig ファイルには、キュー・デバイスを定義するスタンザが含まれています。リモート・プリンターの場合は、デバイス・スタンザのフィールド値の一部が、ローカル・プリンターのものと異なります。

次の表に、リモート・プリンターの場合に特定の意味を持つフィールドのリストを示します。この表には、これらのフィールドのサンプル値またはデフォルト値も示します。

リモート・キュー・デバイス	サンプル値またはデフォルト値	説明
host	sys2	ジョブを印刷する先のリモート・ホスト (印刷サーバー) の名前。
rq	q2	ジョブを印刷する先のリモート・キューの名前。
s_statfilter	/usr/lpd/aixshort	キュー状況要求 (**qchk** など) に対するリモート・キューの状況情報を簡易書式に変換するために使用されるフィルター。これは、リモート印刷サーバーが別の基本オペレーティング・システムである場合のデフォルト値です。
	/usr/lpd/bsdshort	リモート印刷サーバーが BSD システムである場合に、BSD **lpq** コマンド出力 (簡易書式) を変換するために使用されるフィルター。
	/usr/lpd/attshort	リモート印刷サーバーが ATT システムである場合に、ATT **lpstat** コマンド出力 (簡易書式) を変換するために使用されるフィルター。
l_statfilter	/usr/lpd/aixlong	キュー状況要求 (**qchk** など) に対するリモート・キューの状況情報を詳細書式に変換するために使用されるフィルター。これは、リモート印刷サーバーが別の基本オペレーティング・システムである場合のデフォルト値です。
	/usr/lpd/bsdlong	リモート印刷サーバーが BSD システムである場合に、BSD **lpq** コマンド出力 (詳細書式) を変換するために使用されるフィルター。
	/usr/lpd/attlong	リモート印刷サーバーが ATT システムである場合に、ATT **lpstat** コマンド出力 (詳細書式) を変換するために使用されるフィルター。

印刷サーバーとしてのリモート・ホストの構成

リモート・ホストをプリンター・サーバーとして構成できます。

リモート・ホストを印刷サーバーとして使用する場合は、リモート印刷要求を受け入れるようにホストを構成する必要があります。印刷サーバーが印刷の許可を得るためには、ホストが /etc/hosts.lpd ファイルにリストされている必要があります。

1. システム・プロンプトで「**smit mkhostslpd**」と入力し、指示に従ってこの作業を完了します。
2. ホスト名を /etc/hosts.lpd ファイルに追加するには、ホスト・アクセス・リストを開き、編集します。

印刷サーバーの /etc/hosts.lpd ファイルに定義されていないホストから送られた印刷要求は、リジェクトされます。ホストがライン・プリンターにアクセスできないことを示すエラー・メッセージが表示されます。

印刷サーバーとして動作するホストは、印刷要求にサービスを提供するために、**lpd** プロセスも実行する必要があります。SRC の **lssrc -s lpd** コマンドは、**lpd** デーモンの状況を表示します。デーモンがアクティブでない場合は、「**startsrc**」コマンドを使用して、「**lpd**」デーモンを開始します。

リモート・プリンターおよびキュー

リモート・ホストに対して印刷を行うために特別なコマンドは必要ありません。キューを指定できる任意の印刷コマンドを使用します。

例えば、**lpr**、**qprt**、および **enq** の各コマンドは、印刷コマンドです。キューを指定するフラグなど、適切なフラグとオプションを使用して、印刷要求を調整します。ホスト上のリモート・キューの名前を使用します。

また、**smit qprt** 高速パスを使用して、リモート印刷要求を送信することもできます。

qchk や **lpstat** などのキュー状況コマンドは、ローカルおよびリモートの両方の印刷キューに関する情報を表示します。**smit qchk** コマンドを実行するとメニューが表示され、このメニューを使用して、ローカルおよびリモートの両方のキューから入手したいキュー状況情報のタイプを選択できます。

リモート・キュー内の印刷ジョブを取り消すには、**qcan** コマンド、または **lprm** コマンドを使用します。また、**smit qcan** 高速パスを使用することもできます。

全リモート・ホストのリスト

「**ruser**」コマンド、または SMIT を使用して、リモート・ホストのリストを表示できます。

リモート・ホストをリストするには、以下の条件を満たしている必要があります。

- システムは、リモート印刷サーバーとして通信できるように構成する必要があります。
- システムに **lpd** デーモンをインストールする必要があります。
- リモート・ホストをリストするには、TCP/IP の命名規則を理解していなければなりません。

ここで解説する情報は AIX の特定バージョンを使用してテストされたものです。 したがって、その内容は使用される AIX のバージョンおよびレベルによってかなり異なることがあります。

システム・プロンプトで「**smit lshostslpd**」と入力し、指示に従ってこの作業を完了します。

「**ruser -sP**」コマンドを使用することもできます。

リモート・ホストの追加

ruser コマンドまたは SMIT を使用して、リモート・ホストを追加することができます。

- システムは、リモート印刷サーバーとして通信できるように構成する必要があります。
- システムに **lpd** デーモンをインストールする必要があります。
- リモート・ホストを追加するには、TCP/IP の命名規則を理解していなければなりません。

ここで解説する情報は AIX の特定バージョンを使用してテストされたものです。 したがって、その内容は使用される AIX のバージョンおよびレベルによってかなり異なることがあります。

以下のコマンドで、このタスクを実行することもできます。

```
ruser -a -p HostName
```

または

以下の SMIT 高速パスを使用することもできます。

```
smit mkhostslpd
```

リモート・ホストの削除

リモート・ホストを削除することができます。

- システムは、リモート印刷サーバーとして通信できるように構成する必要があります。
- システムに **lpd** デーモンをインストールする必要があります。
- リモート・ホストを削除するには、TCP/IP の命名規則を理解していなければなりません。

ここで解説する情報は AIX の特定バージョンを使用してテストされたものです。 したがって、その内容は使用される AIX のバージョンおよびレベルによってかなり異なることがあります。

リモート・ホストを削除するには、以下のようにします。

システム・プロンプトで、「**smit rmhostslpd**」と入力するか、または以下の「**ruser**」コマンドを入力します。

また、SMIT 高速パス「**smit rmhostslpd**」または次のコマンドを使用してこの作業を実行することもできます。

```
ruser -d -p HostName
```

lpd リモート・サブシステムの始動

lpd リモート・サブシステムを始動するには、**startsrc** コマンドまたは **mkitab** コマンド、あるいは SMIT を使用することができます。

- システムは、リモート印刷サーバーと通信できるように構成する必要があります。
- **lpd** リモート・サブシステムを始動するには、root 権限が必要です。

ここで解説する情報は AIX の特定バージョンを使用してテストされたものです。 したがって、その内容は使用される AIX のバージョンおよびレベルによってかなり異なることがあります。

システム・プロンプトで「**smit mkitab_lpd**」と入力し、指示に従ってこの作業を完了します。

次のコマンドを使用してこの作業を実行することもできます。

- すぐに **lpd** リモート・サブシステムを始動するには、以下のようにします。

  ```
  startsrc -s lpd
  ```

- 次のシステム再始動時に **lpd** リモート・サブシステムを始動するには、以下のようにします。

  ```
  mkitab "lpd:2:once:startsrc -s lpd"
  ```

- 現時点と次のシステム再始動時の両方で **lpd** リモート・サブシステムを始動するには、以下のようにします。

  ```
  startsrc -s lpd; mkitab "lpd:2:once:startsrc -s lpd"
  ```

lpd リモート・サブシステムの停止

lpd リモート・サブシステムを停止するには、**stopsrc** コマンドまたは **rmitab** コマンド、あるいは SMIT を使用することができます。

- システムは、リモート印刷サーバーと通信できるように構成する必要があります。
- **lpd** リモート・サブシステムを停止するには、root 権限が必要です。

ここで解説する情報は AIX の特定バージョンを使用してテストされたものです。 したがって、その内容は使用される AIX のバージョンおよびレベルによってかなり異なることがあります。

システム・プロンプトで「**smit rmitab_lpd**」と入力し、指示に従ってこの作業を完了します。

次のコマンドを使用してこの作業を実行することもできます。

- すぐに **lpd** リモート・サブシステムを停止するには、以下のようにします。

  ```
  stopsrc -c -s lpd
  ```

- 次のシステム再始動時に **lpd** リモート・サブシステムを停止するには、以下のようにします。

  ```
  rmitab "lpd"
  ```

- 現時点と次のシステム再始動時の両方で **lpd** リモート・サブシステムを停止するには、以下のようにします。

  ```
  stopsrc -c -s lpd; rmitab "lpd"
  ```

双方向データの印刷

双方向 (bidi) のデータの印刷は、テキストの性質のため特殊の取り扱いを必要とします。ほとんどの場合、データは入力された順序 (論理発注方式など) で保管され、テキストはレイアウト変換の処理を受ける必要があります。この処理には、再配列が伴い、表示される文字がユーザーにとって読み取り可能な順序で表示されます。データがアラビア語スクリプトを含む場合、処理には *shaping (成形)* という追加のステップが組み込まれ、そこでは文字はテキスト内部の位置に従って正確な絵文字に変換されます。

Bidi データは主に 2 つの形式 (イメージまたはテキストのいずれかとして) で印刷できます。イメージ形式のデータは、印刷されるはずの正確な方法で保管されるので、一般に変更の必要はありません。テキストとして保管されるところでは、データが保管される順序の方式に基づき、レイアウト変換の必要があるか否かにより、プリンターに送信される前にレイアウト変換処理が必要になる場合があります。データが視覚順序付け方式で保管され、形作られる場合、これはレイアウト変換が必要であることを意味します。データが論理順序付け方式で保管される場合、これは再配列と成形が必要であることを意味します。必要な場合、bidi 印刷フィルターは、テキストがプリンターに送信される前に、テキストに対するこのレイアウト変換の実行の主な役割を果たします。

双方向印刷フィルター

AIX に組み込まれている bidi 印刷フィルターでは、テキスト・ベースのデータのみがサポートされます。それは、前処理フィルターとして AIX 印刷サブシステムの一部分として実行されるように設計されています。

3 つの別個のフィルターがあります。

項目	説明
/usr/bin/bprt	ASCII データ・ストリーム用の bidi 印刷フィルター。
/usr/bin/pcl_bprt	PCL データ・ストリーム用の bidi 印刷フィルター。
/usr/bin/ebprt	表形式レイアウト内のデータのサポートを持つ bidi 印刷フィルター。

フィルター **/usr/bin/brpt** および **/usr/bin/pcl_bprt** は、顧客の既存のサポートを維持するために保持されます。**/usr/bin/ebprt** フィルターが表形式のデータ対応機能拡張を持つので、ユーザーが **/usr/bin/ebprt** フィルターを使用することが推奨されます。

/usr/bin/ebprt フィルターを使用しての双方向データの印刷

/usr/bin/ebprt 印刷フィルターは処理の方式に影響を及ぼす 2 つのモードをサポートします。

通常モード
データは、一度にテキストの単一行として処理されます。

テーブル・モード
データは表形式レイアウトにフォーマット設定されています。この操作のモードの要件は、印刷すべきデータが単一行として処理できない表形式フォーマットであるとき、発生します。データが単一行のテキストとして処理しようとすると、フィールドが不正確に位置合わせされ、目的とする表形式構造を損なう結果になります。このため、フィルターは、テーブル内のフィールドの位置を保持するために、処理されるデータのレイアウトを理解する必要があります。

表形式データのサポートおよび期待すべきもの

/usr/bin/ebprt がレポートのフォーマットを理解するために、テキストがフィルターにより受け取られる前に、テーブル内のフィールドの開始と終了を示すセパレーターを組み込んで置く必要があります。セパレーターの挿入は、ユーザーによるファイル生成段階中に行う必要があります。これにより、フィルターは、フィールドの順序を破損しないでデータを処理できます。フィルターには、ユーザーの要求を満たすためにプリンターの最終出力からこれらのセパレーターを隠すオプションが備わっています。使用されるセパレーターを記述する情報は、ユーザーが構成ファイル **/etc/ebprt.conf** に指定する必要があります。データ・フィールドを区切るために、要求された位置に埋め込まれたセパレーターを指定すると、フィルターはフォーマット済みのレイアウトが失われないことを保証します。報告書作成の段階でセパレーターの挿入が困難である、または実行可能でない場合、フィルターはフィールドのセパレーターとして空白文字を使用するように構成できます。このオプションは、出力が希望する形式と一致することを保証しないので、注意して使用する必要があります。

ロケールおよびコード・ページのサポート

フィルターは、Bidi ロケールをサポートするためのみに設計されています。これには、アラビア語ロケールおよびヘブライ語ロケールが対応するコード・ページとともに含まれます。AIX ロケール情報用に「*ナショナル・ランゲージ・サポート ガイドおよびリファレンス*」の付録 A で使用可能なテーブル 3 (アラビア語) およびテーブル 20 (ギリシャ語) を参照してください。ロケール・サポートの問題については、48 ページの『エスケープ・シーケンスおよび PCL サポートの既知の制限』を参照してください。

インストールおよびパッケージ化

bidi 印刷フィルターは、**bos.loc.bidi** ファイルセットの一部分としてインストールされます。

双方向印刷フィルターを使用するための印刷サブシステムの構成

bidi 印刷フィルターは、AIX 印刷プロセスに統合化されます。これは、構成の完了後、ユーザーは、 **lp**、**qprt**、または **enq** などの通常の印刷コマンドを使用して bidi データを印刷できることを意味します。ユーザーは、印刷コマンドを実行する前に環境変数を設定することにより、bidi フィルターの動作を制御できます。

/usr/bin/ebprt 印刷フィルターを AIX 印刷サブシステムの一部として構成するには、以下の手順を実行します。

1. 印刷キューを作成します。
2. 以下の手順を実行して、SMIT 内のプリンター・キュー属性に **/usr/bin/ebprt** プリンター・フィルターを追加します。

SMIT メインメニューから、**Print Spooling > AIX Print Spooling > Change / Show Print Queue Characteristics** をクリックします。前のステップで作成された Queue を選択します。**Default Print Job Attributes** をクリックし、次の項目を設定します。

a. 事前処理の FILTER NAME を "p" に

b. 印刷ファイルの CODE PAGE

c. プリンターの CODE PAGE

SMIT メインメニューから、**Print Spooling > AIX Print Spooling > Programming Tools** をクリックします。**Change / Show Pre-processing Filters** を選択します。**pr** フィルターを **usr/bin/ebprt -w%IwW** に設定します。

注: 印刷しているデータがアラビア語の場合、データが「ISO8859-6」エンコードであるとしても、印刷ファイルのコード・ページ値は常に「IBM-1046」に設定する必要があります。これは、プリンターに送信される出力が成形絵文字を確実に組み込ませるためです。

3. 適切な環境変数を設定します。
   - LANG: フィルター用に実行するインストール済み bidi ロケール。
   - EBPRTCFG: 構成ファイルからロードされるプロファイル名。
4. 印刷コマンドを実行します。

   これは、定義されているキュー名を持つファイル名上で、**lp**、**qprt**、または **enq** コマンドの実行により行うことができます。

/etc/ebprt.conf 構成ファイルの構成

/etc/ebprt.conf 構成ファイルはスタンザ内で構造化されています。各スタンザは、プロファイルを示し、これは実行時中に名前によりロードされます。プロファイル名は、EBPRTCFG 環境変数内に設定され、この環境変数はフィルターの実行前にロードする必要があります。デフォルトで、EBPRTCFG 環境変数は、**default** と名付けられたプロファイルであると想定され、このプロファイルは **/etc/ebprt.conf** 構成ファイルから読み取られます。ユーザーは、デフォルトの動作を変更するため、このプロファイルを一部変更でき、また、このデフォルトの動作は、フィルターが実行される前に EBPRTCFG 環境変数が設定されていない場合に備えて使用されます。これは、**default** プロファイルに対してのみ適用されます。ユーザー定義のすべての他のプロファイルは、明示的にロードする必要があります。

セキュリティーおよびアクセス権

構成ファイルは、デフォルトで **root** ユーザーおよび printq グループにより所有されます。ファイルに設定されるアクセス権により、**group printq** ユーザーおよび **root** ユーザーにのみファイルへの書き込みが許可されますが、全員が読み取り権限を持ちます。

構成オプション

以下の表では、対応する受信済み値とともに構成ファイル・パラメーターをリストします。

以下に示す属性は、次の構文で設定する必要があります。

AttributeName = Value

属性名	考えられる値	説明
PageDirection	RightToLeft (デフォルト)	この値は、ページの中でのテキストの方向を定義します。
	LeftToRight	この値は、ページの中でのテキストの方向を定義します。
NumeralShaping	National	数表示が National (Arabic-Indic) として表示されるように指定します。この属性はアラビア語ロケールに対してのみ適用されます。
	Nominal	数表示が National (Arabic-Western) として表示されるように指定します。この属性はアラビア語ロケールに対してのみ適用されます。
	Contextual (デフォルト)	数表示がコンテキスト内部での位置に基づき表示されるように指定します。この属性はアラビア語ロケールに対してのみ適用されます。
SymmetricSwapping	Yes (デフォルト)	対称スワッピングが使用可能であることを指定します。
	No	対称スワッピングが使用不可であることを指定します。
ArabicSpecialShaping	Yes	アラビア語の twocell 形状の活動化を指定します。これは、LamAlef ファミリー文字および Seen ファミリー文字が 1 つの文字の代わりに 2 つの文字として印刷されることを示します。これは、通常の成形動作に優先します。
	No (デフォルト)	アラビア語の twocell 成形が使用不可であることを指定します。通常の成形が実施されます。
FilterMode	普通紙 (デフォルト)	ファイル内のテキストが非構造化であるとして扱われることを指定します。このオプションが指定された場合、すべてのセパレーター関連の属性は無視されます。
	Tabular	ファイル内のテキストが表形式のレイアウトであるとして扱われることを指定します。
Separator	印刷可能文字または文字のリストデフォルトは垂直バー (|) です。	セパレーターがテーブル内のフィールドを区切ることを指定します。表形式テキストの単一行内で最大 2 つまでの異なるセパレーターが可能です。しかし、セパレーターの長さは、一貫性のあるものである必要があります。文字は、印刷可能文字である必要があります。例: Separator = +| は、ファイルに正符号 (+) セパレーターおよび垂直バー (|) セパレーターの両方が組み込まれていることを示します。セパレーターの長さは、SeparatorLength 属性によって指定されます。
	キーワード SPACE	表のフィールドを区切るため、空白文字が使用されることを指定します。空白文字の数は、SeparatorLength 属性によって指定されます。
SeparatorLength	正数。デフォルトは、1 です。	セパレーターの長さを指定します。これは、単一ファイル内で一貫性がなければなりません。値は正で、最大 2 桁でなければなりません。
HideSeparator	Yes (デフォルト)	セパレーターが表示されないことを指定します。セパレーターは空白文字で取り替えられます。
	No	セパレーターが表示されることを指定します。
FieldDirection	RightToLeft	この値は、フィールド内部でのテキストの方向を定義します。
	LeftToRight	この値は、フィールド内部でのテキストの方向を定義します。
	Contextual (デフォルト)	この値は、フィールド内部でのテキストの方向を定義します。方向および位置合わせはテキスト内の最初の文字に依存します。最初の文字が **bidi** 文字である場合、値は RightToLeft です。最初の文字が Latin である場合、値は LeftToRight です。

属性名	考えられる値	説明
TableColumnOrderReverse	Yes (デフォルト)	表形式レイアウトの欄の順序が反転**される**ことを指定します。指定されない場合、この値は PageDirection から継承されます。PageDirection = RightToLeft の場合、値は Yes です。PageDirection = LeftToRight の場合、値は No です。
	No	表形式レイアウトの欄の順序が反転**されない**ことを指定します。指定されない場合、この値は PageDirection から継承されます。PageDirection = RightToLeft の場合、値は Yes です。PageDirection = LeftToRight の場合、値は No です。
HeaderSize	正数。	見出しとして取り扱う必要がある行数を指定します。これらの行は、ロード済みプロファイルが表形式モードを示す場合でも、テーブルの部分とは見なされません。許可される値は、最大で 3 桁の正数です。デフォルト値 0 は、この機能を使用不可に**します**。
	0 (デフォルト)	
CoverPages	正数。	カバー・ページとして取り扱う必要があるページ数を指定します。これらの行は、ロード済みプロファイルが表形式モードを示す場合でも、テーブルの部分とは見なされません。許可される値は、最大で 3 桁の正数です。デフォルト値 0 は、この機能を使用不可に**します**。
	0 (デフォルト)	
Logging	Yes	メッセージは、デバッグ問題に役立つためにログに記録**されます**。
	No (デフォルト)	メッセージはログに記録**されません**。
in_orientation (高機能)	RTL	入力方向値が **bidi レイアウト・エンジン**にパスされることを指定します。
	LTR	
	Contextual	
out_orientation (高機能)	RTL	出力方向値が **bidi レイアウト・エンジン**にパスされることを指定します。
	LTR	
	Contextual	
in_typeoftext (高機能)	Implicit	入力 typeoftext 値が **bidi レイアウト・エンジン**にパスされることを指定します。
	Visual	
out_typeoftext (高機能)	Implicit	出力 typeoftext 値が **bidi レイアウト・エンジン**にパスされることを指定します。
	Visual	
in_swapping (高機能)	Yes	入力スワッピング値が **bidi レイアウト・エンジン**にパスされることを指定します。
	No	
out_swapping (高機能)	Yes	出力スワッピング値が **bidi レイアウト・エンジン**にパスされることを指定します。
	No	
in_numeralshaping (高機能)	Nominal	入力数表示形状値が **bidi レイアウト・エンジン**にパスされることを指定します。
	National	
	Contextual	
out_numeralshaping (高機能)	Nominal	出力数表示形状値が **bidi レイアウト・エンジン**にパスされることを指定します。
	National	
	Contextual	
in_textshaping (高機能)	Nominal	入力テキスト形状値が **bidi レイアウト・エンジン**にパスされることを指定します。
	Shaped	
out_textshaping (高機能)	Nominal	出力テキスト形状値が **bidi レイアウト・エンジン**にパスされることを指定します。
out_textshaping (高機能)	Shaped	

注:

1. 「高機能」とマークを付けられた属性は、それに続く仕様に従って、レイアウト・エンジンに直接送信される値を指定するために予約されます。このタイプの属性を使用するときは、特に注意してください。それらは (入出力の) 対で設定する必要がありますが、1 つの値のみが設定される場合、マッチング属性の値は、定義されるもう一方の一般属性から継承されます。例えば、in_swapping 属性が「yes」に設定され、out_swapping が設定されていない場合、存在しない値を識別するため SymmetricSwapping 属性が使用されます。
2. 属性キーワードおよび対応する値には、大/小文字の区別はありません。しかし、ワードの最初の文字に大文字を使用すると、構成ファイルが読みやすくなります。

/etc/ebprt.conf 用サンプル構成ファイル

以下に **/etc/ebprt.conf** 用サンプル構成ファイルを示します。

```
#
#   These values reflect the default /usr/bin/ebprt behavior.
#
[Default]
PageDirection = RightToLeft
NumeralShaping = Contextual
SymmetricSwapping = Yes
ArabicSpecialShaping = No
FilterMode = Normal
Separator = |
SeparatorLength = 1
HideSeparator = Yes
FieldDirection = Contextual
TableColumnOrderReverse = Yes
HeaderSize = 0
CoverPages = 0
Logging = No
```

構成の例

次の例では、**MyReport** という名前を持つ構成プロファイルとともに IBM-1046 データを持つファイルを印刷するために、印刷キュー構成の間に設定された値を説明します。

サンプル構成プロファイル

以下のサンプル構成プロファイルは、英語テキストおよびアラビア語テキストを持つ右から左方向の表レポートを印刷するために使用され、その表レポートではフィールドを分離するために 3 つの空白文字が使用されます。フィールド自身は右から左の方向を持ち、数表示はアラブ・インド形式です。

```
[myreport]
PageDirection = RightToLeft
FieldDirection = RightToLeft
FilterMode = Tabular
Separator = SPACE
SeparatorLength = 3
NumeralShaping = National
```

印刷キューの構成

プリンター・キューの構成を作成するには、次の手順を実行します。

1. **ibmeg** と名付けられた印刷キューを作成します。
2. SMIT を使用して、キュー **ibmeg** に **/usr/bin/ebprt** 印刷フィルターを追加します。
   - 前処理の **FILTER NAME** を **p** に設定します。
   - 印刷ファイルの **CODE PAGE** を **IBM-1046** に設定します。
   - プリンターの **CODE PAGE** を **ibm.1046** に設定します。
   - **pr** フィルターを **/usr/bin/ebprt -w%IwW** に設定します。
3. 環境変数を次のように設定します。

   ```
   $ export LANG=Ar_AA
   $ export EBPRT=myreport
   ```

4. **report1** と呼ばれるファイルを印刷します。

   ```
   $ lp –d ibmeg report1
   ```

トラブルシューティング

/etc/ebprt.conf ファイル内にロギング 属性が指定されていると、syslogd デーモンを使用してメッセージをログに記録できるようになります。メッセージは、syslogd 機能 **LOG_LPR** を使用して、優先順位 **LOG_DEBUG** でログに記録されます。syslogd をセットアップする方法の詳細については、コマンド解説書の syslogd デーモンの項を参照してください。

以下に例を示します。

/tmp/ebprt.log という名前のファイルが作成されていて、syslogd デーモンが実行中の場合は、/etc/syslogd.conf ファイルの下にある以下のエントリーがデバッグ・メッセージをキャッチします。

lpr.debug

エスケープ・シーケンスおよび PCL サポートの既知の制限

/usr/bin/ebprt 印刷フィルターによって読み取られるデータは、プレーン・テキストであると予期され、エスケープ・シーケンスがありません。エスケープ・シーケンスはサポートされません。

UTF-8 のサポート

bidi 印刷フィルター、**/usr/bin/ebprt** は、UTF-8 データを現在サポートしていません。

印刷スプーラー

スプーラーのジョブは、キューイング・システム とも呼ばれるもので、特に、複数のプリンターがあるシステム上で、プリンターの使用を管理します。

スプーラー・バックエンド

バックエンド **piobe** は、ローカル・キュー上の印刷ジョブを処理するために使用されるもので、最も頻繁に使用され、基本オペレーティング・システムに付属している最も複雑なバックエンドである可能性があるため、このセクションでは主要な例として使用されています。この方法で **piobe** を使用すると、基本オペレーティング・システムのスプーラー概念をよりよく展開することができます。

このセクションの目的は、スプーラーが、始まりと終わりがあり、さらにその間に離散ポイント (ブラック・ボックスなし) がある実際のプロセスの 1 つであることを例示することです。スプーラーとは、相互作用が特定のキューの構成方法に完全に依存する、一連のコンポーネントです。このことを認識することが、以下の結果につながります。

- 問題判別および解決がさらに容易になる。
- スプーラーを具体的なビジネス・ニーズに特化することがさらに容易になる。
- 以前に検討したことのない、スプーラー変更の機会が見えてくる可能性がある。

フォーマッター・フィルター

フォーマッター・フィルター は、ローカル・プリンター・キューのデフォルト・バックエンド **piobe** によって作成および実行されるパイプラインの一部です。

フォーマッター・フィルターは、入力パラメーターに基づいて、入力ファイルのフォーマットを設定するか、または変更せずに引き渡す機能を備えています。フォーマッターが入力ファイルを変更せずに引き渡す場合でも、入力ファイルの印刷前にプリンターを初期化するプリンター・コマンドを送信し、印刷の完了後にはプリンターを元の状態に復元します。

フォーマッター・フィルターは、仮想プリンターのコロン・ファイルを使用して、スプーラー印刷ジョブの幅広い操作を行う機能を備えています。

ローカル・プリンターとリモート・プリンター

ローカル・プリンター は、ローカル・ホストに接続された実プリンターで、このプリンターにはローカル・キューが存在します。

このキューに対して実行依頼されたジョブはすべて、キューが存在するホスト上で処理され、印刷されます。*リモート・プリンター* は、リモート・ホストに接続された実プリンターです。リモート・プリンターのキューは、バックエンドを指定します。バックエンドの機能は、ローカル・ホストからネットワークを経由してリモート・ホストにスプール・ジョブを送信することです。このキュー (ローカル・ホスト上にある) に対して実行依頼されたジョブはすべて、ネットワーク経由でリモート・ホストに送信されて、このホスト上で処理され、印刷されます。

プリンター・デバイス

プリンター/プロッター・デバイス は、実プリンターを表す `/dev/directory` 内のスペシャル・ファイルです。

このファイルは、リダイレクトによって使用でき (例: `cat FileName > /dev/lp0`)、またユーザー作成のコンパイル済みプログラムに使用することもできます。このデバイス・ドライバーの設定値は、**splp** コマンドを使用して表示および変更できます。スプーラー・コマンドがプリンター・デバイスにアクセスできるように、事前にそのデバイス用の印刷キューを作成する必要があります。

qdaemon プロセス

qdaemon は、ジョブ要求と、ジョブの完了に必要なリソースの両方を追跡します。リソースは実プリンター、その他の実デバイス、またはファイルの場合もあります。

qdaemon は、**srcmstr** プロセスの補助を受けてバックグラウンドで実行されるプロセスです。システムをオンにすると、**startsrc** コマンドが **qdaemon** を開始します。 **qdaemon** は **startsrc** コマンドによって開始でき、**stopsrc** コマンドによって停止できますが、**qdaemon** はシグナル通信のみをサポートするので、**lssrc** コマンドから照会を受けることはできません。

qdaemon は、ジョブ要求と、ジョブの完了に必要なリソースの両方を追跡します。リソースは実プリンター、その他の実デバイス、またはファイルの場合もあります。**qdaemon** は、未処理の要求のキューを維持し、適切な時点で適切なデバイスに要求を送ります。また **qdaemon** は、システム・アカウンティング用にプリンターの使用状況データも記録します。**qdaemon** は、スプーラー・キュー用のバックエンドの実行を開始します。

qdaemon が異常終了すると、**srcmstr** によって再始動されます。

注: srcmstr デーモンは停止しないでください。このプロセスは、システム上で稼働する他のデーモンを制御します。

実 (物理) プリンターと仮想プリンター

実 (物理) プリンター は、シリアル・ポートまたはパラレル・ポート経由で、またはネットワーク端末サーバーなどのネットワーク接続を経由して、システムに接続されたプリンター・ハードウェアです。

実プリンターが、システムのローカル側にあるシリアル・ポートまたはパラレル・ポート経由で接続されている場合は、カーネル内のプリンター・デバイス・ドライバーがプリンター・ハードウェアとやり取りし、プリンター・ハードウェアと仮想プリンター間のインターフェースを提供します。

仮想プリンター は、ハイレベルのデータ・ストリーム (ASCII や PostScript など)、およびそのデータ・ストリームの処理方法を定義する属性およびそれに関連した値のセットです。これには、実プリンターをホスト・コンピューターに接続する方法や、データのバイトをプリンターとの間で送受信するために使用されるプロトコルに関する情報は含まれません。**piobe** バックエンドは、仮想プリンター定義に格納された情報を使用して、印刷ジョブの処理を制御します。属性およびそれに関連した値のセットを格納する物理ストレージ・メディアは、*プリンター・コロン・ファイル* と呼ばれます。

スプーラーの機能およびサービス

基本オペレーティング・システム・スプーラー は、プログラム、構成ファイル、およびデータ・ファイルの集合です。

基本オペレーティング・システム・スプーラーは、次の機能またはサービスを提供します。

- キューを作成するための機能を備えています。キューは、特定の方法でジョブを処理する機能を備えたソフトウェア・エンティティーです。
- ユーザーは、ジョブ (通常はプリンター・ジョブ、ただしこれに限りません) の処理をキューに実行依頼できます。
- デバイス (実プリンターなど)、またはプログラム (コンパイラーなど) へのシリアル・アクセスをキューによって提供し、単一のデバイスまたはプログラムが複数のユーザーに同時に使用されないようにします。
- ユーザーは、状況ファイルによってキューの状況を照会できます。
- ユーザーは、キューの可用性、およびジョブの状況を制御できます。
- 印刷ジョブ・データ・ストリームのさまざまな操作を実行します。
- 処理済みジョブの幅広いデリバリー・メカニズムを備えています。

スプーラー・バックエンド

スプーラー・バックエンド は、処理のためにキューに入れられたスプーラー・ジョブを管理するために、スプーラーの **qdaemon** コマンドによって開始されるプログラムの集合 (パイプライン) です。

印刷キュー用のバックエンドの場合、スプーラー・バックエンドは通常次の機能を実行します。

- **qdaemon** コマンドから、処理される 1 つ以上のジョブのリストを受け取ります。
- 印刷ジョブに対して、データベースから取得したプリンターとフォーマットの属性 (コマンド・ラインで指定されたフラグによって指定変更される) を使用します。
- 印刷ジョブを処理する前に、プリンターを初期化します。
- ASCII 文書の単純なフォーマット設定用のフィルターを提供します。
- フィルターを使用して、印刷ジョブ・データ・ストリームをプリンターがサポートするフォーマットに変換します。
- 各国語文字の印刷のサポートを提供します。
- 印刷ジョブのフィルター済みデータ・ストリームをプリンター・デバイス・ドライバーに渡します。
- 印刷ジョブに対してヘッダー・ページとトレーラー・ページを生成します (要求された場合)。
- 複数部の印刷ジョブを生成します (要求された場合)。
- 用紙切れ、要介入、およびプリンター・エラーの各条件を報告します。
- フィルターによって検出された問題を報告します。
- ジョブが取り消された後にクリーンアップを行います。
- 印刷ジョブに対して、特定の印刷要件を満たすためにカスタマイズできる環境を提供します。

コンパイラーなどのバックエンドは当然コマンド・ラインから直接実行できますが、プリンター・バックエンド・プログラムをユーザーが直接実行することは通常ありません。 **qdaemon** がバックエンドを実行し、ファイルの名前と、ユーザーが指定したジョブ制御フラグをバックエンドに送信します。バックエンドは、`/var/spool/lpd/stat` ディレクトリー内の状況ファイルを介して **qdaemon** とやり取りします。ユーザーは、**qchk** や **lpstat** などのキュー状況照会コマンドを使用して、状況情報を表示できます。例えば、印刷ジョブの場合は、プリンター状況、印刷されるページ数、ジョブの完了パーセンテージなどの情報があります。

基本オペレーティング・システム内では、**piobe** がローカル印刷ジョブを処理するための標準スプーラー・バックエンドです。

スプーラー・ジョブ

スプーラー・ジョブ は、ユーザーがスプーラーに実行依頼する任意のジョブです。

ジョブ実行依頼コマンドはすべて、処理を必要とする 1 つ以上のファイルの名前を末尾に指定する必要があります。例えば、キーワードをバックエンドに渡して、バックエンドが実行する機能をキーワードによって制御することはできません。実行依頼されたジョブは、ファイルシステム内に存在している必要があります。

スプーラーは、さまざまなタイプのジョブを受け入れます。特定のキューのバックエンドが、そのキューに対して実行依頼されたジョブを処理する能力があるかどうかの確認は、システム管理者が行う必要があります。

プリンター・ジョブには次のようなタイプがあります。

- ASCII

- Postscript
- PCL
- HPGL
- GL
- Diablo 630
- ditroff

汎用基本オペレーティング・システム・スプーラー

基本オペレーティング・システム・スプーラーは、印刷ジョブ専用のスプーラーではなく、プリンター・キューに入れられた印刷ジョブを含む、さまざまなタイプのジョブのキューイングに使用できる汎用スプーリング・システムです。

スプーラーは、キューに入れるジョブのタイプを判別できません。キューの作成時に、キューの機能はそのキューのスプーラー・バックエンドによって定義されます。例えば、キューが作成され、キュー・バックエンドが **piobe** (ローカル・プリンター・キュー用のデフォルト・プリンター入出力バックエンド) に設定されている場合、そのキューは印刷キューです。同様に、キュー・バックエンドが **cc** (またはその他のコンパイラー) に設定されている場合、そのキューはコンパイラー・ジョブ用です。スプーラーの **qdaemon** コンポーネントがキューからジョブを選択すると、キューのバックエンドを呼び出してジョブを処理します。

ここでは、スプーラーを入り口点、出口点、および中間の点からなる汎用スプーリング・サブシステムとして扱います。スプーラーに対して実行依頼されたジョブは、システムに入り (ジョブの実行依頼)、点から点への予測可能なパスをたどり (ジョブ処理)、その後でシステムから抜けます (ジョブのデリバリーおよびクリーンアップ)。システム内でのジョブのフローを理解することは、複雑なタスクを実行するためにキューを構成し、問題判別と解決を有効に行うために重要です。次に、キューが印刷キューである場合に特に注目しながら、このジョブ・フローについて詳しく説明します。

スプーラーを構成する部分

基本オペレーティング・システム・スプーラーは、始点、中間点、および終点からなるプロセスまたはサブシステムと見なすことができます。

タスクを実行するために、基本オペレーティング・システム・スプーラーは次の 4 つの基本部分で構成されています。

1. **enq** コマンドは、スプーラーへの実際の入り口点です。したがって、すべてのスプーラー・アクティビティーの始まりです。このコマンドは、ジョブ処理要求を受け入れます。
2. **qdaemon** が受け持つ処理は、**enq** コマンドからスプーラーに対して実行依頼されたジョブをすべて受け入れ、追跡することです。また、必要なリソースがすべて使用可能になった後、キュー・バックエンドがジョブを処理できるようにします。**qdaemon** は、スプーラー・プロセスの中間点の 1 つです。
3. スプーラー・バックエンドは、スプーラーの **qdaemon** コマンドによって呼び出され、いずれかのキュー内のジョブを処理するプログラムの集合です。バックエンドは、プリンターなど特定のデバイスに出力を送ります。バックエンド **piobe** の場合、バックエンドにはフォーマッター・フィルターが含まれており、フォーマッター・フィルターにはプリンター・コロン・ファイルが含まれています。バックエンドは、中間点の 1 つであると同時に終点でもあります。これは、処理済みジョブを最終出力先に配送する特定のプロセスがバックエンドに含まれているからです。

4. 構成ファイル /etc/qconfig は、使用可能なキューおよびデバイスの構成を記述します。**enq** コマンドと **qdaemon** コマンドの両方が、構成ファイルを参照します。この構成ファイルは、基本オペレーティング・システム・スプーラー全体の正しい動作に不可欠なものなので、概念上はスプーラーの他の 3 つの部分と同じく重要なものと考えられます。

スプーラー・データ・フロー: コマンドおよびバックエンド

基本オペレーティング・システム・スプーラーに対してジョブを実行依頼するために、4 つのコマンドを使用できます。これらのコマンドは、**lp**、**lpr**、**qprt**、および **enq** です。

これらのコマンドはそれぞれ、異なる UNIX を起源としています。**lp** の起源は AT&T System V で、**lpr** の起源は BSD、**qprt** と **enq** の起源は両方とも基本オペレーティング・システムです。

ユーザーはこれら 4 つのコマンドのどれを使用してもジョブをスプーラーに実行依頼できますが、スプーラーへの実際の入り口点は **enq** コマンドです。 **lp**、**lpr**、および **qprt** はすべて、**enq** のフロントエンドです。**lp**、**lpr**、および **qprt** はすべて、引数を解析して **enq** への呼び出しを作成します。フロントエンドは、それぞれの動作、およびそれぞれが受け入れるフラグの数とタイプに関して、互いに異なります。

ジョブをスプーラーに実行依頼すると、**enq** はジョブ要求を処理します。ジョブ要求が有効ならば、基本的にはコマンド構文が正しいことになり、ジョブはキューに入れられます。**enq** はジョブ記述ファイル (JDF) を作成し、**qdaemon** に新規 JDF の存在を通知します。

qdaemon は、それぞれの新規 JDF を読み取り、ジョブ要求を追跡するために維持している内部データ構造に、JDF によって指定されたジョブ・パラメーターを読み込みます。**qdaemon** は、キュー状況情報を使用してそれぞれのキューの状況を追跡し、環境が整ったら、キューのジョブを処理するためのバックエンドを呼び出します。

キューのバックエンドは、そのキューに配置されたジョブの処理方法を厳密に決定します。ユーザーがジョブをスプーラーに実行依頼するために使用するコマンドは、ジョブの特定の処理を要求するフラグを指定でき、また **qdaemon** はジョブが処理される時期を決定できますが (最短ジョブ最優先、または先着順サービス)、ジョブの処理に関して言えば、バックエンドがすべての作業を実際に行うプロセスです。 (システム管理者は、/etc/qconfig 内のスタンザを確認し、バックエンドを調べるだけで、特定のキューの機能を簡単に判別できます。)

次の図に、バックエンドに関する最も一般的な 2 つのシナリオ、ローカル・プリンター・キューとリモート・プリンター・キューを示します。ローカル・キューは、バックエンドとして **piobe** (プリンター入出力バックエンド) を使用します。リモート・プリンター・キューは、バックエンドとして **rembak** (REMote BAcKend、リモート・バックエンド) を使用します。

piobe は、すべてのバックエンドと同様に、**qdaemon** によって呼び出されます。**piobe** は、一連のプログラム (パイプライン) をセットアップし、制御します。パイプラインは、印刷ジョブの幅広い制御を行うだけでなく、多量の制御データをプリンターに送信して、例えば処理対象のジョブをプリンターに配送する前にプリンターを特定モードに初期化します。**piobe** は、プリンター・コロン・ファイルに格納されているデータを最初に使用します。**piobe** によってセットアップおよび制御される、パイプライン内の最後のプログラムが、その前にパイプライン内で生成されたバイト・ストリームの物理的な配送を受け持ちます。ローカル・キューの観点から見ると、このプログラムは、ローカル接続プリンター (シリアル接続またはパラレル接続)、またはネットワーク接続プリンターにバイト・ストリームを配送するデバイス・ドライバーを開きます。

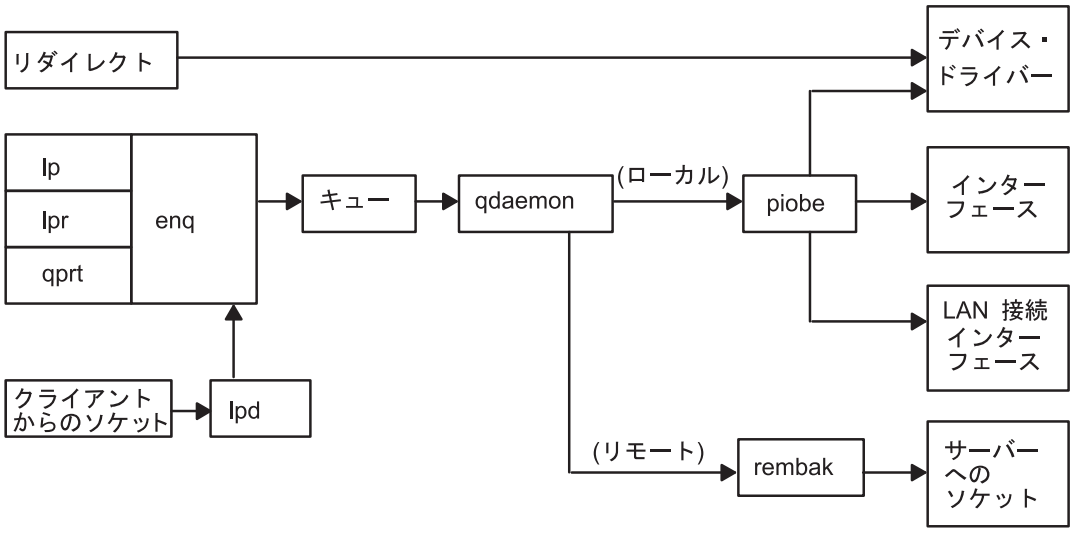

図 1. 基本オペレーティング・システムによる印刷

rembak は、リモート・プリンター・キューが別のホスト上のキューを指している場合の一般的なバックエンドで、印刷サーバーと呼ばれることがよくあります。 **piobe** は印刷ジョブの幅広い操作を実行できますが、**rembak** は TCP/IP ネットワーク経由で印刷サーバーにジョブを転送する処理のみを行います。「基本オペレーティング・システムによる印刷」の図に示したとおり、印刷サーバーが基本オペレーティング・システムをベースとした別のマシンであれば、**rembak** はジョブをネットワーク経由で **lpd** プロセスに転送します。次にこのプロセスは **enq** を呼び出し、このプロセスは JDF を作成し、以下も同様に前の図のとおり処理が進みます。

スプーラー・データ・フロー (enq コマンド)

コマンド **lp**、**lpr**、**qprt**、および **enq** を使用して、ジョブの処理をスプーラーに実行依頼できます。

enq コマンドが、スプーラーへの実際の入り口点です。**lp**、**lpr**、および **qprt** はすべて、自身の引数を解析して、**enq** への呼び出しを作成します。このことを実際に確認するには、root ユーザーとしてシェル・プロンプトで次の手順を行ってください。

1. `mount /bin/echo /bin/enq` と入力します。
2. `qprt -Pasc -fp -z1 -p12 -s courier -C -N 3 /etc/motd` と入力します。
3. `umount /bin/enq` と入力します。

ステップ 2 の **qprt** コマンドは、印刷ジョブをスプーラーに実行依頼し、**asc** という名前のキューに入れるように指示して、毎日のメッセージを 12 ポイント 90 度回転の Courier フォントで 3 部要求します。**qprt** は、コマンド・ライン引数を解析して、**enq** に渡す引数ベクトルを作成します。**qprt** コマンドが引数ベクトルを使用して **enq** の呼び出しを試行するとき、このコマンドは **enq** コマンドより上位にマウントされている **echo** コマンドを代わりに呼び出します。このため、**qprt** コマンドによって生成された引数ベクトルは **echo** コマンドに渡され、このコマンドは引数ベクトルをディスプレイに単にエコー出力します。この手順は、**lp** と **lpr** に対しても同様に機能します。**qprt** が実際は **enq** のフロントエンドであることが分かるほかに、この手法はサポートされないフラグをスプーラーに送る方法を確認するためにも便利です。詳しくは、 145 ページの『フィルター』を参照してください。

ステップ 2 で **qprt** コマンドを実行すると、**TERM** 環境変数に指定された表示エレメントに、次の出力が書き込まれます。

```
-P asc -o -o -f -o p -z -o 1 -o -p -o 12 -o -s courier -C -N 3 /etc/motd
```

これは、**qprt** コマンドの特定インスタンスによって生成される引数ベクトルです。 **echo** が **enq** より上位にマウントされていなければ、次のジョブ実行依頼コマンドが実行されます。

```
enq -P asc -o -f -o p -o -z -o 1 -o -p -o 12 -o -s courier -C -N 3 /etc/motd
```

ジョブ実行依頼コマンドの末尾には、基本オペレーティング・システムからアクセス可能なファイルシステムに存在する 1 つ以上の実ファイル名を指定する必要があります。これは、キューが印刷ジョブ以外のジョブを処理するようにセットアップされている場合にも当てはまります。

注: ステップ 3 は必ず行ってください。そうしなければ、スプーラーが使用不可になります。

enq コマンドを実行すると (直接、または **lp**、**lpr**、または **qprt** によって)、このコマンドはジョブにジョブ番号を割り当てます。デフォルトでは、**lp** はジョブ番号を戻します。**lpr** および **qprt** は、フラグを使用して明示的に要求しない限り、ジョブ番号を戻しません。

enq は JDF を作成して `/var/spool/lpd/qdir` に配置し、JDF の名前をメッセージ・キューに書き込んで、新規 JDF の存在を **qdaemon** にシグナル通知します (SIGUSR2 の送信によって)。その後 **qdaemon** は、JDF の名前をメッセージ・キューから読み取り、JDF に直接アクセスして、現在スプーラー内にあるジョブすべてを追跡するために維持している内部データ構造に、JDF に含まれているデータを読み込みます。この時点で、ジョブはスプーラーに受け入れられます。

JDF は、キュー状況照会を除くすべてのスプーリング・システム操作に対して作成されます。JDF の構造は、印刷要求、ジョブ取り消し要求、キュー制御要求などの間で異なりますが、いずれにしても JDF は作成されます。**lpstat** と同じ機能を備えたコマンドも、作業を実行するために **enq** を呼び出しますが、JDF は作成されず、**qdaemon** は関与しません。

キューに入れられたジョブが待機しているデバイスが使用可能であると **qdaemon** が判断すると、**qdaemon** はキューのバックエンドを呼び出して、JDF によって指定された引数をバックエンドに渡します。バックエンドはジョブを処理します。

バックエンド処理

キューのバックエンドは **qdaemon** によって開始されます。つまり、**qdaemon** はジョブの処理順序が回ってきたと判断すると、キュー・バックエンドの実行環境をセットアップして、バックエンドの引数ベクトルを作成し、**fork** および **exec** によってバックエンドの実行を開始させます。

バックエンドの同時インスタンス数は、`/etc/qconfig` 構成ファイル内にある、このキューのスタンザに *file* パラメーターが存在するかしないかによって制御されます。*file* パラメーターが存在する場合は、バックエンドのインスタンスはこのキューに対して ただ 1 つ存在できます。これは、ジョブが処理可能であると **qdaemon** が判断した場合に限って、バックエンドの実行環境が設定されるからです。バックエンドの実行環境の設定には、*file* パラメーターによって指定されたファイルまたはデバイスに対してバックエンドの stdout を開く処理が含まれます。 **qdaemon** が前のジョブに対してこのアクションを既に実行していた場合に、そのジョブがまだ実行中ならば、**qdaemon** は *file* パラメーターに指定されたファイルまたはデバイスのロックを取得できないので、そのファイルまたはデバイスに対してバックエンドの stdout を開くことができません。このため、**qdaemon** はジョブをキュー内で保留にして、前のジョブが実行を完了し、ファイルまたはデバイスを解放するのを待ちます。このようにして、スプーリング・システムはデバイスへのシリアル・アクセスを提供し、制御します。

file パラメーターが存在しないか、値が FALSE に設定されている場合、**qdaemon** はバックエンドの stdout を `/dev/null` に対して開き、ジョブを即時に実行します。この状況では、シリアル・アクセスを提

供する必要がある具体的なファイルまたはデバイスは存在しないので、ジョブがキューに積み重ねられることはありません。このキューに対して実行依頼されたジョブは、**qdaemon** が実行環境をセットアップできるようになると即時に実行されます。*file* パラメーターが存在しない場合は、実質的にすべてのローカル・ファイルまたはデバイスへのシリアル・アクセスが使用不可になります。

file パラメーターが存在しないキューの分かりやすい一般的な例は、リモート・プリンター・キューです。この状況では、シリアル・アクセスを提供する対象のリソースは実際には別のホスト上に存在します。ローカル・キューが何らかの制御を行う理由はありません。このタイプのキューに対するバックエンド (デフォルトでは基本オペレーティング・システムの **rembak** プログラム) は、単にジョブをネットワーク経由でリモート・キューに送り、シリアル・アクセス制御の処理はリモート・キューが行うようにします。

基本オペレーティング・システムのローカル印刷キューに対するデフォルト・バックエンドは、**piobe** です。複数のキューがすべて同じバックエンドを指定できます。この状況では、**piobe** の複数のインスタンスが同時に存在する可能性があります。つまり、**piobe** をバックエンドとして指定するそれぞれのキューが、**piobe** のインスタンスを生成する可能性があります。ただし、その上で複数のキューが *file* パラメーターに同じ値を指定している場合は、シリアル・アクセス制限が適用されます。**qdaemon** が **piobe** の別のインスタンスに対して既にロックを獲得している場合、**qdaemon** は、指定されたファイルまたはデバイスに対してロックを獲得できません。この制限のためにジョブを処理できないキューのキュー状況は、「**DEV_BUSY**」と表示されます。*file* パラメーターに指定されたファイルに対して **qdaemon** がロックを獲得すると同時に、状況は「**RUNNING**」に変わります。

一般的な印刷ジョブのデータ・ストリーム・フロー

ジョブの処理がスプーラーに対して実行依頼され、**qdaemon** がジョブを受け入れ、ジョブの処理順序が回ってきたと判断した後、キューのバックエンドが呼び出されます。

次の図に、**piobe** がジョブを処理するために、シェルを使用してフィルター・パイプラインを作成し、管理するプロセスを示します。このフィルター・パイプラインを通るジョブのフローは、次のとおりです。

1. バックエンド (piobe) -- ($_{argc}$ サブルーチンと $_{argv}$ サブルーチンを介して、qdaemon から引数を受け取る)。
2. シェル
3. オプションのフィルター
4. **pioformat**
5. デバイス依存のコード
6. **pioout**
7. デバイス・ドライバー

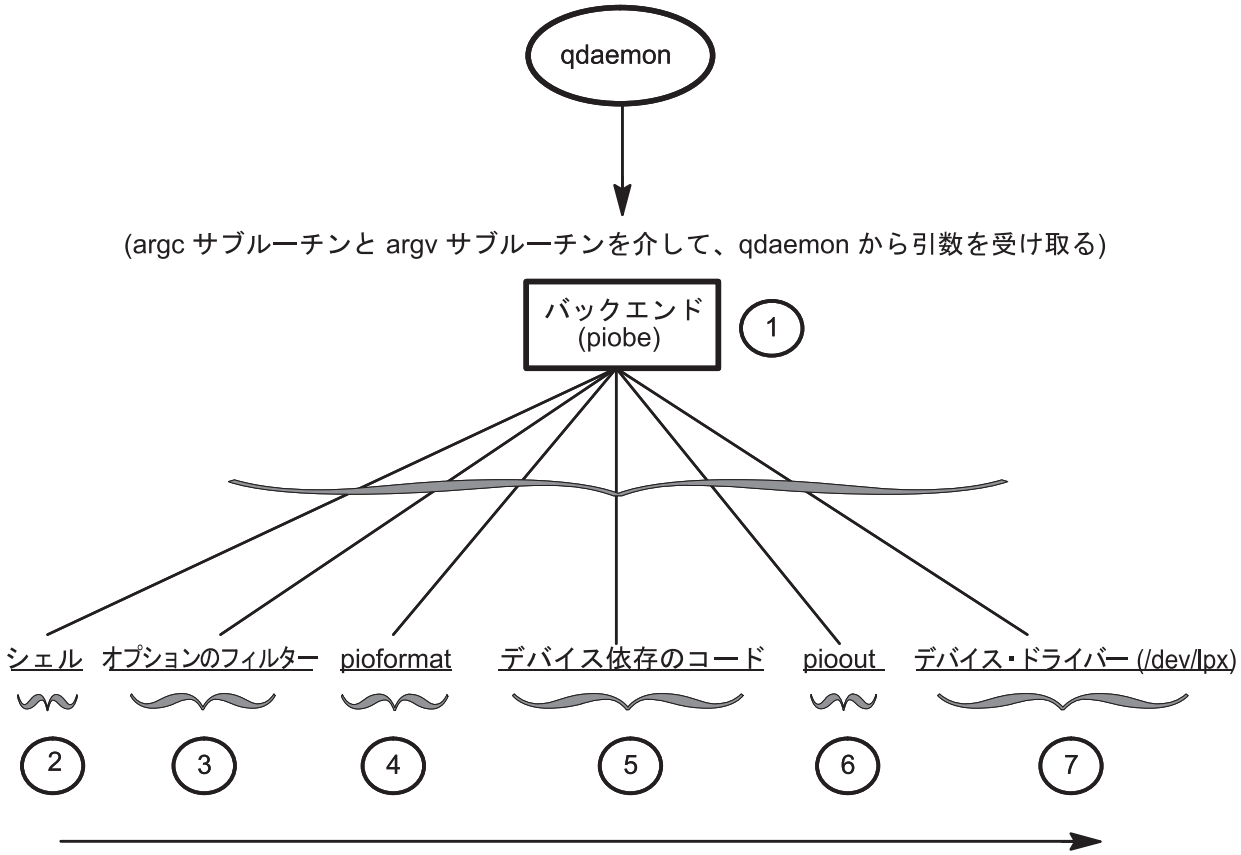

図 2. 一般的な印刷ジョブのデータ・ストリーム・フロー

キューに入れられたジョブが待機しているデバイスが使用可能になると、**qdaemon** はキューのバックエンドを呼び出します。基本オペレーティング・システムの環境では、バックエンドは一般に **piobe** です。**qdaemon** は **piobe** を呼び出し、argc と argv[] を使用して、通常の C プログラミング言語の方式で引数を渡します。

例えば、 54 ページの『スプーラー・データ・フロー (enq コマンド)』のステップ 2 にある次のコマンドを使用する場合を考えます。

```
qprt -Pasc -z1 -fp -p12 -s courier -C -N3 /etc/motd
```

piobe には、次の引数が渡されます。

- argc = 10
- argv[0] = /usr/lib/lpd/piobe
- argv[1] - -f
- argv[2] = p
- argv[3] = -z
- argv[4] = 1
- argv[5] = -p
- argv[6] = 12
- argv[7] = -s
- argv[8] = courier
- argv[9] = /etc/motd

argv[0] は、通常どおりバックエンド自体の名前です。ただし、キュー名を指定する **-Pasc** は、元の引数ベクトルから構文解析によって除去されています。これは、**-C** と **-N3** のフラグと引数についても同様です。

piobe は argv[] の値を使用して、ジョブを要求どおりに処理するために実行する必要があるフィルター・パイプラインを作成します。パイプラインの構造を判別した後、**piobe** はこの構造を実際の処理のためにシェルに渡します。このキューの /etc/qconfig エントリーに **file** パラメーターが存在する場合、**piobe** は **file** パラメーターに指定された値に対して、パイプラインの最後にあるプロセスの stdout を開きます。パイプラインの最後にあるプロセスは、他のファイルまたはデバイスに対して stdout を再度開くことができます。

これらのプロセス間には、図に明示されていない次のような親子関係があることに注意してください。

- **qdaemon** は **piobe** の親です。
- **piobe** はシェルの親です。
- シェルは **pioout** の親です。これは、デバイス・ドライバーにアクセスする前のパイプラインの最後にあるプロセスです。**pioout** は、*デバイス・ドライバーと組み合わせて使用されるインターフェース・プログラム*、または*デバイス・ドライバー・インターフェース・プログラム* と呼ばれます。
- **pioout** は、**pioformat** の親です。
- **pioformat** は、デバイス依存のコードを実行時に動的にロードし、リンクします。このため、デバイス依存のコードは、オペレーティング・システムのプロセス・テーブルにプロセスとして表示されません。
- **pioformat** は、**pr** フィルターなど、オプションのフィルター (存在する場合) の親です。

pr などのオプション・フィルターをコマンド・ラインに指定して (またはコロン・ファイルにハードコーディングして)、**pioformat** による処理の前にジョブの事前フィルターを実行できます。

pioformat は、デバイス独立のフォーマッター・ドライバーと呼ばれます。さまざまなデバイス依存のフォーマッターを動的にロード、リンク、および駆動して、特定のデータ・ストリーム・タイプ (例えば、PostScript、ASCII、GL、PCL) のジョブを処理します。

デバイス依存のコードは、特定のデータ・ストリームと物理プリンターの組み合わせに固有の属性を処理するために設計されたものです。データ・ストリーム・タイプとプリンターの組み合わせは、共通の属性をもつクラスにグループ化できるので、現時点で存在するデバイス依存モジュールは 20 種類より少ない数です。これらのモジュールは、実行時に **pioformat** によってロード、リンク、および駆動されます。

pioout は、ジョブ処理パイプラインの最後にあり、*デバイス・ドライバー・インターフェース・プログラム* と呼ばれます。**pioout** の機能は、処理済みデータ・ストリームを取り出し、その対象のデバイス (通常はプリンター) に配送することです。標準的なローカル印刷キュー環境では、/etc/qconfig 内の **file** パラメーターに指定された、/dev ディレクトリーのキャラクター・スペシャル・ファイルに対して、**pioout** の stdout が開かれます。

これは、プリンター・ハードウェアにデバイス・ドライバーへのアクセスを提供する、/dev ディレクトリー内のキャラクター・スペシャル・ファイルです。

仮想プリンターとフォーマッター・フィルター

スプーラー・キュー・バックエンドが **piobe** である場合、通常はフォーマッター・フィルター が、印刷ジョブを処理するフィルター・パイプラインの最後から 2 番目にあるプロセスです。フォーマッター・フィルターは、2 つのコードで構成されます。

フォーマッター・フィルターは、入力パラメーターに基づいて、入力印刷ファイルのフォーマットを設定するか、または変更せずに引き渡す機能を備えています。フォーマッターが入力ファイルを変更せずに引き渡す場合でも、入力ファイルの印刷前にプリンターを初期化するプリンター・コマンドを送信し、印刷の完了後にはプリンターを復元します。

次の図に示すように、フォーマッター・フィルターは次のコンポーネントから構成されます。

- デバイス独立のフォーマッター・ドライバー
- デバイス依存のフォーマッター

1 つ目は、デバイス独立のフォーマッター・ドライバー **pioformat** です。 2 つ目はデバイス依存のフォーマッターで、存在するものは 20 種類に満たない数です。コードの実行が特定のハードウェア、例えば特定の物理プリンターにまったく依存しない場合、そのコードはデバイス独立です。同様に、コードの実行が特定のハードウェア、例えば特定の物理プリンターに依存する場合、そのコードはデバイス依存です。基本オペレーティング・システム・スプーラーのフォーマッター・フィルター内では、特定の物理プリンターまたはプリンター・クラスの属性 (例えば、そのプリンターまたはプリンター・クラスに固有のサポートされるデータ・ストリーム、エスケープ・シーケンス、制御コードなど) をすべて処理するように設計されたコードが、デバイス依存のフォーマッターに含まれています。

デバイス独立の **pioformat** はフォーマッター・ドライバー と呼ばれ、この名前は行う処理の内容を正確に表しています。**pioformat** は、実行開始時にいくつかの引数を予期します。これらの引数の 1 つは、デバイス依存のフォーマッターの絶対パス名です。実行時に、**pioformat** はデバイス依存のフォーマッターを動的にロード、リンク、および駆動します。次の図に、この関係を示します。

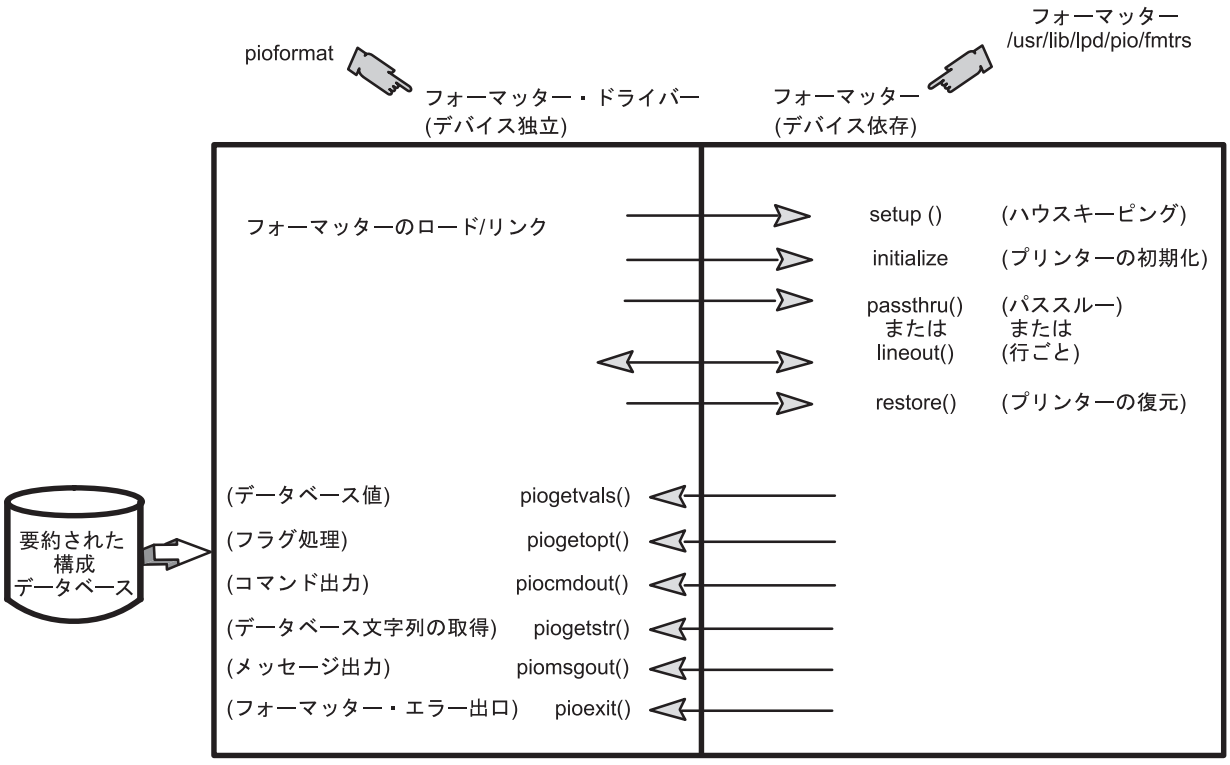

図 3. フォーマッター・フィルター

必要に応じて、**pioformat** コマンドから 5 つのサブルーチンを呼び出すことができるようにする必要があります。**pioformat** 自体はこれらのサブルーチンを含んでいません。これらのサブルーチンはデバイス依存のフォーマッター内にあり、**pioformat** がデバイス依存のフォーマッターをロードおよびリンクしているとき、実行時に **pioformat** に提供されます。

フォーマッター・ドライバーは、パイプラインによって呼び出され、駆動するフォーマッターの名前を渡されます。フォーマッター・ドライバーは、フォーマッターを動的にロードしてリンクし、フォーマッターの **setup** 機能を呼び出します。この機能は、データ・フォーマットまたはデータ・パススルーのどちらが要求されているかを指示します。フォーマッターの **setup** 機能が必要な機能を実行した後、処理はフォーマッター・ドライバーに戻ります。フォーマッター・ドライバーは、**initialize** 機能を呼び出します。**initialize** 機能は、一連のプリンター・コマンドを出力してプリンターを初期化し、フォーマッター・ドライバーに戻ります。

フォーマッター・ドライバーは、**passthru** 機能を 1 回呼び出すか、印刷ファイルの各行ごとに **lineout** 機能を呼び出します。これは、**setup** 機能からの戻りコードによって決まります。 **lineout** 機能が呼び出されると、フォーマッター・ドライバーはすべての垂直スペーシングを自動的に行います (例えば、行送り、垂直タブ、用紙送り、上部マージン、下部マージン)。行送りと垂直タブは、**lineout** 機能によって実行されます。その他の垂直スペーシング機能は、自動的に実行されます。

処理が完了すると、フォーマッター・ドライバーは **restore** 機能を呼び出します。**restore** 機能は、プリンターをデフォルト状態に復元する一連のプリンター・コマンドを出力します。このデフォルト状態は、データベース属性値によって定義されます。

印刷フォーマッターがプリンター・フォーマッターのサブルーチンと相互作用する方法について詳しくは、84 ページの『印刷フォーマッターの例』を参照してください。

/etc/qconfig スプーラー構成ファイル

`/etc/qconfig` ファイルは、基本オペレーティング・システムに対して定義されているキューをすべて記述します。

キューは、名前と番号が付いた、特定のデバイスに対する要求のリストです。デバイスは、これらの要求を一度に 1 つずつ処理できるもの (ハードウェアまたはソフトウェア) です。キューは、デバイスへのシリアル・アクセスを提供します。それぞれのキューは、少なくとも 1 つのデバイスによって処理される必要があり、複数のデバイスによって処理される場合もよくあります。

/etc/qconfig ファイルの構造

`/etc/qconfig` ファイルは、スプーラー・ドメインに関連する最も重要なファイルです。

- `/etc/qconfig` ファイルには、スプーラーに認識されているすべてのキューの定義が含まれています。
- システム管理者は、`/etc/qconfig` ファイルの内容を読めば、それぞれのキューの機能を見分けることができます。
- 推奨されませんが、`/etc/qconfig` ファイルを編集すれば (148 ページの『/etc/qconfig ファイル』を参照)、スプーラーを停止せずにスプーラー・キューを変更できます。

qdaemon は、`/etc/qconfig` の ASCII バージョンを読み取り、バイナリー・バージョン `/etc/qconfig.bin` を作成します。`/etc/qconfig` が **qdaemon** を構文解析するには、このファイルが特定の構造化フォーマットに準拠している必要があります。次の `/etc/qconfig` *ファイル構造* の例に、このフォーマットの詳細を示します。

```
                ローカル・キュー
queue_name:
        device = device_name
        up = TRUE or FALSE
        discipline = fcfs or sjn
device_name:
        file = physical_device_name or FALSE
        header = always or group or never
        trailer = always or group or never
        access = both or write
        backend = full_path_name_to_backend_program
                リモート・キュー
queue_name:
        device = device_name
        up = TRUE or FALSE
        host = remote_hostname
        s_statfilter = full_path_to_short_filter
        l_statfilter = full_path_to_long_filter
        rq = remote_queue_name
device_name:
        backend = full_path_name_to_backend_program
```

/etc/qconfig は、スタンザと呼ばれるテキスト・ブロックで構成されます。それぞれのキューは、1 対のスタンザによって表現されます。対の 1 つ目のスタンザはキュー・スタンザと呼ばれ、対の 2 つ目のスタンザはデバイス・スタンザと呼ばれます。スタンザは、キューの属性と機能を記述するパラメーターとパラメーター値で構成されます。

qdaemon が /etc/qconfig の ASCII バージョンを解析する際に、最初に識別されるコメント化されていない行は、1 つのワードとそれに続くコロンで構成されている必要があります。この行は、キュー・スタンザの開始を示します。このワードは、ユーザーがジョブを実行依頼できるキューの名前です。この行の後に、タブによってインデントされた 1 つ以上の行が必要です。これらの行の 1 つは、device = *device_name* にする必要があります。**device** パラメーターの値は、キュー・スタンザからデバイス・スタンザへのリンクです。このパラメーターに、他の機能はありません。キューが最初にセットアップされるとき、オペレーティング・システムは **device** パラメーターの値として、プリンターの名前 (**lp1** など) を頻繁に使用します。キューが実際に **lp1** を使用するようにセットアップされていることもありますが、**device** パラメーターの値として **lp1** を使用することの意味は、デバイス・スタンザの名前が **lp1** になることのみです。オペレーティング・システムに **lp1** として認識されている実プリンターが存在するかどうかは関係ありません。

タブによってインデントされた行の後、qdaemon は **device** パラメーターの値であるワードと、その後に続くコロンを検出する必要があります。この行は、デバイス・スタンザの開始を示します。このワードは、対応するキュー・スタンザがシリアル・アクセスを提供する対象のデバイスの名前です (ユーザーが意識する必要は通常ありません)。この行の後に、タブによってインデントされた 1 つ以上の行が必要です。これらの行の 1 つは、backend = *full_path_name_to_backend* にする必要があります。ローカル・スプーリング環境では、このスタンザには重要な意味を持つ 2 つのパラメーターがあります。

file パラメーターは、キューがシリアル・アクセスを提供する対象の実デバイスを指定します。スプーリング・システムに対して実行依頼されるジョブは、このデバイスへのキューに入れられることに注意する必要があります。**lp1** としてオペレーティング・システムに認識されているプリンターを使用するようにキューがセットアップされている場合、**file** パラメーターの値は /dev/lp1 になります。キューを作成するオペレーティング・システム・ルーチンは、デフォルトでデバイス・スタンザの名前として実デバイスの名前を使用します。この理由から、**device** パラメーターの意味に紛らわしさが生じることがあります。

backend パラメーターは、ジョブの処理順序が回ってきたことを **qdaemon** が判断した後、スプーリング・サブシステムに対して実行依頼されたジョブを処理するプログラムの絶対パスを指定します。

スプーラー・キュー、仮想プリンター、および物理プリンター

キュー、仮想プリンター、および物理プリンターを定義するための /etc/qconfig ファイル構造の例を示します。

この 4 つのキュー - 4 つの仮想プリンター - 1 つの物理プリンター の例では、4 つのキューを 1 つの物理プリンター (この例では /dev/lp1) に対して定義する、/etc/qconfig のインスタンスを示します。 4 対のスタンザはすべて、文字列 **lp1** を使用してキュー・スタンザをデバイス・スタンザに接続している点に注意してください。それぞれのデバイス・スタンザ内の **file** パラメーターは、基本オペレーティング・システムに **lp1** として認識されている (そのデバイス・ドライバーの入り口点は /dev/lp1 である) プリンターが、これらのキューのいずれかに対して実行依頼されるジョブの実際の物理出力先であることを指定します。これらのキューを SMIT によって定義する場合、キュー定義を実際に作成するコマンドには、それぞれのスタンザの対を一方から他方に接続する文字列が必要です。この例の物理プリンターは **lp1** なので、それぞれのキュー・スタンザ内の **device** パラメーター値としても、それぞれのデバイス・スタンザの名前としても、文字列 **lp1** が使用されています。次の /etc/qconfig *ファイル構造* の例に、このフォーマットの詳細を示します。

```
asc:
        device = lp1
lp1:
        file = /dev/lp1
        header = never
        trailer = never
        access = both
        backend = /usr/lib/lpd/piobe

gl:
        device = lp1
lp1:
        file = /dev/lp1
        header = never
        trailer = never
        access = both
        backend = /usr/lib/lpd/piobe

pcl:
        device = lp1
lp1:
        file = /dev/lp1
        header = never
        trailer = never
        access = both
        backend = /usr/lib/lpd/piobe

ps:
        device = lp1
lp1:
        file = /dev/lp1
        header = never
        trailer = never
        access = both
        backend = /usr/lib/lpd/piobe
```

これらのスタンザの対はそれぞれ、キューを定義します。キューのバックエンドが **piobe** ならば、それぞれのキューには仮想プリンターも関連しています。仮想プリンター定義を手動で作成することも可能ですが、仮想プリンター定義は通常は SMIT と **piomkpq** コマンドを使用してキュー定義と同時に作成されます。仮想プリンター定義は、/etc/qconfig には含まれていません。キューのスプーラー・バックエンドが **piobe** であることから、この定義が存在することは分かりますが、基本オペレーティング・システムのファイルシステム内で他の場所に格納されています。仮想プリンター定義の識別とアクセスには、キューの名前が使用されます。

基本オペレーティング・システムに **lp1** として認識されている物理プリンターは、具体的には少なくとも 4 つの異なるデータ・ストリーム・タイプをサポートします。これらのタイプは、ASCII (**asc**)、プロッター・エミュレーション (**gl**)、プリンター・コマンド言語 (**pcl**)、および PostScript (**ps**) です。仮想プリンター定義に関連したキューはそれぞれ、特定のデータ・ストリーム・タイプを処理することを目的としています。このため、4 つのキューがあります。これは、物理プリンターと仮想プリンターを論理的に分離するという、基本オペレーティング・システムの概念に基づいています。

スプーラー・キュー名と状況のフォーマット

スプーラー・キュー名 (キュー・スタンザの名前) は、長さが 7 文字を超えることがありますが、キュー状況照会の出力には先頭 7 文字のみが表示されます。デバイス名 (デバイス・スタンザの名前) は、キュー状況照会の出力では 5 文字に制限されます。

スプーラー・キュー状況照会の中で、リモート・スプーラー・キューは 2 回示されます (ローカル・キューについて 1 回、印刷サーバー上のリモート・キューについて 1 回)。例えば、`/etc/qconfig` に次のエントリーがあるとします。

```
myps:
        device = @kricket
        up = TRUE
        host = kricket
        s_statfilter = /usr/lib/lpd/aixshort
        l_statfilter = /usr/lib/lpd/aixlong
        rq = myps
@kricket:
        backend = /usr/lib/lpd/rembak
```

コマンド **lpstat -p***myps* は次の結果を戻します。

```
Queue   Dev   Status    Job Files    User      PP %  Blks Cp  Rnk
------- ---   --------- --- -------- --------- -- -  ---- --  ---
myps    @krik READY
myps    myps  READY
```

出力の 1 行目は、**myps** という名前のローカル・スプーラー・キューに、名前が **@krik** としてリストされているデバイス・スタンザがあり、キューの状況は **READY** であることを示しています。 2 行目は、ターゲット・リモート・スプーラー・キュー (この名前も **myps**) に、**myps** としてリストされたデバイス・スタンザがあり、このキューの状況も **READY** であることを示しています。 (この作成者は、ローカル・スプーラー・キュー名を印刷サーバーのスプーラー・キュー名と同じにすることを習慣にしています。こうすれば、スプーラー・キュー状況照会の出力内で 2 つの行を視覚的に分類しやすくなります。)

プリンター・バックエンドのプログラミング

プリンター・バックエンドは、基本オペレーティング・システムの標準機能です。

基本オペレーティング・システムのプリンター・バックエンドは、スプーラー (通常は **qdaemon** コマンド) から印刷要求を受け取って処理します。プリンター・バックエンドは、**piobe** コマンドによってインプリメントされます。

プリンター・バックエンドは、オブジェクト・データ・マネージャー (ODM) 定義済みデータベースにインストールされたプリンターをすべてサポートします。印刷サブシステムの管理を支援するために、プリンター・バックエンドをカスタマイズできます。詳しくは、14 ページの『印刷の管理』を参照してください。また、プリンター・バックエンドを変更して、サポートされないプリンターと各国語サポート (NLS) コード・ページ変換テーブルを追加することもできます。

プリンターをプリンター・バックエンドに追加するには、そのプリンター用のプリンター・コロン・ファイルを追加する必要があります。多くの場合、類似のプリンターのプリンター・コロン・ファイルを複写して、少し変更を加えて使用できます。既存のプリンター・コロン・ファイルを変更するだけでは不十分ならば、印刷フォーマッターを作成できます。変更内容が印刷フォーマッターの範囲を超える場合は、プリンター・バックエンドを新規に作成する必要があります。

詳しくは、次に示すセクションを参照してください。

- 107 ページの『プリンター・コロン・ファイルを使用したプリンターの追加』は、プリンター・コロン・ファイルを複写する手順を説明しています。
- 74 ページの『プリンター・コロン・ファイルのエスケープ・シーケンス』は、プリンター・コロン・ファイルを変更する際に役立つ情報を説明しています。
- 92 ページの『libqb 内のバックエンド・ルーチン』と 86 ページの『バックエンドと qdaemon の相互作用』は、プリンター・バックエンドを新規に作成する際に役立ちます。

印刷ファイル内の NLS コード・ポイントをプリンター用のコード・ポイントに変換する手順は、コード・セットが 1 バイトかマルチバイトかによって異なります。詳しくは、次の項を参照してください。

- 93 ページの『プリンター・コード・ページ変換テーブル』
- 94 ページの『マルチバイト・コード・セットの場合のプリンター・コード・ページ変換』

サード・パーティー・ベンダーは、特殊用途のためにプリンター・バックエンドをカスタマイズできます。

プリンター・バックエンドのデータ・フロー

バックエンドの主な目的は、文字をデバイス (通常はプリンター) に送ることです。

プリンター・バックエンドは、印刷されるファイルまたはファイル・グループごとに 1 回呼び出され、それぞれのファイルの名前がパラメーターとしてバックエンドに渡されます。バックエンドはファイルを開き、読み取って、デバイスに送ります。バックエンドを動作させる方式としては、書き込み先を標準出力にして、**qdaemon** プロセスが適切なファイル・ディスクリプターに対してデバイスを開くようにすることをお勧めします。このためには、`qconfig` ファイルの **file** フィールドを設定する必要があります。

印刷するファイルの名前は、直接パス名または相対パス名のどちらにすることもできます。バックエンドのユーザー ID とグループ ID は、**enq** コマンドを呼び出したプロセスのものです。

バックエンドを呼び出すと、バックエンドはユーザーの環境にアクセスします。ユーザーの環境にアクセスするために、バックエンドは **getenv** サブルーチンを呼び出す場合があります (詳しくは、**getenv** サブルーチンを参照してください)。例えば、ユーザーのディレクトリーにアクセスするために、**getenv(PWD)** はディレクトリー名のポインターを戻します。バックエンドはこのポインターを使用して、このディレクトリーに対して読み書きを行うことができます。

バックエンドが標準出力に書き込むと、**qdaemon** はデバイスを root ユーザー・モードで開きます。バックエンド自体がデバイスを開く必要がある場合は、デバイスを開くための適切な許可がバックエンドに必要です。バックエンドは印刷ジョブを送るユーザーの許可のもとで実行されるので、デバイスの保護を変更するか、バックエンド set-user-ID または set-group-ID をインストールする必要があります。

デフォルトでは、**stdin**、**stdout**、および **stderr** はすべてヌル・デバイス (`/dev/null`) に対して開かれますが、`qconfig` ファイルの **file** フィールドと **access** フィールドを使用して、**stdout** (および場合によっては **stdin**) の設定値を指定変更できます。

仮想プリンターの定義と属性

仮想プリンター定義 は、特定のプリンターの属性または特性と、特定のデータ・ストリーム・タイプの属性を対応付けるファイルです。

特定のプリンターが複数のデータ・ストリーム・タイプをサポートする場合は、仮想プリンター定義を作成して、プリンターの属性とそれぞれのデータ・ストリーム・タイプを対応付ける必要があります。このため、プリンターが ASCII と PostScript の両方のデータ・ストリームをサポートする場合は、そのプリンターに対して 2 つの仮想プリンター定義を作成する必要があります。

コロン・ファイルは、プリンターまたはプロッターの仮想プリンター定義を格納します。コロン・ファイルは、定義済みデータベース・ディレクトリー、およびカスタマイズされたデータベース・ディレクトリーにあります。プリンター・バックエンドは、コロン・ファイルに格納された属性値を使用して、印刷要求のフォーマットを設定します。

属性値はすべて、値が文字列、整数、またはブール値のどれを表している場合にも、コロン・ファイル内では文字列として存在します。属性値には、他の属性値への組み込み参照、または値の内容を動的に決定する組み込みロジックを含めることができます。

コロン・ファイルの詳細、および属性文字列の中で組み込み参照およびロジックを使用する方法については、79 ページの『プリンター・コロン・ファイルの規則』、および 74 ページの『プリンター・コロン・ファイルのエスケープ・シーケンス』を参照してください。

仮想プリンターの属性

仮想プリンターの作成に使用されるコマンド (**mkvirprt** コマンドまたは **smit virprt** コマンド) は、定義済みの仮想プリンター定義をコピーし、指定されたキューとキュー・デバイス用にカスタマイズされた仮想プリンター定義を作成します。

カスタム定義の属性値は、**chvirprt** コマンドまたは **smit lsvirprt** コマンドを使用してさらに変更できます。

特定のプリンター・デバイスによってサポートされるそれぞれのデータ・ストリーム・タイプごとに、仮想プリンターを作成する必要があります。サポートされるデータ・ストリーム・タイプには、次のようなものがあります。

データ・ストリーム・タイプ	属性名/値のコード	説明
asc	a	拡張 ASCII
pcl	c	Hewlett-Packard PCL
630	d	Diablo 630
gl	g	Hewlett-Packard GL
	p	パススルー (変更せずにプリンターに送られる)
ps	s	PostScript
855	a	Texas Instruments 855
kji	k	漢字

mkvirprt コマンドまたは **smit virprt** コマンドを使用して仮想プリンターを作成する場合は、定義済みプリンターのリストから目的のプリンターを選択するようにプロンプトが出されます。新規プリンター用にプリンター・ポートを新しく構成した場合は、その新規プリンター・ポートを選択します。仮想プリンター・コマンドが実行されると、システムは印刷キューを作成し、定義済みデータベース・ディレクトリー

/usr/lib/lpd/pio/predef/* 内にある選択されたプリンターのコロン・ファイルを、カスタマイズされたデータベース・ディレクトリー /var/spool/lpd/pio/custom/* にコピーします。

注: フラグが指定されていない場合、**mkvirprt** コマンドは対話式になります。

仮想プリンター定義に格納されている属性値を変更したり、さらにカスタマイズしたりするには、**chvirpt** コマンドまたは **smit lsvirprt** コマンドを使用します。**smit lsvirprt** を使用して属性値を変更するには、= (等号) 記号の前後にスペースを入れずに、*attribute_name=attribute_value* と入力します。

仮想プリンター定義にあるそれぞれの属性名は、固有であることが必要です。属性名には、文字 **a** から **z**、**A** から **Z**、**0** から **9**、および _ (下線) を含めることができます。属性名の先頭を数字にすることはできません。すべての属性名の長さは 2 文字にする必要がありますが、例外としてグループ・ヘッダー属性名の長さは 5 文字にすることができます。

グループ・ヘッダーの属性名は、先頭を _ _ (2 つの下線) にする必要があり、5 文字より長くすることはできません。グループ・ヘッダー属性は、関連した属性のグループの先頭を示します。

サポートされる PostScript レーザー・プリンター (4029 ページ印刷装置) の代表的な属性の例を示します。それぞれの例には、**lsvirprt** コマンドと **smit lsvirprt** コマンドが仮想プリンター属性をどのように表示するかを示し (それぞれの属性のディスクリプターとともに)、これらと同じ属性がプリンター・コロン・ファイルにどのように格納されているかを示します。

デフォルト仮想プリンター・フラグ値の属性

デフォルト・フラグ値の属性は、_ **_FLG** グループ・ヘッダー属性の下にグループ化されます。

属性に対応するフラグを印刷コマンドに使用すると、これらの属性の値がコマンド・ラインからのものに指定変更されます。例えば、仮想プリンター定義の **_l** 属性には、ページに印刷する行数の値が含まれています。**_l** 属性に格納されているデフォルト値が 66 であるとします。次の印刷要求は、**-l** フラグを使用してページあたりの行数を指定していません。

```
qprt -P Pro myfile
```

プリンター・サブシステムは、**_l** のデフォルト値 66 を使用して印刷要求を処理します。次の印刷要求は、**-l** フラグを使用して、ページあたりのテキスト行数を 50 に指定しています。

```
qprt -l 50 -P Pro myfile
```

この **-l** フラグ値が、Pro プリンターに対する仮想プリンター定義の **_l** 属性のデフォルト値を指定変更します。

デフォルト・フラグ値属性の先頭文字は、常に _ (下線) です。2 文字目は、デフォルト値を格納する対象のコマンド・フラグに対応します。

次の例では、_ **_FLG** グループ・ヘッダーの下にある属性値の一部を示しています。これらは、サポートされる PostScript レーザー・プリンターに対する代表的な値です。

```
Name  Description                                     Value
_ _FLG VALUES THAT MAY BE OVERRIDDEN WITH FLAGS
        ON THE COMMAND LINE
_1    Page Headings Wanted For Text Converted         !
        to PostScript? (!: no;  +: yes)
_2    Use Two Columns for Text Converted to           !
        PostScript? (!: no;  +: yes)
_3    Gaudy Mode Wanted for Text Converted to         !
        PostScript? (!: no;  +: yes)
_4    Print Garbage File Anyway for Text              !
```

```
       Converted to PostScript? (!: no;  +: yes)
_5    List Characters Not In Font When Converting      !
       Text to PostScript? (!: no;  +: yes)
_6    Font Name for Header Line of Text Converted      300
       to PostScript
_A    stderr returned? 0:no; 1:yes, & pipelines;   1
       2:yes, & values, pipelines
_H    Name To Replace Host Name On Burst Page
_J    Restore the Printer at the End of the ?          +
       Print Job (!: no;  +: yes)
_L     Wrap Long Lines (!: no;  +: yes)                +
```

前に示した属性は、コロン・ファイルには次のように格納されています。

```
:056:_ _FLG::
:466:_1::!
:467:_2::!
:469:_3::!
:470:_4::!
:471:_5::!
:472:_6::300
:013:_A::1
:022:_H::
:027:_J::+
:030:_L::+
```

仮想プリンターのシステム管理属性

_ _SYS グループ・ヘッダー属性は、**sh**、**si**、**st** などの属性の値を格納します。**sh** 属性と **st** 属性は、ヘッダー・ページとトレーラー・ページのパイプラインを格納しています。

si 属性は、プリンターが注意を必要とするときに、プリンター介入メッセージを受け取るユーザーを指定します。ヌル・ストリングは、印刷ジョブを実行依頼したユーザーに介入メッセージを送るように指定します。ユーザー名はコンマで区切ります。必要に応じて属性を変更するには、SMIT の「Virtual Printers (仮想プリンター)」オプション、または **chvirprt** コマンドを使用します。

例えば、**si=** は印刷ジョブを実行依頼したユーザーがメッセージを受け取るように指定し、**si=mary** はユーザー **mary** がメッセージを受け取るように指定し、**si=,jim@server02** は印刷ジョブを実行依頼したユーザーと、ノード **server02** のユーザー **jim** が両方とも介入メッセージを受け取るように指定します。

システム管理属性の先頭文字は、**s** です。

サポートされる PostScript レーザー・プリンターの代表的な **_ _SYS** 属性のいくつかを、次に示します。

```
_ _SYS OTHER VALUES OF INTEREST TO THE SYSTEM
        ADMINISTRATOR
sh     Pipeline for Header Page                      %Ide/pioburst
                                                     %F[H] %Idb/H.p
                                                     s | %Ide/piofo
                                                     rmat -@%Idd/%I
                                                     mm -!%Idf/piof
                                                     pt%f[j]
si     Users, Separated by Commas, to Get Intervention
         Messages; Null String Is Job Submitter
sp     Command Line Flags Prohibited For All -d values;
         Ignored: cmnrBDMPRT
st     Pipeline for Trailer Page                     %Ide/pioburst
                                                     %F[H] %Idb/T.p
                                                     s | %Ide/piofo
                                                     rmat -@%Idd/%I
```

```
                                                    mm -!%Idf/piof
                                                    pt%f[j]
sw     Width of Attribute Value Area On Header Page 78
         (0 means ignore width)
```

これらと同じ属性値が、プリンター・コロン・ファイルには次のように格納されています。

```
:060:_ _SYS::
:321:sh::%Ide/pioburst %F[H] %Idb/H.ps | %Ide/pioformat -@%Idd/%Imm
-!%Idf/piofpt %f[j]
:322:si::
:323:sp::
:324:st::%Ide/pioburst %F[H] %Idb/T.ps | %Ide/pioformat -@%Idd/%Imm
-!%Idf/piofpt %f[j]
:325:sw::78
```

仮想プリンターの入力データ・ストリーム属性

_ **_IDS** グループ・ヘッダー属性は、さまざまな入力データ・ストリームのパイプラインを格納する属性リストの先頭にあります。

このグループの属性には、拡張 ASCII 入力データ・ストリームのパイプラインを格納する **ia** 属性と、PostScript 入力データ・ストリームのパイプラインを格納する **is** 属性があります。また、**ip** 属性もこのグループの代表的な属性です。**ip** (またはパススルー) 属性は、フォーマッター・フィルターからプリンターに出力を変更せずに渡します。

入力データ・ストリーム属性の先頭文字は、**i** です。2 文字目は、入力データ・ストリームのタイプを指定します。

次に示す _ **_IDS** 属性の例は、サポートされる PostScript レーザー・プリンター (4029 ページ印刷装置) の代表的な入力データ・ストリームのパイプラインを示しています。

```
_ _IDS PIPELINES FOR INPUT DATA STREAMS (2 char, 1st="i",
         2nd=data stream name)
ia     Pipeline for Input Data Stream "a" (extended    /usr/bin/enscr
         ASCII)                                        ipt -p- -q%?%G
                                                       _2%t -2%;%?%G_
                                                       z%t -r%;%?%G_3
                                                       %t -G%;%?%G_l%
                                                       t%e -B%;%?%G_L
                                                       %t%e -c%;%?%Ch
                                                       %t%fbh%e%?%L_h
                                                       %t -b'%I_h'%;%
                                                       ; -L%G_l%d -f%
                                                       ?%Cs%t%f!s%e%I
                                                       _s%;%G_p%d %?%
                                                       G_l%t-F%Iw7%G_
                                                       p%d%;%?%G_4%t
                                                       -g%;%?%G_5%t -
                                                       o%;%?%L_f%t%e
                                                       %I@1%; | %Iis
il     Command Line Flags Prohibited For Input Data    /interleaf/ile
         Stream; Ignored: cmnrBDMPRT                   af5/bin/pl2ps
                                                       -ppd IBM17521.
                                                       PPD -r 1270-nf
                                                       -np | %Ide/pio
                                                       format -@%Idd/
                                                       %Imm-!%Idf/pio
                                                       fpt %f[juJZ]
in     Pipeline for Input Data Stream "n" (troff       /usr/bin/psc |
         (ditroff) intermediate output)                s%Ii
ip     Pipeline for Input Data Stream "p"              %Iis
         (pass-through)
is     Pipeline for Input Data Stream "s" (PostScript) %Ide/pioformat
```

```
                                                -@%Idd/%Imm -!
                                                 %Idf/piofpt %U
                                                 H %f[juJZ]
```

コロン・ファイルには、これらと同じ属性が次のフォーマットで格納されています。

```
:057:_ _IDS::
:274:ia::/usr/bin/enscript -p- -q%?%G_2%t -2%;%?%G_z%t -r%;%?%G_3%t
-G%;%?%G_1%t%e -B%;%?%G_L%t%e -c%;%?%Ch%t%fbh%e%?%L_h%t -b'%I_h'%;%;
-L%G_l%d -f%?%Cs%t%f!s%e%I_s%;%G_p%d %?%G_1%t-F%Iw7%G_p%d%;%?%G_4%t
-g%;%?%G_5%t -o%;%?%L_f%t%e %I@1%; | %Iis
:001:il::/interleaf/ileaf5/bin/pl2ps -ppd IBM17521.PPD -r 1270 -nf -
np | %Ide/pioformat -@%Idd/%Imm -!%Idf/piofpt %f[juJZ]
:465:in::/usr/bin/psc | %Iis
:277:ip::%Iis
:273:is::%Ide/pioformat -@%Idd/%Imm -!%Idf/piofpt %UH %f[juJZ]
```

仮想プリンターの禁止フラグの属性

_ _PFL グループ・ヘッダー属性の下にグループ化される属性は、特定のデータ・ストリーム用のプリンター・バックエンドによってリジェクトされるコマンド・フラグの名前を格納します。

フロントエンド印刷要求 (**qprt** など) に禁止コマンド・フラグを使用した場合、フラグはリジェクトされ、そのフラグがシステム管理によって禁止されているというメッセージが返されます。禁止フラグ属性名の先頭文字は **I** で、2 文字目はリジェクトされるデータ・ストリーム・タイプを示します。

1 つのデータ・ストリームに対して複数のフラグを禁止にするには、スペース、コンマなどの区切り文字を入れずに 1 文字のフラグ名を並べて格納します。例えば、拡張 ASCII 入力データ・ストリームに対する **-e** (強調印刷) フラグと **-E** (縦倍角印刷) フラグをリジェクトするには、**smit lsvirprt** コマンドを実行し、次のように入力してこの属性を設定します。

```
Ia=eE
```

次の例は、サポートされる PostScript プリンター上の PostScript データ・ストリームに禁止フラグ属性を設定する **Is** 属性を示しています。 **Is** 属性のディスクリプターは、文字列 **Ignored: cmnrBDMPRT** を含んでいます。この文字列は、個々の文字 **cmnrBDMPRT** によって表されるフラグを無視するようにバックエンドに指示します。これらのフラグは、バックエンドではなくスプーリング・サブシステムに向けたフラグです。このため、これらのフラグのいずれかを禁止フラグとしてリストしても、バックエンドに影響はなく、フラグは禁止されません。

```
_ _PFL FLAGS PROHIBITED FOR INPUT DATA STREAMS (2
         char,1st="I",2nd=data str name)
Is     Command Line Flags Prohibited For Input Data
         Stream; Ignored: cmnrBDMPRT
```

コロン・ファイルには、前記の例の属性が次のように格納されています。

```
:059:_ _PFL::
:001:Is::
```

仮想プリンターのフィルター・フラグ属性

_ _FIL グループ・ヘッダー属性の下にグループ化される属性は、テキスト・フィルター・フラグのコマンド・ストリングを格納します。

属性名の先頭文字は常に **f** で、2 文字目はフィルターのタイプを示します。**-p** や **-n** などのフィルター・フラグは、印刷ジョブのフォーマット設定に使用するフィルターのタイプをバックエンド・プログラムに対して指定します。フィルター属性は次のように指定します。

項目	説明
fp	**pr** フィルター
fn	*ditroff* (デバイス独立の *troff*) データを含むファイルのフォーマットを設定します。
fl	制御文字を印刷し、改ページを抑止します。
ft	**troff** コマンドによって作成されたデータを含むファイルのフォーマットを設定します。
fd	DVI フィルターが、**tex** によって作成されたファイルのフォーマットを設定します。
fg	標準プロット・データ・ファイル (**plot** によって作成されたファイル) のフォーマットを設定します。
fv	ラスター・イメージ・ファイルのフォーマットを設定します。
fc	**cifplot** によって作成されたデータを含むファイルのフォーマットを設定します。
ff	各行の先頭文字を FORTRAN 紙送り制御文字として解釈します。
fb	アラビア語およびヘブライ語のロケール・サポートを決定します。`/usr/bin/bprt` を指定する必要があります。幅は **80** に設定し、データ・ストリームは拡張 ASCII を示す **a** に設定する必要があります。分音記号を含む文書を印刷するには、フラグ **- tashkeel** を追加します。

フィルター属性に格納される値が、指定されたフィルターのコマンド・ストリングを示します。サポートされる PostScript レーザー・プリンターのエントリーは、次のようなものです。

```
_ _FIL COMMAND STRINGS FOR FILTER FLAGS (2 char,
         1st="f",2nd=flag)

fn     Command String for the "n" Filter.            /usr/bin/psc
                                                     %is
fp     Command String for the "p" Filter             /bin/pr -l%G_l
                                                     %d -w%G_w%d%F[
                                                     h] %I@1%ia
fb     Command String for the "b" Filter.            /usr/bin/bprt
                                                     -w%I_w -d%I_d
                                                     -tashkeel
```

これらと同じ属性値が、コロン・ファイルには次のように格納されています。

```
:055:_ _FIL::
:269:fn::/usr/bin/psc%is
:270:fp::/bin/pr -l%G_l%d -w%G_w%d%F[h] %I@1%ia
```

fd 属性は、代表的なフィルター属性です。この属性は、仮想プリンターに対して DVI フィルターを指定するために使用されます。このフィルターを指定するには、SMIT または **chvirprt** コマンドを使用します。例えば、SMIT を使用して DVI フィルターを指定するには、次のように入力します。

```
smit lsvirprt
```

目的の仮想プリンターを選択し、次のように入力します。

```
fd=/usr/bin/dvi_to_printer%ip
```

ここで *dvi_to_printer* は、**tex** からの DVI 出力をプリンターが予期するフォーマットに変換するフィルターの絶対パス名を指定します。`%ip` を指定すると、印刷ファイルを処理するために、ASCII パイプライン (**ia** 属性) の代わりにパススルー・パイプライン (**ip** 属性) が強制的に使用されます。パススルー・パイプラインを使用すると、フィルターからの出力は変更されずにプリンターに渡されます。

fd 属性に DVI フィルターを指定した後、**lpr -d** *DviFile* や **qprt -fd** *DviFile* などの印刷コマンドを送信できます。それぞれのコマンドの **-d** フラグと **-fd** フラグは、DVI フィルターを通して *DviFile* (**tex** によって作成される出力ファイル) を渡し、結果をプリンターに送ります。

仮想プリンターのディレクトリー属性

ディレクトリー属性は、_ **_DIR** グループ・ヘッダー属性の下にグループ化されます。これらの属性は、印刷要求を処理するために必要な各種ファイルのパス名を格納します (変換テーブル、ヘッダー・ページとトレーラー・ページのテキストを格納するファイル、ダウンロード可能なフォント・ファイル、一時ファイルなど)。

ディレクトリー属性名の先頭文字は **d** で、2 文字目はディレクトリーを指定します。

次の例に、サポートされる PostScript レーザー・プリンターのディレクトリー属性値をいくつか示します。

```
_ _DIR                                  DIRECTORIES
d1  Directory Containing Stage 1        /usr/lib/lpd/pio/trans1
    Translate Tables
    (data stream to intermed.)
d2  Directory Containing Stage 2        /usr/lib/lpd/pio/trans2
    Translate Tables
    (intermediate to printer)
dD  Directory Containing Dummy Device   /usr/lib/lpd/pio/
    Files For Printers Driven By,
    But Not Attached To, the dev
    Host Computer
dF  Directory Containing Flags files    /var/spool/lpd/pio/@local/flags
    (keeps track of loaded fonts)
```

これらと同じ属性値が、コロン・ファイルには次のように格納されています。

```
:053:_ _DIR::
:160:d1::/usr/lib/lpd/pio/trans1
:161:d2::/usr/lib/lpd/pio/trans2
:509:dD::/usr/lib/lpd/pio/dev
:414:dF::/var/spool/lpd/pio/@local/flags
```

仮想プリンターの各種属性

_ **_MIS** グループ・ヘッダー属性は、各種のプリンター属性をグループ化します。各種属性は、文字 **m** で始まり、プリンター記述やプリンター・モデル番号などの値を格納します。

デバイス名とキュー名も、各種グループに格納されます。 **mn** 属性はデバイス名を格納し、**mq** 属性はキュー名を格納します。

次に、サポートされる PostScript レーザー・プリンターの代表的な各種属性を示します。

```
_ _MIS MISCELLANEOUS
mA    Printer Data Stream Description                 PostScript
mD    Name of message catalog Containing Attribute    pioattr1.cat
        Descriptors
mF    Path Name of Font File To Be Downloaded (must
        include download commands)
mL    Printer Description                             IBM 4029 Laser
                                                      Printer
mN    Printer model number                            029
mY    Datastream Mode to Which Printer is Restored at 3
        End of Job (0: IBM PPDS;  1: HP PCL;  2:
        Plotter;  3: PostScript)
mc    String to Send to Printer "mz" Times When       \0
        Job Is Cancelled
md    Output Data Stream Type (example: ascii);       ps
        Initialized By "piodigest"
mf    Path Name of the Default Formatter (used when   %Idf/piofpt
        running standalone)
mi    Input Data Stream Names (1 character,           s,l
        separated by commas) for mp Attribute
```

```
mm    File Name Of (Digested) Data Base; Init. By
        "piodigest" (mt.md.mn.mq:mv)
mn    Device name (example: lp0); Initialized By        lp1
        "piodigest"
mo    Command String to Invoke Device Driver I/F        %Ide/pioout %v
        Program (end of pipeline)                       [ABCDFINOPRS]
mp    Strings (separated by commas) That Identify       %%!,\320OPS
        Print File Data Types (see mi)
mq    Queue Name; Initialized By "piodigest"            ps1
```

これらと同じ属性が、コロン・ファイルには次のフォーマットで格納されています。

```
:058:_ _MIS::
:330:mA::PostScript
:332:mD::pioattr1.cat
:287:mF::
:331:mL::IBM 4029 LaserPrinter
:295:mN::4029
:516:mY::3
:301:mc::\0
:302:md::ps
:303:mf::%Idf/piofpt
:304:mi::s,l
:305:mm::
:306:mn::lp1
:307:mo::%Ide/pioout %v[ABCDFINOPRS]
:308:mp::%%!,\320OPS
:309:mq::ps1
```

仮想プリンターの作業変数属性

作業変数属性 (フォーマット設定中に値が変更される) は、文字 *w* で始まり、**_ _WKV** グループ・ヘッダー属性の下にリストされます。

サポートされる PostScript プリンターの代表的な作業変数属性のいくつかを、次に示します。

```
_ _WKV WORK VARIABLES
w7     Font Name for Header LIne of Text Converted to   %?%S_s%"Courie
        Postscript                                      r"%=%tCourier-
                                                        Bold%e%S_s%"Ti
                                                        mes-Roman"%=%t
                                                        Times-Bold%e%S
                                                        _s%"Helvetica"
                                                        %=%tHelvetica-
                                                        Bold%e%S_s%"Ti
                                                        mes-Italic"%=%
                                                        tTimes-BoldIta
                                                        lic%e%S_s%"Hel
                                                        vetica-Oblique
                                                        "%=%tHelvetica
                                                        -BoldOblique%e
                                                        %Iw8%;
wl    Smallest legal sheetfeeder drawer number          0
wu    Largest legal sheetfeeder drawer number           3
```

コロン・ファイルには、これらと同じ値が次のように格納されています。

```
:062:_ _WKV::
:472:w7::%?%S_s%"Courier"%=%tCourier-Bold%e%S_s%"Times-Roman"%=%tTim
es-Bold%e%S_s%"Helvetica"%=%tHelvetica-Bold%e%S_s%"Times-Italic"%=%t
Times-BoldItalic%e%S_s%"Helvetica-Oblique"%=%tHelvetica-BoldOblique%
e%Iw8%;
:370:wl::0
:381:wu::3
```

仮想プリンターのコマンド集合属性

コマンド集合属性は、_ _**CAG** グループ・ヘッダー属性の下にグループ化されており、プリンターを初期化するコマンドや、印刷ジョブの完了後にプリンターをリストアするコマンドなどの値を格納します。

このカテゴリーの属性は、文字 **c** で始まります。サポートされる PostScript プリンターに対する代表的なコマンド集合属性には、次のものがあります。

```
_ _CAG COMMAND AGGREGATES
ci    Command To Initialize the Printer              %Iez\4%?%G_j%{
                                                     1}%=%tstatusdi
                                                     ct begin%Iat %
                                                     Iar %?%Gmw%t%I
                                                     aF%; end%;
cr    Command To Restore the Printer at Job End      %o\4%Iex
```

これらと同じ属性は、コロン・ファイルには次のように格納されています。

```
:051:_ _CAG::
:144:ci::%Iez\4%?%G_j%{1}%=%tstatusdict begin %Iat %Iar %?%Gmw%t%IaF
%; end%;
:152:cr::%o\4%Iex
```

仮想プリンターの ASCII 制御コード属性

仮想プリンター属性の _ _**CTL** グループは、プリンターによって使用される ASCII 制御コードを格納します。

これらの属性は文字 **a** で始まり、用紙を次のページに送るために使用される制御コードなどの値を格納します。サポートされる PostScript プリンターに対する代表的な制御コードは、次のとおりです。

```
_ _CTL CONTROL CODES (ASCII)
aF    PostScript Command to Set Simplex/Duplex and   %?%G_Y%ttrue
        Tumble Mode                                  duplex %?%G_Y%
                                                     {1}%=%tfalse t
                                                     umble%etrue tu
                                                     mble%;%efalse
                                                     duplex%;
af    ASCII Control Code to Advance the Paper to     showpage
        Top of Next Page (FF)
ar    Cannot access message catalog pioattr1.cat.    %G_6%d setreso
                                                     lution
at    Cannot access message catalog pioattr1.cat.    %G_u%d setpape
                                                     rtray
```

コロン・ファイルには、これらの属性が次のように格納されています。

```
:052:_ _CTL::
:512:aF::%?%G_Y%ttrue duplex %?%G_Y%{1}%=%tfalse tumble%etrue tumble
%;%efalse duplex%;
:113:af::showpage
:119:ar::%G_6%d setresolution
:115:at::%G_u%d setpapertray
```

仮想プリンターのエスケープ・シーケンス属性

エスケープ・シーケンス属性は、文字 **e** で始まり、_ _**ESC** グループ・ヘッダー属性の下にグループ化されます。

代表的な PostScript プリンターの値は、次のとおりです。

```
_ _ESC ESCAPE SEQUENCES
ex    Command to Restore Printer Datastream Mode     \33[K\3\0\4\61
        (used only on restore)                       %?%GmY%{2}%>%t
                                                     %{8}%c%e%GmY%{
```

```
                                                    1}%+%c%;
ez      (used only on init/restore) Set initial     \33[K\5\0\4\61
          conditions                                \10\0\0
```

これらと同じ値は、コロン・ファイルには次のように格納されています。

```
:054:_ _ESC::
:514:ex::\33[K\3\0\4\61%?%GmY%{2}%>%t%{8}%c%e%GmY%{1}%+%c%;
:263:ez::\33[K\5\0\4\61\10\0\0
```

プリンター・コロン・ファイルのエスケープ・シーケンス

プリンター・バックエンドのデータベース・コロン・ファイル内で、属性値の組み込み参照とロジックは、属性文字列内の適切な位置にエスケープ・シーケンスを指定することによって定義されます。

これらのエスケープ・シーケンスをプリンターのエスケープ・シーケンスと混同しないでください。それぞれのエスケープ・シーケンスの先頭文字は、常に **%** (パーセント記号) 文字です。この文字は、エスケープ・シーケンスの開始を示します。2 文字目 (および場合によっては後続の文字) は、実行する操作を定義します。エスケープ・シーケンスの残りの文字 (ある場合) は、指定された操作の実行に使用されるオペランドです。

エスケープ・シーケンスによって実行される計算には、演算対象の整数や文字列へのポインターを保持するスタックを使用でき、内部変数 **a** から **z** を使用して整数値を後で使用するために保管できます。

% 文字はエスケープ・シーケンスの開始を定義するために使用されるので、データの一部である **%** 文字は、データベース内では 2 つの連続する **%** 文字 (**%%**) として表現する必要があります。作成される文字列には、ただ 1 つの **%** 文字が現れます。

属性文字列に指定できるエスケープ・シーケンスを次の表にリストして説明します。これらは、端末装置用の `terminfo` ファイルのエスケープ・シーケンスを基に、プリンター用に変更および拡張されたものです。

エスケープ・シーケンス	説明
%%	**%** (パーセント記号) 文字を生成します。

スタックからの ASCII 出力:

スタックからの ASCII 出力:

項目	説明
%d	スタックから整数値をポップし、先行ゼロなしの ASCII に変換します。 ASCII 数字の桁数を格納するために十分な大きさのフィールド幅を作成します。$_{printf}$ サブルーチンの **%d** と類似しています。
%[1-9]d	スタックから整数値をポップし、ASCII に変換します。 **d** の前に指定された数字に応じて、結果は 1 から 9 文字の長さになります。指定されたフィールド幅が値で埋まらない場合は、値の左側にゼロが埋め込まれます。値がフィールドに収まらない場合は、幅を超える上位桁が破棄されます。例えば、スタックからの値が **243** の場合、**%4d** の結果は **0243** になり、**%2d** の結果は **43** になります。スタック値が **-243** の場合、**%5d** の結果は **-0243** になります。

スタックからのバイナリー出力:

スタックからのバイナリー出力:

項目	説明
%c	スタックから整数値をポップし、最下位バイトを除くすべてを破棄します。
%h	スタックから整数値をポップし、下位 2 バイトを除くすべてを破棄します。
%a	スタックからの 2 バイトの順序が交替する (下位バイト、その次に上位バイト) ことを除き、**%h** と同様です。

入力文字列

入力文字列:

項目	説明
%Ixx	**xx** という名前の文字列属性を組み込みます。**%I** は再帰的に使用できます。つまり、組み込まれる文字列にも **%I** を含めることができます。組み込まれる文字列は現行スタックを継承しないことに注意してください。代わりに、新規スタックが割り当てられます。
%I[. . .]	複数の組み込みを連続して行う場合は、属性名をコンマで区切り、大括弧で囲むことができます。例えば、文字列 **%Icp%Icc%IeW** は **%I[cp,cc,eW]** として指定できます。
%Dxx	**xx** 属性によって指定した絶対パス名をもつファイルの内容を、プリンターにダウンロードします。印刷ジョブにはそのファイルの読み取り権限が必要です。この演算子の主な用途は、プリンターへのフォントのダウンロードです。
%"sss"	**sss** 文字列定数へのポインターをスタックにプッシュします。文字列ポインターに対して実行できる唯一の演算は、**%=** を使用して、文字列をそのスタック上にポインターがある別の文字列と比較することです。
%`xx	**xx** 属性によって指定されたコマンド・ストリングがシェルに渡されたときに生成される、標準出力を挿入します。` は抑音符号であることに注意してください。
%' "*String* **"**	引用符付き文字列をコマンドとしてサブシェルに渡します。引用符付き文字列内の二重引用符は、内部の引用符が文字列の区切り文字として読み取られないように、逆引用符にする必要があります。 ` は抑音符号であることに注意してください。

スタックへの入力整数

スタックへの入力整数:

項目	説明
%#xx"..@.."	**xx** という名前の文字列属性の選択された部分を抽出します。選択基準は、パターン **"...@..."** によって定義されます。選択パターンは、次の 3 つの部分で構成されています。 1. 抽出される文字列の直前にある文字列。プレフィックスの正規表現を指定しない場合、抽出される文字列は、サフィックスの正規表現によって指定されるパターンの前にある文字列全体で構成されます。 2. 抽出される文字列は、現在処理されている属性の **%#xx"..@.."** 演算シーケンスを置き換えます。 3. 抽出される文字列の直後にある文字列。サフィックスの正規表現を指定しない場合、抽出される文字列は、プレフィックスの正規表現によって指定されるパターンの後にある文字列全体で構成されます。 文字列属性の値がヌルならば、文字列は抽出されません。プレフィックスまたはサフィックスの正規表現が非ヌルで、属性値文字列に対応する一致項目がない場合、文字列は抽出されません。 **注:** アットマーク (@) と引用符 (") 文字は、リテラルとしての意味で使用する場合には別々の引用符の対で囲む必要があります。そうでなければ、プログラムはこれらの記号を区切り文字として読み取ります。 **%#** 演算子を別の **%#** 演算子の正規表現部分に組み込む場合は、アンパーサンド (@) と引用符 (") 文字をリテラルとしての意味で使用することはできません。この状況を避けるためには、組み込まれる **%#** 演算子を別の属性値に入れて、この新しい属性を外側の **%#** 演算子の正規表現に組み込んでください。

スタックへの入力整数:

項目	説明
%Gxx	**xx** という名前の整数属性を取得し、スタックにプッシュします。属性が整数でなく文字列の場合、文字列は ASCII 整数であると見なされます。これは $_{atoi}$ サブルーチンを使用して 2 進整数に変換され、スタックにプッシュされます。
%'c'	文字定数 **c** をスタックにプッシュします。スタックでは、この定数は整数値の下位バイトになります。上位バイトは 0 (ゼロ) に設定されます。
%{nn}	整数定数 **nn** をスタックにプッシュします。定数は 10 進値で、正または負のどちらも指定できます。

内部変数:

内部変数 **a** から **z** は、**%P**、**%Z**、および **%g** によって使用される整変数です。これらはゼロに初期化され、値は **%P** または **%Z** によって変更された場合に限って変更されます。これらの変数は 2 つの独立したセットに分かれています。一方のセットは **piobe** コマンドによってパイプラインの作成に使用され、もう一方のセットはフォーマッター専用に使用されます。フォーマッターのセットの値は、フォーマッターの処理期間中は維持されます。

内部変数:

項目	説明
%P[a-z]	スタックから整数値をポップし、指定された内部変数に格納します。例えば、**%Pf** は整数値をスタックから変数 **f** に移動します。
%Z[a-z]	指定された内部変数をゼロにリセットします。例えば、**%Zg** は変数 **g** に値 0 を格納します。
%g[a-z]	指定された内部変数の値をスタックにプッシュします。内部変数の値は変更されません。例えば、**%gb** は変数 **b** の整数値を読み取り、スタックにプッシュします。

算術演算子

算術演算子

項目	説明
%+ %- %* %/ %m	結果をスタックにプッシュします。
%+	スタックからポップした最初 2 つの値を加算します。例えば、**%{5}%{6}%+** は値 11 をスタックにプッシュします。
%-	スタックからポップした最初の値を、スタックからポップした 2 番目の値から減算します。例えば、**%{12}%{3}%-** は値 9 をスタックにプッシュします。
%*	スタックからポップした最初 2 つの値を乗算します。例えば、**%{2}%{3}%*** は値 6 をスタックにプッシュします。
%/	スタックからポップした 2 番目の値を、スタックからポップした最初の値で割ります。例えば、**%{6}%{2}%/** は値 3 をスタックにプッシュします。
%m	(モジュラス) **%/** と同様ですが、商ではなく剰余がスタックにプッシュされる点が異なります。例えば、**%{17}%{9}%m** は値 8 をスタックにプッシュします。

注: スタックから最初にポップされる値は、スタックに最後にプッシュされた値で、スタックから 2 番目にポップされる値は、スタックに最初にプッシュされた値です。

関係演算子および論理演算子

関係演算子および論理演算子

項目	説明
%= %> %< %!	真ならば 1、偽ならば 0 をスタックにプッシュします。
%=	スタックからポップした最初 2 つの値が同等かどうか。例えば、**%{2}%{2}%=** は値 1 (真) をスタックにプッシュし、**%{2}%{3}%=** は値 0 (偽) をスタックにプッシュします。
%>	スタックからポップした 2 番目の値が、スタックからポップした最初の値よりも大きいかどうか。例えば、**%{2}%{3}%>** は値 0 (偽) をスタックにプッシュします。
%<	スタックからポップした 2 番目の値が、スタックからポップした最初の値よりも小さいかどうか。例えば、**%{2}%{3}%<** は値 1 (真) をスタックにプッシュします。
%!	スタックからポップした値を否定し、結果をスタックにプッシュします。ゼロ以外の値は 0 になり、0 の値は 1 になります。例えば、**%{0}%!** は値 1 (真) をスタックにプッシュし、**%{1}%!** は値 0 (偽) をスタックにプッシュし、**%{2}%!** は値 0 (偽) をスタックにプッシュします。

注: スタックから最初にポップされる値は、スタックに最後にプッシュされた値で、スタックから 2 番目にポップされる値は、スタックに最初にプッシュされた値です。

ビット単位論理演算子:

ビット単位論理演算子:

項目	説明
%& %| %^ %~	結果をスタックにプッシュします。
%&	スタックからポップした最初 2 つの値の AND 演算を行います。例えば、**%{6}%{3}%&** は値 2 をスタックにプッシュします。
%|	スタックからポップした最初 2 つの値の OR 演算を行います。例えば、**%{6}%{3}%|** は値 7 をスタックにプッシュします。
%^	スタックからポップした最初 2 つの値の排他的 OR 演算を行います。例えば、**%{6}%{3}%^** は値 5 をスタックにプッシュします。
%~	スタックからポップした最初の値に対して 1 の補数を計算し、各ビットの値を反転します。例えば、**%{-1}%~** は値 0 (すべてのビットがオフ) をスタックにプッシュします (-1 の 2 の補数表記を前提として)。

条件 (if-then-else) 演算子:

条件 (if-then-else) 演算子:

項目	説明
%? expr %t thenpart %e elsepart %;	**%t** は値をスタックからポップしてテストします。値が **TRUE** (ゼロ以外) ならば、**thenpart** が実行されます。そうでなければ、**elsepart** (存在する場合) が実行されます。

else-if 構文

```
%? c1 %t b1 %e c2 %t b2 %e c3 %t b3 %e b4 %;
```

ここで、**c1**、**c2**、**c3** は条件を表し、**b1**、**b2**、**b3**、**b4** は本体を表します。例えば、**%?%{1}%t%{2}%e %{3}%;** は値 2 をスタックにプッシュし、**%gx%{6}%?%=%t%{2}%e%{3}%;%d** は内部変数 **x** の値が 6 ならば値 2 を出力します。値 **x** が 6 でなければ、値 3 が出力されます。

複雑なロジックを作成する際には、構造化形式で表記すると役に立つことがあります。前の例を構造化形式で表記すると、次のようになります。

項目	説明
%gx	値 **x** をスタックにプッシュします。
%{6}	値 6 をスタックにプッシュします。
%?%=%t	スタック値が等しい場合は
%{2}	値 2 をスタックにプッシュします。
%e	そうでなければ
%{3}	値 3 をスタックにプッシュします。
%;	endif
%d	値を ASCII フォーマットで出力します。

パススルー:

パススルー:

項目	説明
%x	(piocmdout サブルーチン呼び出しのみ。詳しくは、piocmdout サブルーチンを参照してください。) piocmdout サブルーチンへの **passthru** 引数によって指定されたバイト数を、入力から出力にパススルーします。詳しくは、piocmdout サブルーチンへの **passthru** 引数を参照してください。

ループ

ループ:

項目	説明
%wx	**While** ループ。対応する **%;** に達すると、内部変数 *x* (*x* は **a** から **z**) の値が 1 だけ減らされます。結果が 0 より大きい場合は、**%wx** の後に続く文字に実行が移ります。

モード:

項目	説明
%o	コマンド・ラインから (またはフォーマッターの処理中に) 更新された値の代わりに、データベースから取得したオリジナルのデフォルト値のみの使用を開始します。
%r	**%o** の前に使用されていた値の使用に戻ります。

パイプラインの指定変更:

項目	説明
%p	メインのパイプライン内で、プレフィックス・フィルター・パイプラインを組み込む先を指示します。存在しない場合は、メインのパイプラインの先頭が想定されます。属性名の先頭文字が **i** でない (つまり、メインのパイプラインでない) 場合は無視されます。
%z	メインのパイプライン内で、**pioout** 文字列 (デバイス・ドライバー・インターフェース・ルーチン) を組み込む先を指示します。存在しない場合は、メインのパイプラインの末尾が想定されます。属性名の先頭文字が **i** でない (つまり、メインのパイプラインでない) 場合は無視されます。
%ix	プレフィックス・フィルター文字列 (つまり、属性の 2 文字からなる名前の先頭文字が **f** である) 内でのみ指定できます。**x** 変数は、パイプライン ID 文字を表します。**%ix** 変数は、メインのパイプラインの属性名が **iy** でなく **ix** であることを示します。ここで、**y** は **-d** フラグに指定された (またはデフォルトの) パラメーターです。特殊なケースとして、**%i!** はヌル・ストリングをメインのパイプラインとして使用するように指定します。

コマンド・ライン・フラグ:

これらの演算子は、通常はパイプライン定義に使用され、印刷ジョブの実行依頼者によって指定されたフラグに適用されます。フォーマッターによって使用される属性文字列内で指定された場合は、フォーマッターに渡されるフラグに適用されます。有効なフラグ文字は **a** から **z**、**A** から **Z**、および **0** から **9** です。

項目	説明
%Cy	フラグ **y** がコマンド・ラインで指定された場合に、スタックに値 1 (真) をプッシュします。そうでなければ、値 0 (偽) をスタックにプッシュします。
%Fxy または **%F[...]**	**%?%Cy%t-x %I_y%;** の省略表現。**y** フラグがコマンド・ラインに指定された場合は、**-x yarg** を生成します。ここで、**yarg** は **y** フラグに指定された引数です。**!** が **x** に対して指定された場合、**-x** は生成されません。**yarg** に無保護の (奇数のバックスラッシュが直前に付いていない) 単一引用符または二重引用符が含まれている場合、エラー・メッセージが発行され、印刷ジョブは終了します。 **%Fxy** を使用して複数のフラグを指定する場合に、各フラグの **x** 値と **y** 値が同一ならば、フラグ文字のリストを大括弧で囲んで指定できます。例えば、**%Faa%Fbb%Fcc** は **%F[abc]** のように指定できます。 **y** または **[. . .]** によって参照される値の属性名は、先頭文字が _ (下線)、2 文字目が **y** または文字列 **[. . .]** に含まれる文字です。
%fxy または **%f[. . .]**	**%Fxy** および **%F[. . .]** と同様ですが、フラグ名と引数の間にスペースを入れない (引数がヌル・ストリングでない限り) 点が異なります。
%vxy または **%v[...]**	**%fxy** および **%f[. . .]** と同様ですが、 **pioout** コマンド (デバイス・ドライバー・インターフェース・プログラム) のコマンド・ストリング内でのみ使用され、**piobe** コマンド (印刷ジョブ・マネージャー) によって指定された指定変更値のフラグと引数を生成します。引数が定義済みのデフォルト値と等しい場合、フラグは生成されません。 **%v** の場合、**y** または **[. . .]** によって参照される値の属性名は、先頭文字が @ (アットマーク)、2 文字目が **y** または文字列 **[. . .]** に含まれる文字です。
%Ux または **%U[. . .]**	**piobe** コマンドに対して、**x** フラグ (または文字列 **[. . .]** に含まれる各フラグ) がパイプラインによって参照されていなくても、実際には参照されることを指示します。例えば、**x** フラグが、パイプライン内のフィルターではなくプリンター・コマンドによって参照されるようになります。これを使用すれば、フラグがコマンド・ラインで指定された場合に、**piobe** コマンドがフラグをリジェクトしなくなります。

プリンター・コロン・ファイルの規則

プリンターおよびプリンター・データ・ストリームの属性は、コロン・ファイルに格納されています。

コロン・ファイルは、`/usr/lib/lpd/pio/predef` ディレクトリーと `/var/spool/lpd/pio/@local/custom/*` ディレクトリーにあります。 `/usr/lib/lpd/pio/predef` ディレクトリーには定義済みデータベースがあり、`/var/spool/lpd/pio/@local/custom/*` ディレクトリーにはカスタマイズされたデータベースがあります。

次に、コロン・ファイル内のプリンターと属性の名前と値に関する規則を説明します。

コロン・ファイルのフォーマット

定義済みデータベースとカスタマイズされたデータベースのどちらのコロン・ファイルにも、それぞれの属性ごとに 5 つのフィールド (コロンで区切られる) があります。

コロン・ファイル内の各属性に対する 5 つのフィールドは、次のとおりです。

項目	説明
メッセージ・カタログ ID	属性の説明を保管するメッセージ・カタログを指定します。メッセージ・カタログ ID は、次の 3 つのうちいずれかの形式で指定できます。 • ヌル・ストリング: **mD** 属性の文字列値が、メッセージ・カタログのファイル名であると想定されます (例: `mydescriptors.cat`)。 • 1 文字: **pioattr.** *x***.cat** の省略形。ここで、*x* は 1 文字のカタログ ID です。この形式のカタログ ID は、通常はオペレーティング・システムによってのみ使用されます。 • カタログ・ファイル名: メッセージ・カタログのファイル名 (例: `mydescriptors.cat`)。 1 文字形式、またはカタログ・ファイル名形式のカタログは、**mD** 属性によって指定されたカタログ・ファイル名を指定変更します。 **メッセージ番号** この属性の説明を含む、カタログ内のメッセージ・インデックスを指定します。先行ゼロは無視されます。 **属性名** 2 文字を指定します。例外として、グループ・ヘッダー属性は 5 文字です。 **制限フィールド** 属性の制限を指定します。 **属性値の文字列** ゼロ文字から 1000 文字を指定します。

次に、コロン・ファイルの 1 行の例を示します。

```
:023:_w::80
```

属性名は **_w**、属性値文字列は **80**、属性の説明は **mD** 属性に指定されたメッセージ・カタログのメッセージ番号 **23** に保管されています。

注: 属性の説明はすべてメッセージ・カタログに保管されています。属性の説明が複数のプリンターに対して同じならば、各プリンターのデータベース内で、その属性は同じカタログとメッセージ番号を参照できます。異なるプリンターに対する異なる説明が同じ属性名に含まれている場合は、別々のメッセージ番号が使用されます。

仮想プリンターの属性名

仮想プリンターの属性名について定められている規則を説明します。

仮想プリンターの属性名には、次の規則が定められています。

- それぞれの属性名は固有であることが必要です。
- 属性名には、文字 **a** から **z**、**A** から **Z**、**0** から **9**、および _ (下線) を含めることができます。名前は数字から始めることはできません。
- すべての属性名の長さは 2 文字にする必要があります (例外として、グループ・ヘッダー属性名の長さは 5 文字にすることができます)。
- グループ・ヘッダーの属性名は、先頭を _ _ (2 つの下線) にする必要があり、5 文字より長くすることはできません。*グループ・ヘッダー属性* (従来はコメント属性と呼ばれていた) は、関連した属性のグループの先頭を示します。例えば、グループ・ヘッダー属性 _ **_FLG** は、コマンド・ライン・フラグのデフォルト値を定義する属性のグループの先頭を示します。属性のグループ化は読みやすさのために行われるもので、属性の処理方法には影響しません。

- _ (下線) で始まる属性名 (グループ・ヘッダーを除く) は、属性名の 2 文字目と同じ名前のコマンド・ライン・フラグによって指定変更できます。例えば、**qprt** コマンドに **-w 132** を指定すると、コロン・ファイル内で **_w** 属性に対して指定されている値が、値 **132** に指定変更されます。

仮想プリンターの自動属性

自動 属性は、自動的に指定される名前と値で、データベースには格納できません。

仮想プリンターの自動属性

項目	説明
@0	常にヌル・ストリングです。この属性名は、ヌル・ストリングの属性名が必要なすべての場所で使用できます。
@1	印刷するファイルの絶対パス名を含む文字列。この属性名は、パイプラインを定義する属性、およびパイプラインによって組み込まれる属性のみに適用されます。**-c** フラグが **qprt** コマンドに指定された場合、印刷されるファイルは一時ファイルです。
@2	piocmdout サブルーチンによって **%x** がコマンド・ストリング内で検出されたときにパススルーされるバイト数を示す整数 (piocmdout サブルーチンに渡される **passthru** パラメーターから得られる)。
@3	プリンターの接続方法を示す整数値。 **0** パラレルまたはシリアルのどちらでもない **1** パラレル **2** シリアル
@4	仮想プリンターの構成と印刷ジョブの処理に使用される、静的データ・ファイルとユーティリティー・プログラムがあるサブディレクトリー (burst、etc、fmtrs、fonts、predef、trans1、および trans2) を含む pio ディレクトリーの絶対パス名。このディレクトリーは、**qdaemon** によって呼び出される **piobe** コマンドを含むディレクトリーのサブディレクトリーであることが必要です。**@4** の値は、通常は /usr/lib/lpd/pio ディレクトリーです。
@5	仮想プリンターの構成と印刷ジョブの処理に使用される、動的データ・ファイルがあるサブディレクトリー (custom、ddi、dev、および flags) を含む pio ディレクトリーの絶対パス名。**@5** の値は、通常は /var/spool/lpd/pio ディレクトリーです。

次の属性名が、**piobe** コマンド (印刷ジョブ・マネージャー) から **pioout** コマンド (デバイス・ドライバー・インターフェース・プログラム) への通信に使用されます。属性値は、パイプラインに指定されている、デバイス・ドライバー・インターフェース・プログラムに渡されるフラグ引数によって参照されます。

項目	説明
@A	既に印刷済みのバイト数。
@B	印刷する合計バイト数。
@C	印刷ジョブの取り消し時に取り消し文字列 (**@D**) をプリンターに送信する回数。
@D	印刷ジョブを取り消す場合にプリンターに送信する文字列。
@I	「intervention required (介入が必要)」というメッセージを送信する先のユーザー。
@O	**pioout** コマンドによって生成され、データをプリンターに送信する代わりに保管するファイルの名前。
@P	印刷ファイルの最初のバイトが送信される前にプリンターに送信されるファイル (通常はヘッダー・ページ) の名前。
@S	印刷ファイルの最後のバイトが送信された後にプリンターに送信されるファイルの名前。

仮想プリンターの予約済み属性名

予約済み 属性名は、印刷ジョブ・マネージャーが前提とする名前です。

仮想プリンターの予約済み属性

項目	説明
先頭 2 文字が _ _	グループ・ヘッダー属性。
先頭文字が @	値は自動的に提供されます。
先頭文字が _	フラグ引数のデフォルト値。
先頭文字が i	入力データ・ストリームのパイプライン。
先頭文字が l	入力データ・ストリームの禁止フラグ。
先頭文字が f	フィルター・フラグのコマンド・ストリング。

先頭文字が z で、2 文字目が D、P、または S:

項目	説明
zD	`/var/spool/lpd/pio/custom/*` ディレクトリー内にあるときのコロン・ファイルのデフォルト状態 (**+** は拡張、**!** は縮小を示す)。
zP	コロン・ファイルの親コロン・ファイルの名前。親コロン・ファイルは、`/usr/lib/lpd/pio/predef/*` ディレクトリーにあることが前提です。
zS	コロン・ファイルの現在の状態 (**+** は拡張、**!** は縮小を示す)。

項目	説明
先頭文字が y	端末接続プリンターの値。

仮想プリンターの推奨属性名

推奨 属性名は、多くのフォーマッター・フィルターが前提としている名前です。

仮想プリンターの推奨属性

項目	説明
先頭文字が s	システム管理者値。
先頭文字が d	ディレクトリー・パス。
先頭文字が m	各種の値 (定数)。
先頭文字が w	作業値 (フォーマット設定中に変更される)。
先頭文字が c	コマンドの集合。
先頭文字が a	ASCII 制御コード。
先頭文字が e	プリンター・エスケープ・シーケンス。
先頭文字が t で 2 文字目が 0 から 9	フォーマッターによって使用される、ゼロ以上のステージ 2 変換テーブルの絶対パス名。複数の値はコンマで区切る必要があります。

属性値

属性値に関する規則が定められています。

属性値には、次の規則が定められています。

- プリンター名の形式は **4201-3** で、プリンター名 (**4201**) とモデル番号 (**3**) を表しています。
- 定義済みデータベースのファイル名の形式は、*PrinterType.DataStreamType* です。例えば、`4216-31.asc` は 4216 モデル 31 プリンターと ASCII データ・ストリームを示しています。
- カスタマイズされたデータベースのファイル名の形式は、`QueueName:QueueDeviceName` です (例: `proq:mypro`)。
- \ (バックスラッシュ) とその後に続く 1 から 3 桁の 8 進数字からなる属性値によって、非 ASCII 値を表現できます。8 進数シーケンスの開始を示すものではない \ (バックスラッシュ) は、**** または **\134** のどちらかによって表現する必要があります。

- 文字は形式 **\xXX** の 16 進表記によって表現できます。ここで、**XX** は 16 進値を表します。
- ブール値は、真を示す + (正符号) と、偽を示す ! (感嘆符) によって表現できます。
- 属性値はコロン・ファイル内にあるので、コロン文字が属性値に含まれていてはなりません。代わりに、コロンは **\072** によって表現します。
- ルックアップ・テーブルからの変換を必要とする整数属性を参照する属性値は、コロン・ファイル内の、参照される整数属性の後に指定する必要があります (例えば、文字列 **red** から等価な整数値 2 に変換する場合)。整数値は、コロン・ファイル内で定義されている順序どおりにコロン・ファイルから作成されます。属性値を最初にリストしておけば、整数属性が参照されるとき、**%G** エスケープ・シーケンスによって参照される前に整数属性が確実に変換されるようになります。
- シェル・コマンドはすべて、**bsh** ではなく **ksh** を使用して実行します。

コロン・ファイルの制限フィールド

コロン・ファイル内の制限フィールドには、SMIT ダイアログ情報と妥当性検査情報があります。

コロン・ファイル内の制限フィールドには、2 種類の情報 (SMIT ダイアログ情報と妥当性検査情報) があります。

SMIT ダイアログ情報

この情報は SMIT オブジェクトの作成に使用され、オブジェクト・データ・マネージャー (ODM) のコロン・ファイル属性を表します。これらのオブジェクトは、ファイルの印刷、プリンターの設定、およびデフォルト・ジョブ特性のダイアログに使用されます。

制限フィールドを使用すると、それぞれのオブジェクトに対して作成される sm_cmd_opt ODM オブジェクトのタイプについて、一定の制御が可能です。属性が常に表示されるか、まったく表示されないか、パイプライン内で参照された場合のみ表示されるかを制御できます。次のフィールドを変更できます。

- id_seq_num
- entry_type
- cmd_to_list_mode
- required
- op_type
- multi_select
- disp_values
- aix_values
- values_msg_file
- values_msg_set
- values_msg_id
- help_msg_id
- help_msg_loc

妥当性検査情報

コロン・ファイルが完全に作成され、印刷ジョブが実行依頼されると、妥当性検査情報によって属性値が妥当性検査されます。

印刷フォーマッターの例

この例では、印刷フォーマッターが、本書で説明しているプリンター・フォーマッター・サブルーチンとどのように相互作用するかを示しています。

印刷フォーマッターを作成する手順は、次の 4 つのステップからなります。

1. 次に示す印刷フォーマッター・ソース・ファイルの作成
2. インポート・ファイルの作成
3. エクスポート・ファイルの作成
4. 印刷フォーマッターのコンパイルとリンク

印刷フォーマッターのソース・ファイル

ASCII エディターを使用して、sample.c という名前のフォーマッター・ソース・ファイルを作成します。このファイルには次に示す行が含まれている必要があります。

```
#include <stdio.h>
#include <piostruct.h>

/* STRING CONSTANTS */
/* Initialize Printer, Restore Printer, Form Feed */
#define INIT_CMD    "ci"
#define REST_CMD    "cr"
#define FF_CMD      "af"

/* INTEGER and STRING VARIABLES */
/* page length, page width, top margin, bottom margin */
#define Pglen           (*(_Pglen + piomode))
#define Pgwidth         (*(_Pgwidth + piomode))
#define Tmarg           (*(_Tmarg + piomode))
#define Bmarg           (*(_Bmarg + piomode))

/* indentation, begin page, form feed?, pass-through? */
#define Indent          (*(_Indent + piomode))
#define Beginpg         (*(_Beginpg + piomode))
#define Do_formfeed     (*(_Do_formfeed + piomode))
#define Passthru        (*(_Passthru + piomode))

/* initialize printer?, restore printer? */
#define Init_printer    (*(_Init_printer + piomode))
#define Restoreprinter  (*(_Restoreprinter + piomode))

/* Command names: form feed, vertical increment and decrement */
#define Ff_cmd          (*(_Ff_cmd + piomode))
#define Vincr_cmd       (*(_Vincr_cmd + piomode))
#define Vdecr_cmd       (*(_Vdecr_cmd + piomode))

/* Work variables for vertical increment and decrement */
#define Vincr           (*(_Vincr + piomode))
#define Vdecr           (*(_Vdecr + piomode))

/* Variables referenced by above #defines */
int *_Pglen, *_Pgwidth, *_Tmarg, *_Bmarg, *_Indent, *_Beginpg, *_Do_
formfeed, *_Passthru, *_Init_printer, *_Restoreprinter, *_Vincr, *_V
decr;
struct str_info *_Ff_cmd, *_Vincr_cmd, *_Vdecr_cmd;

/* TABLE OF ATTRIBUTE VALUES */
struct attrparms attrtable[] = { /*
name  data type  lookup  address of pointer */
"_b",  VAR_INT,   NULL,   (union dtypes *) &_Bmarg,
"_g",  VAR_INT,   NULL,   (union dtypes *) &_Beginpg,
"_i",  VAR_INT,   NULL,   (union dtypes *) &_Indent,
"_j",  VAR_INT,   NULL,   (union dtypes *) &_Init_printer,
"_l",  VAR_INT,   NULL,   (union dtypes *) &_Pglen,
```

```
"_t",  VAR_INT,   NULL,   (union dtypes *) &_Tmarg,
"_w",  VAR_INT,   NULL,   (union dtypes *) &_Pgwidth,
"_J",  VAR_INT,   NULL,   (union dtypes *) &_Restoreprinter,
"_Z",  VAR_INT,   NULL,   (union dtypes *) &_Do_formfeed,
"wp",  VAR_INT,   NULL,   (union dtypes *) &_Passthru,
"wf",  VAR_STR,   NULL,   (union dtypes *) &_Ff_cmd,
"wi",  VAR_STR,   NULL,   (union dtypes *) &_Vincr_cmd,
"wy",  VAR_STR,   NULL,   (union dtypes *) &_Vdecr_cmd,
"wV",  VAR_INT,   NULL,   (union dtypes *) &_Vincr,
"wD",  VAR_INT,   NULL,   (union dtypes *) &_Vdecr,
NULL,  0     ,   NULL,    NULL };
int pglen, tmarg, bmarg, vpos, vtab_base;
struct shar_vars sharevars;

struct shar_vars * /*** Setup Processing ***/
setup(argc, argv, passthru)
   unsigned argc;
    char *argv[];
    int passthru:
{

/* Initialize variables and command line values */
(void) piogetvals(attrtable, NULL);
(void) piogetopt(argc, argv, NULL, NULL);
/* (need to verify values entered by user) */

/* Initialize work variables */
pglen = Pglen * Vincr;
tmarg = Tmarg * Vincr;
bmarg = Bmarg * Vincr;
piopgskip = Beginpg - 1;

/* Check for pass-through option */
if (Passthru = passthru)
          return(NULL);

/* Initialize pointers to vertical spacing */
/* variables shared with formatter driver  */
/* (Refer to /usr/include/piostruct.h)     */
sharevars._pl            = &pglen;
sharevars._tmarg         = &tmarg;
sharevars._bmarg         = &bmarg;
sharevars._vpos          = &vpos;
sharevars._vtab_base     = &vtab_base;
sharevars._vincr         = &Vincr;
sharevars._vincr_cmd     = (&Vincr_cmd)->ptr;
sharevars._vdecr         = &Vdecr;
sharevars._vdecr_cmd     = (&Vdecr_cmd)->ptr;
sharevars._ff_cmd        = (&Ff_cmd)->ptr;
sharevars._ff_at_eof     = &Do_formfeed;
return(&sharevars);
}

initialize() /*** Initialize the Printer ***/
{
if (Init_printer)
           (void) piocmdout(INIT_CMD, NULL, 0, NULL);
return(0);
}

lineout(fileptr)  /***  Format a Line  ***/
FILE *fileptr;
{
int ch, charcount = 0;
for (ch = 0; ch < Indent; ch++)
                  pioputchar(' ');
while ((ch=piogetc(fileptr)) != '\n' && ch != EOF
             && charcount < Pgwidth) {
           charcount++;
                     pioputchar(c);
```

```
}
vpos += Vincr;
return(charcount);
}

passthru()  /*** Pass-through Option ***/
{
int ch;
while ((ch = piogetc(stdin)) != EOF)
                      pioputchar(ch);
if (piodatasent && Do_formfeed)
                    (void) piocmdout(FF_CMD, NULL, 0, NULL);
return(0);
}

restore() /*** Restore the Printer ***/
{
if (Restoreprinter)
              (void) piocmdout(REST_CMD, NULL, 0, NULL);
return(0);
}
```

印刷フォーマッターのコンパイルおよびリンク

エディターを使用して、sample.imp という名前のインポート・ファイルを作成します。このファイルには、次に示す内容が含まれている必要があります。

```
#!
main
piogetvals
piogetopt
piomsgout
pioexit
piomode
piodatasent
piopgskip
statusfile
piocmdout
piogetstr
```

エディターを使用して、sample.exp という名前のエクスポート・ファイルを作成します。このファイルには、次に示す内容が含まれている必要があります。

```
#!
setup
initialize
passthru
restore
lineout
```

次のように入力して、フォーマッターをコンパイルし、リンクします。

```
cc -o sample -bI:sample.imp -bE:sample.exp sample.c
```

バックエンドと qdaemon の相互作用

qdaemon とバックエンドは、状況ファイルを介してやり取りをします。

qdaemon とバックエンドは、状況ファイルを介してやり取りをします。92 ページの『libqb 内のバックエンド・ルーチン』では、この通信要件を満たすためにバックエンドが使用する必要があるライブラリー・ルーチンのセットについて説明しています。これらのルーチンは、/usr/lib/libqb.a ライブラリーにあります。

状況ファイル

それぞれのキュー、および関連したキューごとに、状況ファイルが存在します。

qdaemon プロセスがバックエンドを呼び出すときに (詳しくは、**qdaemon** を参照)、次のパラメーターをこの順序で渡します。

1. `/etc/qconfig` ファイルにあるパラメーター。詳しくは、`/etc/qconfig` を参照してください。
2. **enq** コマンド (詳しくは、**enq** を参照) が認識しなかったフラグ (指定された順に)。コマンド・ラインでは、これらのフラグの前に **-o** オプションが指定されます。
3. 印刷される 1 つ以上のファイルの名前。

それぞれのデバイスおよび関連したキューごとに、状況ファイルが存在します。これらのファイルは、`/var/spool/lpd/stat` ディレクトリーにあります。

状況ファイルは、**qdaemon** プロセスとバックエンドの通信手段を提供します。**qdaemon** は、ファイルの日付、バースト・ページを印刷するかどうか、印刷部数などの情報を渡します。バックエンドは、直前に実行を完了したジョブの課金情報を返します。またバックエンドは、印刷したページ数とジョブの完了パーセントを定期的に更新します。この情報は、**qchk** コマンドによって読み取られます。詳しくは、**qchk** を参照してください。

注: バックエンドは、状況ファイルへの書き込みを明示的に行ってはなりません。書き込みを行うには、**libqb** ライブラリー・ルーチンを呼び出す必要があります。

このルーチンが呼び出される理由は、次のとおりです。

- バックエンドが状況ファイルに直接アクセスする手間が省けます。
- バックエンドを再作成する必要なく、状況ファイルのフォーマットを変更できます。状況ファイルのフォーマットが変更された場合には、バックエンドの再リンクのみが必要です。

ライブラリー・ルーチンに共通する特定のデータを初期化するために、バックエンドはルーチン **log_init** を呼び出す必要があります。詳細については、92 ページの『libqb 内のバックエンド・ルーチン』を参照してください。呼び出しは次のとおりです。

```
log_init();
```

このルーチンを呼び出して、状況ファイル・インターフェースを初期化する必要があります。失敗した場合、**log_init** ルーチンはライブラリーにある他の **log_** ルーチンと同様に、値 -1 を戻します。

複数部の印刷

enq -N コマンドを使用して、ファイルを複数部印刷できます。

enq -N コマンドは、ファイルを複数部印刷します。例えば、ファイル `filename` を 5 部印刷するには、次のコマンドを入力します。

```
enq -N5 filename
```

enq コマンドは、この情報を **qdaemon** プロセスに渡し、プロセスはこの情報を状況ファイルに書き込みます。バックエンドは `get_copies` ルーチンを呼び出してこの情報を取得する必要があります。このルーチンは、要求された合計部数を戻します。詳細については、92 ページの『libqb 内のバックエンド・ルーチン』を参照してください。

ジョブ状況情報

qchk コマンドを使用して、現在実行中の印刷ジョブに関する情報を表示できます。

qchk コマンドは、現在実行中のジョブに関する情報を表示します。開始元、タイトル、印刷ページ数、完了パーセントなどの情報があります。この情報はすべて、状況ファイルから得られます。ほとんどの情報は、バックエンドが最初に起動されたときに、**qdaemon** によってセットアップされます。ただし、**pages printed** フィールドと **percent done** フィールドは例外で、これらはバックエンド自体による入力が必要です。

この情報を提供するために、バックエンドは定期的に **libqb** を呼び出して、次の機能を実行する必要があります (92 ページの『libqb 内のバックエンド・ルーチン』を参照)。

- **log_progress(pages,percent)**
- **log_pages(pages)、個々の機能ごとに**
- **log_percent(percent)、個々の機能ごとに**

バックエンドはこれらのルーチンをいつでも呼び出すことができますが、それぞれのページの最後に 1 回呼び出すことをお勧めします。

印刷ジョブの課金

バックエンドがジョブを完了すると、**qdaemon** は状況ファイルを読み取って課金情報を取得します。

`qconfig` ファイルが課金用にセットアップされていれば、課金情報がファイルに書き込まれ、課金は最終的にアカウンティング・プログラムによって処理されます。この結果、印刷要求を実行依頼しているユーザーに請求書 (実際の、または仮想的な) が送られます。

バックエンドは、ルーチン `log_charge(charge)` を使用して、課金情報を **qdaemon** に返します。(92 ページの『libqb 内のバックエンド・ルーチン』を参照してください。) バックエンドは、このルーチンを終了時に呼び出す必要があります。また、ジョブの印刷中に、`log_progress` とともにこのルーチンを呼び出す必要があります。詳しくは、『ジョブ状況情報』を参照してください。一方、ジョブが取り消された場合は、その時点までに印刷されたページに対して課金は行われません。

現行のアカウンティング・プログラムはすべて、課金を印刷ページ数として解釈します。ただし、バックエンドは印刷ページ数に任意の乗数 (整数または分数) を掛けた課金を設定できます。

ジョブ・アカウンティングについて詳しくは、48 ページの『印刷スプーラー』を参照してください。

終了コード

バックエンドが終了すると、例えばジョブが正常に完了したかどうか、デバイスがまだ使用可能かどうかを示す情報を得るために、終了コードが **qdaemon** によって検査されます。

バックエンドが終了すると、例えばジョブが正常に完了したかどうか、デバイスがまだ使用可能かどうかを示す情報を得るために、**qdaemon** はバックエンドの終了コードを検査します。このため、バックエンドが終了コードについて同じ規則を使用することが重要です。バックエンドは、ここに示すコードの値を得るために、**#include <IN/standard.h>** を使用する必要があります。

許容される終了コードは、次のとおりです。

項目	説明
EXITOK	問題は検出されませんでした。
EXITBAD	パラメーターを処理できませんでした。よくある 2 つの例は、フラグが有効でないことと、ファイルを開けなかったことです。**qdaemon** は、デバイスの状態 (**qchk** によって表示される) をオフに設定し、コンソールにメッセージを送信し、いずれかのユーザーが状態を明示的にオンに再設定する (**enq -P***queuename* **-U** コマンドを使用して) までは、そのデバイス上で他のジョブを実行しません。
EXITERROR	バックエンドはジョブの印刷を完了できませんでした。**qdaemon** は、同じデバイス上で同じジョブを最初から再始動します。**qdaemon** は、ジョブを再始動する回数に制限を適用します。
EXITFATAL	手操作による介入を必要とするデバイスの問題が原因で、ジョブを完了できませんでした。**qdaemon** は、デバイスの状態 (**qchk** によって表示される) をオフに設定し、コンソールにメッセージを送信し、いずれかのユーザーが状態を明示的にオンに再設定する (**enq -P***queuename* **-U** コマンドを使用して) までは、そのデバイス上で他のジョブを実行しません。
EXITSIGNAL	バックエンドは **SIGTERM** シグナル (**#include <signal.h>**) によって中断されました。
EXITWARN	バックエンドは **qdaemon** に警告を発行しました。ジョブは正常に完了している場合もしていない場合もありますが、どちらの場合にも、**qdaemon** がバックエンドから **EXITWARN** を受け取ると、**qdaemon** は問題を説明するメッセージを戻します。

戻されるエラー・メッセージ

エラー・イベントが発生すると、バックエンドはメッセージをユーザーに送信します。

メッセージを送信する前に、バックエンドは **PIO_IPCWRITEFD** 環境変数を検査する必要があります。環境変数が設定されていれば、メッセージはパイプによって印刷スーパーバイザーに送信されます。印刷スーパーバイザーはメッセージを解釈して、ユーザーに送信します。**PIO_IPCWRITEFD** 環境変数が設定されていない場合、バックエンドは `sysnot` ルーチンを使用してユーザーにメッセージを送信します。

qdaemon 印刷スプーラーは、常に `sysnot` (92 ページの『libqb 内のバックエンド・ルーチン』) ルーチンを使用してメッセージを送信します。基本オペレーティング・システム以外の印刷スプーラーは、`sysnot` ルーチンまたはパイプのどちらを使用してメッセージを送信することもできます。

sysnot ルーチン

バックエンドは、`sysnot` ルーチンを使用してユーザーに直接メッセージを送信できます。

`sysnot` ルーチンは、ユーザーにメッセージをメールで送信するか、ユーザーの端末装置にメッセージを書き込むことができます。`sysnot` ルーチンを呼び出すには、次の構文を使用します。

```
sysnot(user, host, message, pref)
      char *user;
      char *host;
      char *message;
      unsigned int *pref;
```

pref パラメーターの値は、DOMAIL または DOWRITE のどちらかにする必要があります。DOMAIL は、エラー・メッセージをユーザーにメールで送信します。DOWRITE は、ユーザーがログオンしている場合には、ユーザーの端末装置にメッセージを書き込みます。ユーザーがログオンしていない場合、メッセージはユーザーにメールで送信されます。DOMAIL と DOWRITE の定数は、`/usr/include/IN/backend.h` ファイルに定義されています。

詳細については、92 ページの『libqb 内のバックエンド・ルーチン』を参照してください。

パイプ

バックエンドは、パイプを使用して印刷スーパーバイザーにメッセージを送信することによって、ユーザーにメッセージを送信できます。このメカニズムは、プリンター・バックエンドと印刷スーパーバイザーの間で単方向通信パスを提供します。

印刷スーパーバイザーは、名前なしのパイプを開いて、2 つのファイル・ディスクリプターを取得する必要があります (1 つは読み取り操作用、1 つは書き込み操作用)。印刷スーパーバイザーは、fork サブルーチンと exec サブルーチンを使用してプリンター・バックエンドを呼び出す前に、**PIO_IPCWRITEFD** 環境変数の書き込み側をエクスポートする必要があります。**PIO_IPCWRITEFD** 環境変数が設定されている場合、プリンター・バックエンドはメッセージをパイプの書き込み側に書き込みます。

印刷スーパーバイザーは、通常は select サブルーチンを呼び出して、着信メッセージ用のパイプの読み取り側をポーリングします。waitpid サブルーチンを使用してプリンター・バックエンドの終了状況を検査するほかに、印刷スーパーバイザーはパイプ上での入出力をポーリングします。印刷スーパーバイザーは、**SIGCHLD** シグナルのシグナル・ハンドラーをセットアップし、パイプ上でブロック読み取りを実行します。シグナル・ハンドラーは、プリンター・バックエンドの終了状況を検査して、必要なアクションを実行します。未読メッセージがパイプに残っていなければ、印刷スーパーバイザーはパイプを閉じ、他のクリーンアップ作業に進みます。

メッセージのフォーマット

プリンター・バックエンドによって送信されるメッセージはそれぞれ、メッセージ・ヘッダー・フレーム、任意数のパラメーター・ヘッダー・フレーム (ない場合もある)、完全に展開されたメッセージ、および任意数のパラメーター (ない場合もある) からなるテキストで構成されます。

メッセージ・ヘッダーは、メッセージ・タイプ、メッセージ・カタログ情報、展開メッセージ・テキストの長さ、および可変メッセージ・パラメーターの数を指定します。可変メッセージ・パラメーターは、メッセージ・カタログから抽出された基本メッセージ・テキストから、展開メッセージ・テキストを作成するために使用されます。メッセージ・ヘッダー・フレームとメッセージ・パラメーター・ヘッダー・フレームの構造フォーマットは、`/usr/include/piostruct.h` ファイルに定義されています。

パイプからメッセージを抽出する際に、印刷スーパーバイザーはメッセージ・ヘッダー・フレームを読み取り、次にメッセージ・パラメーター・ヘッダー・フレーム (0 から 9、メッセージ・ヘッダー・フレーム内で指定されたパラメーターの数によって示される) を読み取ります。印刷スーパーバイザーは、展開メッセージ・テキストを読み取り (この長さはメッセージ・ヘッダー・フレーム内で指定される)、次にパラメーター (存在する場合) を読み取ります。パラメーターのタイプと長さは、個々のメッセージ・パラメーター・ヘッダー・フレーム内で指定されます。

メッセージのタイプは、メッセージ・ヘッダー・フレーム内で指定されます。メッセージ・タイプは次のとおりです。

- **ID_VAL_EVENT_ABORTED_BY_SERVER**
- **ID_VAL_EVENT_WARNING_RESOURCE_NEEDS_ATTENTION**

実際のメッセージ・テキストは、展開フォーマットです。サーバーのロケールのメッセージ・カタログ・ファイルからパラメーターが抽出された後、パラメーターがメッセージ・テキスト内に配置されます。印刷スーパーバイザーはこのメッセージ・テキストを使用でき、また提供されたメッセージ・カタログ情報とメッセージ・パラメーターから、独自のメッセージ・テキストを作成することもできます。ただしプリンター・バックエンドは、メッセージ・カタログ情報 (メッセージ番号、セット番号、およびカタログ名) と可変メッセージ・パラメーターをどの場合にも提供できません。このため、印刷スーパーバイザーは、カタログ名フィールド (**pm_catnm** フィールド) を検査して、カタログ名がヌル・ストリングかどうかを判別します。カタログ名がヌル・ストリングならば、印刷スーパーバイザーは提供された展開メッセージ・テキストを使用する必要があります。

カタログ名が指定されている場合、印刷スーパーバイザーはカタログからメッセージを抽出し、提供されたメッセージ・パラメーターをメッセージ内に配置できます。メッセージ・パラメーターは、整数型または文

字列型です。ただし、メッセージ・パラメーターは、展開メッセージ・テキストに連結された文字列としてプリンター・バックエンドから渡されます。印刷スーパーバイザーが指定されたカタログからメッセージを抽出し、メッセージ内にパラメーターを配置する場合には、次の規則が適用されます。

- パラメーターの型は整数または文字列のどちらかですが、パイプ内では常に後続 **NUL** 文字が付いた文字列として渡されます。文字列フォーマットの各パラメーターの長さは、パラメーターに関連したヘッダー・フレーム内で指定されます。
- 抽出されたメッセージには、printf サブルーチンによって認識されるエスケープ・シーケンスが含まれていることがあります。このため、メッセージにデータを追加する際に、印刷スーパーバイザーは **%s**、**%d**、**%c** などのエスケープ・シーケンスを検査し、パラメーターを適切に変換します。**%n$s** または **%n$d** を使用して、定位置パラメーターが指定される場合もあります。この場合、印刷スーパーバイザーは指定された順序でパラメーターを入力します。
- 最大で 9 つのパラメーターを指定できます。このため、印刷スーパーバイザーは *char 型の変数を 9 つ使用でき、提供された適切なパラメーター・ストリングにこれらの変数を割り当てることができます。定位置指定子と整数指定子をすべて置換した後、パラメーターを printf サブルーチンに渡すことができます。例えば、抽出されたメッセージ・テキストに次の内容が含まれているとします。

  ```
  Error %8$d in opening %6$s file
  ```

  印刷スーパーバイザーは、このメッセージを次のように変換します。

  ```
  Error %s in opening %s file
  ```

  印刷スーパーバイザーは、最初の変数パラメーター・ポインターを 8 番目のパラメーターに割り当て、2 番目の変数パラメーター・ポインターを 6 番目のパラメーターに割り当て、残りの変数パラメーターをヌル・ストリングに割り当てます。印刷スーパーバイザーは、sprintf サブルーチンまたは同様なサブルーチンを呼び出して、9 つの変数パラメーター・ポインターを関数へのパラメーターとして渡します。
- プリンター・バックエンドは、各パラメーターの正しい型 (整数または文字列) を指定しますが、パイプ内ではすべてのパラメーターが文字列として渡されます。抽出されたメッセージにパラメーターを配置する際に、フィールド幅と精度の処理には適切な型を使用する必要があります。
- プリンター・バックエンドは、メッセージのメッセージ・カタログ情報とパラメーターを渡す場合も、渡さない場合もあります。このため、印刷スーパーバイザーは拡張メッセージ自体を受け入れることができるか、またはカタログ情報とパラメーターを受け入れてから、メッセージを適切に構築できることが必要です。

キューの状態

qchk コマンドは、特定のデバイスの状況を表示します。

表示されるテーブルのエントリーの 1 つは、キューの現在の状態を示します。この情報は、状況ファイルから得られます。有効なキュー状態とその説明のリストについては、状況ファイル内の `/usr/include/IN/backend.h` を参照してください。

通常、**qdaemon** は状況ファイルを最新に保ちます。ただし、一部のバックエンドは、デバイスに出力を送信できなくなった場合に状態を明示的に WAITING に設定し (**#include <IN/backend.h>**)、出力が再開したときに再び RUNNING に設定できます。例えば、各ページの最後に一時停止してユーザー応答を待機するバックエンドは、この期間は状況を WAITING に設定できます。

`log_status(status)` ルーチンを使用して、ジョブの状況を RUNNING から WAITING に変更し、再び元に戻すことができます。詳しくは、92 ページの『libqb 内のバックエンド・ルーチン』を参照してください。パラメーターは、新しく変更する状況です。

キュー・デバイス上で **DEV_WAIT** 状態が発生した場合は、`enq -U -Pqueue` を発行してキューを作動可能状態にすることを試みます。これが有効でない場合は、そのキュー内のジョブをすべて移動してから、他のキューのフラッシュを行って **qdaemon** を停止するために、`enq -G` を発行します。その後、**qdaemon** を再始動します。

SIGTERM の受信時の終了

ユーザーが **qcan** を使用して実行中のジョブを取り消すと、コマンドは要求を **qdaemon** に渡します。

バックエンドは、シグナルを受信した直後に印刷を停止する必要があります。これを実現するための方法は 2 とおりあります。

1 つ目の方法として、バックエンドが SIGTERM に関して特別な処理を実行できない場合には、シグナルによってバックエンド・プロセスが即時に停止します。このオプションは最も単純ですが、バックエンドは終了前にクリーンアップ (回線速度のリセット、用紙先端への用紙送り、電話のハングアップ) を実行できません。

2 つ目の方法として、バックエンドが SIGTERM をキャッチし、必要なクリーンアップ・タスクを実行して、EXITSIGNAL **(#include <IN/standard.h>)** を出して終了できます。この特殊な終了コードは、ジョブが取り消されたことを **qdaemon** に指示します。

SIGTERM をキャッチするバックエンドは、シグナルを受信した後、速やかに終了する必要があります。

詳しくは、**qcan** コマンド、および **qdaemon** コマンドを参照してください。

libqb 内のバックエンド・ルーチン

バックエンドは、一連のライブラリー・ルーチンを使用して **qdaemon** プロセスと通信します。

このトピックでは、バックエンドが **qdaemon** プロセスと通信するために使用する一連のライブラリー・ルーチンを定義します。これらのルーチンは、`/usr/lib/libqb.a` ライブラリー内にあり、バックエンドを作成する作業が可能な限り容易になるように設計されています。これらのバックエンド・ルーチンは、**ld** または **cc** のコマンド・ライン・オプション **-lqb** を使用することで対応可能になります。

バックエンドとともにこれらのルーチンを使用する方法については、 86 ページの『バックエンドと qdaemon の相互作用』を参照してください。

バックエンド・ルーチンの使用

項目	説明
`get_align()`	**TRUE** または **FALSE** を戻し、位置合わせ用紙送りを印刷するかどうかを示します。プリンターがアイドル状態で、すぐ後に新規ジョブを印刷する場合にのみ、用紙送りが印刷されます。用紙送りにより、用紙が用紙先端に位置合わせされるので、プリンターがアイドル状態のときに用紙が動かされた場合に便利です。
`get_cmd_line()`	ユーザーによって呼び出された **enq** コマンド・ラインを含む文字配列へのポインターを戻します。戻される文字列には、名前 `/usr/bin/enq`、指定されたファイル名、または **enq -o** オプションを使用してバックエンド・ルーチンに送信されたオプションは含まれません。例えば、ユーザーがコマンド・ライン `enq -Plp0 -Bgn -o -i15 filename` を入力した場合、**get_cmd_line** 関数は文字列 **-Plp0 -Bgn** を戻します。この関数は、ジョブの実行依頼時にユーザーが指定したコマンド・ライン・オプションを、バックエンド・ルーチンが取得する必要がある場合に便利です。
`get_copies()`	印刷部数を戻します。戻り値の型は **int** です。
`get_device_name()`	デバイス名を含む文字配列へのポインターを戻します。
`get_feed()`	印刷される送りページ数を戻します。戻り値の型は unsigned int です。送りページは、プリンターがアイドル状態になったときにのみ印刷されるブランク・ページです。これにより、プリンターから用紙を切り取りやすくなります。
`get_from()`	印刷要求を行った担当者の名前を含む文字配列を戻します。戻り値の型は char* です。
`get_header()`	**NEVER**、**ALWAYS**、または **GROUP** を戻します (**#include <IN/backend.h>**)。戻り値の型は unsigned int です。ヘッダーは、ファイルのタイトル、日付、宛先などの情報を示す、ファイルの前にあるページです。
`get_job_number()`	現在の印刷ジョブ番号を戻します。戻り値の型は int です。
`get_mail_only()`	ユーザーがメールだけを指定している場合に **TRUE** を戻します。

バックエンド・ルーチンの使用

項目	説明
get_qdate()	要求がキューに入れられた日付を示す文字列を戻します。戻り値の型は **char*** です。
get_queue_name()	キュー名を含む文字配列を戻します。
get_title()	印刷されるジョブのタイトルを含む文字配列を戻します。戻り値の型は **char*** です。
get_trailer()	**NEVER**、**ALWAYS**、または **GROUP** を戻します。戻り値の型は unsigned int です。トレーラーは、出力のユーザー名を示す、ファイルの後にあるページです。
get_to()	ジョブの実行対象である担当者の名前を含む文字配列を戻します。戻り値の型は char* です。
get_was_idle()	プリンターがジョブの開始時にアイドル状態だった場合は **TRUE** を戻します (用紙送りを行うか行わないかの判断に便利)。
	現行ジョブの印刷に対する課金を戻します。
log_charge(*charge*) int *charge*;	現行ジョブの印刷に対する課金を戻します。
log_init	ライブラリー・ルーチンに共通する特定のデータを初期化します。
log_pages(*pages*)	印刷ページ数について状況ファイルを更新します。
log_percent(*percent*)	ジョブの完了パーセントについて状況ファイルを更新します。
log_progress(log_pages (*int*),log_percent(*char*))	印刷ページ数とジョブの完了パーセントについて状況ファイルを更新します。この関数は、log_pages と log_percent を使用します。
log_status(*status*)	ジョブの状況を RUNNING から WAITING に変更し、再び元に戻します。パラメーターは、新しく変更する状況です。
put_header(*fnaddr*,*width*)	後続の用紙送りを行わずにヘッダー・ページを印刷し、印刷行数を戻します。*fnaddr* パラメーターと *width* パラメーターはオプションです。 **int (*fnaddr);** *fnaddr* パラメーターは、ヘッダー・ページに文字を印刷するために使用するフォーマット・サブルーチンを定義します。デフォルトは、putchar サブルーチンです。 **int *width;** *width* パラメーターは、用紙の幅を定義します。*width* パラメーターのデフォルト値は **80** です。
put_trailer(*user*,*fnaddr*,*width*)	後続の用紙送りを行わずに *user* のトレーラー・ページを印刷し、印刷行数を戻します。 *fnaddr* パラメーターと *width* パラメーターはオプションです。 **char *user;** **int (*fnaddr);** *fnaddr* パラメーターは、ヘッダー・ページに文字を印刷するために使用するフォーマット・サブルーチンを定義します。デフォルトは、putchar サブルーチンです。 **int *width** *width* パラメーターは、用紙の幅を定義します。*width* パラメーターのデフォルト値は **80** です。
sysnot(*user*,*host*,*message*,*pref*)	バックエンドがジョブを実行できない場合に、*message* を *user* に送信します。 **char *user;** **char *host;** **char *message;** **unsigned int *pref;** *pref* パラメーターの値は、ユーザーにメッセージをメールで送信するか、ユーザーの端末装置にメッセージを書き込むかを指定します。 /usr/include/IN/backend.h ファイルに定義されている有効な値は、次のとおりです。 **DOMAIL** エラー・メッセージをユーザーにメールで送信します。 **DOWRITE** ユーザーがログオンしている場合には、ユーザーの端末装置にメッセージを書き込みます。ユーザーがログオンしていない場合、メッセージはユーザーにメールで送信されます。

プリンター・コード・ページ変換テーブル

印刷ファイルのコード・ポイントからプリンター用のコード・ポイントへの変換は、2 つのステージからなるプロセスです (東洋言語のコード・ポイントの変換は、異なる方法で処理されます)。

最初のステージでは、印刷ファイルからのコード・ポイントが、中間コード・ページのコード・ポイントに変換されます。中間コード・ページは、サポートされる文字すべてに対応する 16 ビット整数のコード・ポイントで構成されます。中間コード・ページは、/usr/lib/lpd/pio/etc/codepage.txt ファイル内で定義されます。

マルチバイト・コード・セットの場合のプリンター・コード・ページ変換

印刷ファイルからコード・セットへのマルチバイト・コード・セット (MBCS) 変換は、1 バイト・コード・セット (SBCS) コード・ポイントの変換とは異なります。

マルチバイト環境での印刷ファイルからコード・セットへの変換は、2 つのステージからなるプロセスです。

コード・セット変換の最初のステージでは、印刷ファイルの入力コード・セットが処理コード・セットに変換されます。処理コード・セットは、**iconv** サブルーチンとロケール・データベース (DB) によってサポートされる MBCS コード・セットのいずれかであることが必要です。例えば、IBM-943、IBM-eucTW、IBM-eucKR などのコード・セットがあります。2 番目のステージでは、処理コード・セットがプリンター用の適切な出力コード・セットに変換されます。変換用の **iconv** コンバーターが存在する場合は、**iconv** サブルーチンがコード・セットを変換します。入力または出力のコード・セットと処理コードが同じならば、コード・セット変換は実行されません。

プリンター依存のコロン・ファイルの **Ti** 属性と **To** 属性は、変換するコード・セットの可能なフローを定義します。**Ti** 属性は、入力コード・セットと処理コード・セットの組み合わせを次のように指定します。

```
[Input_code_set, ... ]Process_code_set, ...
```

To 属性は、処理コードと出力コードの組み合わせを次のように指定します。

```
Process_code_set [Output_code_set0, Output_code_set1,
Output_code_set2, Output_code_set3,... ], ...
```

例えば、日本語プリンターの **To** 属性は、次のように定義されます。

```
::To::IBM-943[IBM-932, IBM-932, IBM-932], ibm-eucJP[IBM-932,
IBM-932, IBM-932,IBM-932]
```

それぞれの CSID ごとに出力コード・セットが指定されている場合は、文字セット ID (CSID) のすべての文字が ROM フォントを使用して印刷されます。そうでなければ、Xwindows フォントからのビットマップ・イメージが使用されます。Xwindows フォント・ファイル (それぞれの CSID のフォント・イメージを含む) のタイプは、/usr/lib/X11/nls ディレクトリーからファイルを読み取ることによって選択されます。

ステージ 1 のプリンター・コード・ページ変換

コード・ページ変換の最初のステージでは、中間コード・ページが作成されます。

次に示す例では、仮のコード・ページ 123 から中間コード・ページにコード・ポイントを変換するためのステージ 1 変換テーブルを生成します。

```
#include <piostruct.h>
#include <fcntl.h>
/*** Table to Translate Code Points for Input Code Page ***/
/*** "123" to Code Points for the Intermediate Code Page    ***/
short table[256] = {
/* 00 (000) */ CP, CP, CP, CP,
.
.
.
/* FC (252) */ CP, SC, 126, CP };
/*** Write the Table to a File (Error Processing Not Shown) ***/
main ( ) {
int fildes;
int fmt_type = 1;
fildes = open("/usr/lib/lpd/pio/transl/123", O_CREAT | O_WRONLY,\
```

```
0664);
write(fildes, "PIOSTAGE1XLATE00", 16);
write(fildes, &fmt_type, sizeof(fmt_type));
write(fildes, table, sizeof(table));
return(0);
}
```

コード・ポイント 252 の CP は、コード・ポイントを変更せずにコピーすることを示しています。コード・ポイント 253 の SC は、文字が中間コード・ページ内で定義されておらず、置換文字を代わりに印刷する必要があることを示しています。コード・ポイント 254 の 126 は、コード・ポイント 254 をコード・ポイント 126 に変換することを示しています。

qprt コマンドの **-X** フラグは、印刷ファイルのコード・ページ名を指定します。この値が 123 ならば、フォーマッターは /usr/lib/lpd/pio/trans1/123 ファイルからテーブルを読み取り、ステージ 1 変換に使用します。

ステージ 2 のプリンター・コード・ページ変換

コード・ポイント変換の 2 番目のステージでは、1 つ以上のステージ 2 変換テーブルが、中間コード・ページからプリンターに適したものにコード・ポイントを変換します。

データベース・コロン・ファイル内の **t0 から t9** 属性は、ステージ 2 変換テーブルの絶対パス名を指定します。**t0 から t9** 属性はそれぞれ、名前をコンマで区切ることによって複数のステージ 2 変換テーブルを指定できます。印刷フォーマッターは、これらのステージ 2 変換テーブルを読み込み、1 つのリングに連結します。現行プリンター・コード・ページ用のテーブルから始めて、フォーマッターは入力印刷ファイルの各文字を処理します。最初に行う判断は、文字がそのプリンター・コード・ページ内で定義されているかどうかです。言い換えれば、コード・ポイント値がテーブル内のコード・ポイント数より大きくなく、値が **SC** でないことを判断します。

文字がコード・ページに含まれていれば、変換されたコード・ポイントがプリンターに送信されます。フォーマッターは、適切なプリンター・コマンド・ストリングを送信して、プリンター・コード・ページを選択します。規則により、プリンター・コマンド・ストリングの 2 文字の属性名は、コマンド名配列のインデックス 0 にあります。文字がコード・ページに含まれていない場合、フォーマッターはリング内で次にあるステージ 2 変換テーブルの処理を繰り返します。文字を印刷できる変換テーブルがリング内で見つからない場合、フォーマッターは置換文字 (下線) を代わりに印刷します。

次の C 言語コード例は、**XYZ.999** という名前のステージ 2 変換テーブルを生成します。このテーブルは、中間コード・ページのコード・ポイントを、プリンターのコード・ページ用のコード・ポイントに変換します。**c1** 属性は、コード・ページ **XYZ.999** を選択するようにプリンターに指示する、プリンター・コマンド・ストリングを格納していることが前提です。

```
#include <piostruct.h>
#include <fcntl.h>
/*** Table to Translate Code Points for the Intermediate ***/
/*** Code Page to Code Points for a Printer Code Page ***/
struct transtab table[] = {
/* 00 (000) */ {CP}, {CP}, {CP}, {CP},
.
.
.
/* FC (252) */ {63}, {CP}, {94,1}, {SC} };

/*** Command Names for the Translate Table ***/
char cmdnames[][2] = {
{'c', '1'},              /* index 0 - select the code page */
{'e', 'b'} };            /* index 1 - next byte is graphic */
```

```
/*** Write the Table To a File (Error Processing Not Shown) ***/
main() {
int fildes;
int num_commands = sizeof(cmdnames) / 2;
fildes = open("/usr/lib/lpd/pio/trans2/XYZ.999", O_CREAT |
O_WRONLY,\ 0664);
write(fildes, "PIOSTAGE2XLATE00", 16);
write(fildes, &num_commands, sizeof(num_commands));
write(fildes, cmdnames, sizeof(cmdnames));
write(fildes, table, sizeof(table));
return(0);
}
```

コード・ポイント 252 の {63} は、コード・ポイント 252 をプリンターに送信する前にコード・ポイント 63 に変換することを示しています。コード・ポイント 253 の {CP} は、コード・ポイント 253 を変換せずにプリンターに送信することを示しています。コード・ポイント 254 の {94,1} は、コード・ポイント 254 をプリンターに送信する前にコード・ポイント 94 に変換することを示しています。{94,1} の ,1 は、コード・ポイントを送信する前に、コマンド名配列のインデックス 1 に入った 2 文字の属性名をもつプリンター・コマンド・ストリングをプリンターに送信することを示しています。コード・ポイント 255 の SC は、このステージ 2 変換テーブルによって記述されているプリンター・コード・ページを使用して、中間コード・ページのコード・ポイント 255 の文字を印刷できないことを示しています。

マルチバイト・コード・セットのプリンター・コード・ページ変換テーブル

変換テーブルは、2 つのコード・セットが共有しないコード・ポイント間のマップで構成されます。

プリンター・バックエンドは、コード・セットが iconv サブルーチンによってサポートされていない場合でも、/usr/lib/lpd/pio/transJP ディレクトリーに提供されている変換テーブルを使用して、他のコード・セットとの間で情報をやり取りできます。

入力または出力のコード・セットが iconv サブルーチンによってサポートされていない場合は、/usr/lib/lpd/pio/transJP ディレクトリー内で見つかった変換テーブルを使用して、サポートされないコード・セットがサポートされるコード・セットのいずれかに変換されるか、処理コード・セットに直接変換されます。 root 権限のあるユーザーは、変換テーブルを作成することによって、新規コード・セットをプリンターに追加できます。

新規変換テーブルの命名規則は、*FromCodeSetName_ToCodeSetName* です。変換テーブルはすべて、trans_dir ファイル内で定義する必要があります。変換テーブル内のコード・ポイントの **f_cp** は、前もってアルファベット順にソートしておく必要があります。

trans_dir ファイルと codeset.alias ファイルは、/usr/lib/lpd/pio/transJP ディレクトリー内にあります。trans_dir ファイルのフォーマットは、次のとおりです。

```
FromCodeSetName ToCodeSetName NameofTranslationFile
```

コード・セット別名は codeset.alias ファイル内で定義されます。 codeset.alias ファイルのフォーマットは、次のとおりです。

```
CodeSetName AliasName ...
```

例えば、IBM-943 プリンター上で新規コード・セットを使用して作成された MBCS ファイルを印刷するには、次の手順で行います。

1. /usr/lib/lpd/pio/transJP ディレクトリーに変換テーブルを作成します。新規ファイルの命名規則は、*NewCodeSetName*_IBM-943 です。
2. trans.dir ファイル内で変換テーブルを定義します。 *NewCodeSet* という名前の新規コード・セットを定義するためのフォーマットは、次のとおりです。

```
newcodeset IBM-943 newcodeset_IBM-943
```

3. 必要な場合は、trans.alias ファイルに別名を定義します。
4. 次の例のように、コロン・ファイルに入力コードとしてコード・セット名を追加します。

```
::Ti::[NewCodeSetName, ...]IBM-943, ...
```

Xwindows フォントと qprt コマンド

MBCS プリンター・バックエンドは、/usr/lib/X11/fonts ディレクトリーに定義されている Xwindows フォントを使用して、プリンターの ROM に保管されていない文字を印刷します。

qprt コマンドの **-F** フラグと **-I** (大文字の i) フラグは、プリンターに対して Xwindow フォントを指定します。これらの **qprt** コマンド・オプションのデフォルト値は、コロン・ファイル内で **_F** 属性と **_I** 属性の値として指定されます。

qprt -F フラグはフォントを指定します。 Xwindow フォントの絶対パス名、フォント別名、または Xwindow 論理機能記述 (XLFD) を **-F** フラグに使用できます。

-I フラグは、Xwindow フォントを検索するためのフォント・パスを指定し、**_I** 属性エントリーを作成します。**_I** 属性のコロン・ファイル・フォーマットは、次のとおりです。

```
::_I::/usr/lib/X11/fonts/JP,/usr/lib/X11/fonts
```

qprt -I コマンドを使用して別のフォント・パスを指定すると、プリンター・バックエンドは **_I** コロン・ファイルにリストされているデフォルト・パスではなく、指定されたフォント・パスを検索します。**-I** オプションにヌル値が指定されている場合、バックエンドはデフォルトの /usr/lib/X11/fonts ディレクトリーを想定します。

絶対パス名、フォント別名、または XLFD を使用して特定の Xwindows フォント・ファイルを指定するには、次のように入力します。

```
$ qprt -F '*-27-*-ibm_udcjp' foo.txt    /* XLFD names list */
$ qprt -F IBM_JPN17                     / * Font alias name */
```

次の例では、fonts.alias ファイルと fonts.dir ファイルを検索して、**qprt** コマンドの **-X** オプションに指定されたコード・セットに該当するフォントを見つけるように、MBCS プリンター・バックエンドに指示します。

変換テーブルの例

次に、変換テーブルの例を示します。

```
#include <fcntl.h>
struct trans_table              /*Translation Table Structure  */
{
       unsigned int reserv1;    /* Reserved                    */
       unsigned int f_cp;       /* From code point             */
       unsigned int reserv2;    /* Reserved                    */
       unsigned int t_cp;       /* To code point               */
};
/*
*Table to translate code points for input code set(NewCodeSet)
*to code points for the process code set(IBM-943).
*/
struct trans_table table[] =
{
        {0x0,0x81ca,0x0,0xfa54},{0x0,0x9e77,0x0,0x954f},\
        {0x0,0x9e8d,0x0,0x938e},
        /*  ....    */
        [0x0,0xfad0,0x0,0x8d56}
```

```
};
/* Write the table. Error processing not shown.  */
main()
{
        int ftrans;
        long hdsize = 32;        /* Header size               */
        long cpsize = 4;         /* Code point size           */
        long rsv1 = 0, rsv2 = 0; /* Reserved area             */
ftrans = open("usr/lib/lpd/pio/transJP/newcodeset_IBM-932",
               O_CREAT | O_WRONLY, 0664);
write(ftrans, "PIOSMBCSXLATE000", 16);
write(ftrans, &hdsize, sizeof(long));
write(ftrans, &cpsize, sizeof(long));
write(ftrans, &rsv1, sizeof(long));
write(ftrans, &rsv2, sizeof(long));
write(ftrans, table, sizeof(table));
return(0);
}
```

プリンター接続ファイル

接続ファイルは、プリンター接続の開発者が新しいプリンター接続をサポートする System Management Interface Tool (SMIT) 画面を作成するためのシンプルなインターフェースを提供します。

接続ファイルでは、各新規接続タイプが定義されます。接続ファイルには、各種印刷タスクを実行するのに使用される SMIT オブジェクト ID の名前が含まれています。接続タイプの名前は、10 文字に制限されています。接続ファイルについて詳しくは、以下を参照してください。

プリンター接続ファイルの SMIT インターフェース

接続ファイルは、SMIT メニューから SMIT オブジェクト ID への分岐を指示します。

それぞれの接続ファイルは、次に示す SMIT メニュー・オプションの一部またはすべてからの分岐を制御します。

- **Start a Print Job (印刷ジョブの開始)**
- **Add a Print Queue (印刷キューの追加)**
- **Add an Additional Printer to an Existing Queue (追加プリンターを既存のプリンター・キューに追加)**
- **Change/Show Print Queue Characteristics (印刷キューの特性の変更/表示)**
- **Change/Show Printer Connection Characteristics (プリンター接続の特性の変更/表示)**
- **Remove a Print Queue (印刷キューの除去)**
- **Pre-Processing Filters (事前処理フィルター)**

例えば、SMIT ダイアログ画面から「**Add a Print Queue (印刷キューの追加)**」メニュー・オプションを選択したとき、ユーザーが最初に要求される情報は、使用する接続タイプです。ユーザーが必要な接続タイプを選択すると、SMIT は接続タイプ・ファイルを検索して、分岐先の SMIT オブジェクト ID ファイルを発見します。

新規プリンター接続用の SMIT セレクターとダイアログは、新規接続タイプの印刷キューを追加、変更、および除去するダイアログを作成する必要があります。新規 SMIT ダイアログの名前は、接続ファイル内で指定されます。新規接続タイプ用のキューを作成、変更、または除去するときに、このファイル内のダイアログ名に自動的に分岐します。

プリンター接続ファイルの命名規則

接続ファイルには接続規則が使用されます。

接続ファイルの名前は、次の命名規則に従って指定する必要があります。

```
Attachment_type.attach
```

Attachment_type 文字列は、接続を識別する固有の文字列にする必要があります。接続ファイルはすべて、/usr/lib/lpd/pio/etc ディレクトリーに置いてください。次の接続ファイルが提供されています。

項目	説明
local.attach	ローカル・システム接続プリンターのファイル。
ascii.attach	ASCII 端末接続プリンターのファイル。
file.attach	ファイル出力の接続ファイル。
remote.attach	リモート印刷キューの接続ファイル。

プリンター接続ファイルの構造

接続ファイルは ASCII ファイルです。

接続ファイルの各行は、次のフォーマットを使用してフィールドを定義します。

```
FieldName = Value
```

接続ファイル内で、次のフィールド名には特別な意味があります。

- **description**
- **seq_num**
- **supported**
- **unsupported**

次のフィールド名は、SMIT セレクター ID を定義します。*Value* 変数には、SMIT セレクター ID を格納する必要があります。各フィールドのセレクター ID 値は、分岐のターゲットを指定します。SMIT フィールドは次のとおりです。

- **submit_job**
- **add_queue**
- **add_printer**
- **remove_queue**
- **printer_conn**
- **change_queue**
- **change_filters**

すべての接続ファイルに、**description**、**add_queue**、および **remove_queue** の各フィールドが含まれている必要があります。その他のフィールドはすべてオプションです。ヌル値が入ったフィールドは、そのフィールドが存在しない場合と同様に扱われます。接続ファイルの他の内容に制限はありません。

次の例では、接続ファイルの名前は term_serv.attach です。

```
description = term_serv.cat,1,3; Printer Attached to Terminal Server
seq_num = 2
submit_job = term_serv_start_job
add_queue = term_serv_add
add_printer = term_serv_printer
remove_queue = term_serv_remove
```

```
printer_conn = term_serv_printer_conn
change_queue = term_serv_change
change_filters = term_serv_change_filters
unsupported = ibm6252,ibm6262
```

プリンター接続ファイルのフィールド定義

フィールド定義とその接続タイプ・フィールド、フィールド値のフォーマット、およびフィールド値の実例を説明します。

フィールド定義およびタイプ

項目	説明

description	SMIT の「Attachment type (接続タイプ)」メニューに表示される説明文字列を指定します。SMIT の「Attachment type (接続タイプ)」メニューは、システム上でサポートされるすべての接続タイプをリストします。サポートされるタイプのリストに接続タイプを表示するには、このフィールドが必須です。
	
	説明フィールド値のフォーマットは、次のとおりです。
	Message_catalog,*Set*,*Message_#*; *DefaultTextString*
	
	Message_catalog、*Set*、および *Message_#* の値は、必須ではありません。例えば、次の 2 つのエントリー例では、SMIT に同じメニュー項目が作成されます。最初の例は、**term_serv.cat** メッセージ・カタログ、セット番号 **1**、およびメッセージ番号 **3** を使用します。メッセージが見つからない場合、SMIT は引用符で囲まれたデフォルト・テキストを使用します。2 番目の例では、メッセージ・カタログは指定せず、引用符で囲まれたメッセージがメニューに自動的に使用されます。
	`description = term_serv.cat,1,3; Printer Attached to Terminal Server`
	
	`description = Printer Attached to Terminal Server`

seq_num	SMIT の「Attachment Type (接続タイプ)」セレクター・メニューに特定の接続タイプを表示する順序を指定します。このフィールドが存在しない場合、接続は特定の順序を付けずに表示されます。例えば、接続をメニューの 2 番目に表示するには、次のように入力します。 `seq_num = 2`

supported/unsupported	接続によってサポートされている、またはサポートされていないプリンター・タイプのリストを定義します。**supported** フィールドの値は、接続タイプによってサポートされるプリンターのリストを SMIT ダイアログ画面に生成するために使用されます。これら 2 つのフィールドは同時には使用できません。
	
	supported フィールドと unsupported フィールドの値のフォーマットは、プリンター・タイプのコンマ区切りのリストです。例えば、サポートされるプリンターのリストから **ibm6252**、**ibm6262**、および **ibm4029** の各プリンターを除外するには、次のように入力します。
	`unsupported = ibm6252, ibm6262, ibm4029`
	
	使用可能なプリンター・タイプのリストに **hplj-3**、**hplj-3-si**、および **hplj-2** の各プリンターを表示するには、次のように入力します。
	`supported = hplj-3, hplj-3-si, hplj-2`

submit_job	印刷ジョブを開始するために分岐する先の SMIT セレクター ID の名前を指定します。このフィールドが存在しない場合は、**enq** ダイアログの値が使用されます。例えば、term_serv 接続タイプのキューが選択された場合に、「**Start a Print Job (印刷ジョブの開始)**」メニュー・オプションから **term_ser_start_job** セレクターに分岐するには、次のように入力します。 `submit_job = term_serv_start_job`
add_queue	印刷キューを追加するために分岐する先の SMIT セレクター ID の名前を指定します。例えば、「**Add a Print Queue (印刷キューの追加)**」メニュー・オプションから **term_serv_add** セレクター ID に分岐するには、次のように入力します。 `add_queue = term_serv_add`

フィールド定義およびタイプ

項目	説明
add_printer	既存キューにプリンターを追加するために分岐する先の SMIT セレクター ID の名前を指定します。機能上は、これによって既存キューにキュー・デバイスが追加されます。 SMIT の「**Existing Print Queue (既存の印刷キュー)**」メニュー・オプションから **term_serv_printer** セレクター ID に分岐するには、次のように入力します。 `add_printer = term_serv_printer`
remove_queue	印刷キューを除去するために分岐する先の SMIT セレクター ID の名前を指定します。「**Remove (除去)**」ダイアログ画面は、印刷キューの作成時に作成された他の任意のキュー、キュー・デバイス、仮想プリンター、およびプリンター・デバイスを除去します。「**Remove a Print Queue (印刷キューの除去)**」メニュー・オプションから **term_serv_remove** セレクター ID に分岐するには、次のように入力します。 `remove_queue = term_serv_remove`
printer_conn	印刷キューのプリンター接続特性を変更するために分岐する先の SMIT セレクター ID の名前を指定します。代表的なポート通信特性は、ボー・レート、パリティー、ストップ・ビットなどです。 SMIT の「**Printer Port Communication Characteristics (プリンター・ポート通信特性)**」メニュー・オプションから **term_serv_printer_conn** セレクター ID に分岐するには、次のように入力します。 `printer_conn = term_serv_printer_conn`
change_queue	プリンター・キューの特性を変更するために分岐する先の SMIT セレクター ID の名前を指定します。SMIT の「**Change/Show Print Queue Characteristics (印刷キューの特性の変更/表示)**」メニュー・オプションから **term_serv_change** セレクター ID に分岐するには、次のように入力します。 `change_queue = term_serv_change`
change_filters	印刷キューに対して定義された事前処理フィルターを変更するために分岐する先の SMIT セレクター ID の名前を指定します。SMIT の「**Change/Show Pre-Processing Filters (事前処理フィルターの変更/表示)**」メニュー・オプションから **term_serv_change_filters** セレクター ID に分岐するには、次のように入力します。 `change_filters = term_serv_change_filters`

プリンター・コロン・ファイルの limits フィールドのオペレーター

limits フィールドを使用すると、コロン・ファイルの作成者が、特定の属性のために作成される ODM オブジェクトのタイプを制御できます。

コロン・ファイルの **limits** フィールドには、次の 2 種類の情報があります。

- SMIT ダイアログ情報
- 妥当性検査情報

SMIT ダイアログ情報は、オブジェクト・データ・マネージャー (ODM) データベース内でコロン・ファイル属性を表現する SMIT オブジェクトを作成するために使用されます。オブジェクトは、ファイルの印刷、プリンターの設定、およびデフォルト・ジョブ特性のダイアログ画面に使用されます。

limits フィールドのために作成されるオブジェクトは、**sm_cmd_opt** オブジェクト・クラスに含まれます。**limits** フィールドを使用して、**sm_cmd_opt** オブジェクト・クラス内の次のフィールドを制御できます。

- **id_seq_num**
- **entry_type**
- **cmd_to_list_mode**
- **required**
- **op_type**
- **multi_select**

- **cmd_to_list_mode**
- **disp_values**
- **aix_values**
- **values_msg_file**
- **values_msg_get**
- **help_msg_id**
- **help_msg_loc**

これらの属性は、常に表示する、表示しない、または属性がパイプライン内で参照された場合のみ表示するように設定できます。これらのフィールドについて詳しくは、『"sm_cmd_opt (SMIT ダイアログ/セレクター・コマンド・オプション) オブジェクト・クラス』 (「 *プログラミングの一般概念: プログラムの作成およびデバッグ* の」) を参照してください。

妥当性検査情報は、コロン・ファイルを要約するとき、および印刷ジョブを実行依頼するときに、属性値を妥当性検査するために使用されます。

プリンター・コロン・ファイルの limits フィールドの内容

limits フィールドは、プリンター・コロン・ファイルの 4 番目のフィールドです。

コロン・ファイル属性のフォーマットは、次のとおりです。

```
Message_Catalog:Message_Number:Attr_Name:Limits:Value
```

limits フィールドの情報には、2 つの構成要素があります。最初の構成要素は、アクションを指定する単一文字のオペレーターです。文字値は、**C**、**D**、**E**、**F**、**G**、**H**、**I**、**L**、**M**、**Q**、**R**、**S**、**T**、または **V** のいずれかです。2 番目の構成要素はデータです。データに複数の文字がある場合は、大括弧 ([ ]) で囲む必要があります。

例えば、**limits** フィールドの内容が **'E#'** ならば、**sm_cmd_opt** オブジェクト・クラスの **entry_type** フィールドは、**#** に割り当てられた数値と等しくなります。 **'E#'** 値を指定した **entry_type** フィールドは、数値入力のみを受け入れます。

別の例として、**limits** フィールドの内容が **'[none,full,emulator=0,1,2]'** ならば、**sm_cmd_opt** オブジェクト・クラスには次の値が入ります。

フィールド名	値
disp_values	none, full, emulator
aix_values	0,1,2

limits フィールドのオペレーターは、SMIT で次のタイプの制御を行います。

- 属性の表示
- 属性を表現するフィールドの特性
- 属性の妥当性検査と補助操作のタイプ (例えば、ポップアップ・メニュー、リング・リスト)

例えば、**qprt** および **admvirprt** の両 SMIT ダイアログでは、次の規則が適用されます。

- 属性に対して **Dy** (**limits** オペレーター **D** と、「はい」を示す値 **y** の組み合わせ) を **limits** フィールドに指定すると、その属性は常に表示されます。
- 属性に対して **Dn** (**limits** オペレーター **D** と、「いいえ」を示す値 **n** の組み合わせ) を **limits** フィールドに指定すると、その属性はまったく表示されません。

qprt SMIT ダイアログでは、このほかに次の規則が適用されます。

- パイプライン内で参照される、プリンター・コロン・ファイル内で定義された _ (下線) から始まる属性すべて (例えば **_j** および _i) が表示されます。
- パイプライン内で参照される、プリンター・コロン・ファイル内で定義された C 組み合わせフラグから始まる属性すべて (例えば **Cs** および **Ca**) が表示されます。

admvirprt SMIT ダイアログに固有の規則は、次のとおりです。

- プリンター・コロン・ファイル内で定義された、_ (下線) または C 組み合わせフラグから始まる属性すべてが表示されます (属性の **limits** フィールドに **Dn** が指定されていない限り)。

プリンター・コロン・ファイルの limits フィールドのオペレーター

limits フィールドを使用すると、コロン・ファイルの作成者が、特定の属性のために作成される ODM オブジェクトのタイプを制御できます。

コロン・ファイルの **limits** フィールドには、次の 2 種類の情報があります。

- SMIT ダイアログ情報
- 妥当性検査情報

SMIT ダイアログ情報は、オブジェクト・データ・マネージャー (ODM) データベース内でコロン・ファイル属性を表現する SMIT オブジェクトを作成するために使用されます。オブジェクトは、ファイルの印刷、プリンターの設定、およびデフォルト・ジョブ特性のダイアログ画面に使用されます。

limits フィールドのために作成されるオブジェクトは、**sm_cmd_opt** オブジェクト・クラスに含まれます。**limits** フィールドを使用して、**sm_cmd_opt** オブジェクト・クラス内の次のフィールドを制御できます。

- **id_seq_num**
- **entry_type**
- **cmd_to_list_mode**
- **required**
- **op_type**
- **multi_select**
- **cmd_to_list_mode**
- **disp_values**
- **aix_values**
- **values_msg_file**
- **values_msg_get**
- **help_msg_id**
- **help_msg_loc**

これらの属性は、常に表示する、表示しない、または属性がパイプライン内で参照された場合のみ表示するように設定できます。これらのフィールドについて詳しくは、『"sm_cmd_opt (SMIT ダイアログ/セレクター・コマンド・オプション) オブジェクト・クラス』 (「 *プログラミングの一般概念: プログラムの作成およびデバッグ* の」) を参照してください。

妥当性検査情報は、コロン・ファイルを要約するとき、および印刷ジョブを実行依頼するときに、属性値を妥当性検査するために使用されます。

表示オペレーター:

表示オペレーターは、SMIT ダイアログ内で複数のフラグがどのように関係するか、フラグのオプションがどのように表示されるか、どのフラグとオプションが使用可能かを定義します。

項目　説明

C　相互依存フラグ (例えば、書体とピッチに影響するフラグ) をサポートするには、フラグの組み合わせを使用する必要があります。通常、SMIT ダイアログ・フィールドとコマンド・ライン・フラグの間にはただ 1 つの一致が存在します。組み合わせフラグ・オペレーターを使用すると、SMIT ダイアログ内の 1 つのフィールドによって、複数のコマンド・ライン・フラグを表現できます。個々のフラグではなく組み合わせフラグのみが表示されるように、参照されるフラグには SMIT ダイアログに対して非表示 (**Dn**) タイプのマークを付ける必要があります。

C オペレーターの構文は、次のとおりです。

```
C[xx,yy,...]
```

xx と *yy* の値は、フラグ属性です。**C** 属性を定義する場合は、SMIT ユーザーに表示されるポップアップ・リストを定義するために、**limits** フィールドに **R** リング・オペレーターも指定する必要があります。**R** オペレーターは、リストにあるオプションとコマンド・ライン・フラグのマッピングも定義します。

```
:111:Cs:C[_s,_p]R[Courier 10, Prestige 12= -s Courier
   -p10, -s Prestige -p12]):-s %I_s -p %I_p
:999:_s:Dn:Courier
:222:_p:Dn:10
```

この例では、**C** オペレーターは **-s** フラグと **-p** フラグが組み合わせ属性であることを定義しています。**R** リングの定義により、ポップアップ・メニューから「Courier 10」オプションが選択されると、コマンド・ラインのフラグは **-s Courier -p10** になります。 **-s %I_s -p%I_p** 属性値は、SMIT ダイアログが作成されると解決され、デフォルトとして表示されるリング内の項目を決定します。

D　表示モードを指定します。値が **y** ならば、オブジェクトは **sm_cmd_opt** オブジェクト・クラス内に作成されます。値が **n** ならば、オブジェクトは作成されません。**D** オペレーターにより、プログラマーは特定のフラグが SMIT に表示されないように抑止できます。このオペレーターを指定しない場合は、フラグが入力パイプライン内で参照されるとオブジェクトが作成されます。

S　**sm_cmd_opt** オブジェクト・クラスの **id_seq_num** フィールドのシーケンス番号を指定します。シーケンス番号は、ダイアログ画面にある他の項目を基準に、項目の位置を制御します。**S** オペレーターを指定しないと、コロン・ファイル内で検出された順序に従って、ダイアログの項目に ID 番号 100 から始まる番号が付けられます。

S オペレーターの値は、最大長 16 文字の文字列です。例えば、次の S オペレーター・エントリーは、位置 100 に項目を配置します。

```
:100:_1:S[100]:60
```

フィールド特性オペレーター:

フィールド特性オペレーターがいくつかあります。

項目　説明

E　**sm_cmd_opt** オブジェクトの **entry_type** フィールドを制御します。 **E** オプションに指定できる値は、次のとおりです。

- **#**　数値入力が許可されることを示します。
- **f**　ファイル入力が許可されることを示します。有効なファイル名を指定する必要があります。
- **n**　入力が許可されないことを示します。このフィールドはタイプ入力を受け入れることができません。
- **r**　英数字入力が許可されることを示します。
- **t**　テキスト入力が許可されることを示します。
- **x**　16 進入力が許可されることを示します。

SMIT ダイアログ・フィールドへの数値入力を許可するには、次のように入力します。

```
:100:_L:E#:60
```

項目　説明

Q　**sm_cmd_opt** オブジェクトの **required** フィールドの値を制御します。 **required** フィールドは、このダイアログの **cmd_to_exec** コマンドにフィールド値を送るかどうかを決定します。

単一の文字が値のタイプを指定します。デフォルト値は **n** で、これはフィールド値が変更された場合のみ **sm_cmd_opt** オブジェクトのフラグと値を渡すことを示します。**required** フィールドに指定できる値は、次のとおりです。

n　番号を表示します。最初に表示された値をユーザーが変更しない限り、フラグを送りません。**n** 値はデフォルトです。

y　「はい」を示します。**prefix** フィールドと、**entry** フィールドの値 (ヌルであっても) を常に送ります。

+　**prefix** フィールドと値を送ります。値は非ブランクの 1 文字を指定する必要があります。

?　値がヌルである場合を除き、常に **prefix** フィールドと値を送ります。

prefix フィールドと、**entry** フィールドの値が常に **cmd_to_exec** に送られるようにするには、次のように入力します。

```
:100:_L:Qy:60
```

補助操作と妥当性検査のオペレーター:

SMIT ダイアログの補助操作は、ユーザーに要求するリストと入力のタイプを決定します。

SMIT ダイアログ定義の補助操作は、ユーザーに要求するリストと入力のタイプを決定します。ダイアログ内で使用可能なリストのタイプは、リスト、複数選択リスト、範囲リスト、オプション・リング、または複数選択オプション・リングです。属性によって使用される補助操作のタイプを指定する **limits** フィールド・オペレーターは、**L**、**M**、**G**、および **R** です。

一度にサポートされる補助操作のタイプはただ 1 つです。デフォルトは **op_type=n** です。**n** の値は、フィールドに対して補助操作が許可されないことを示します。

次の補助操作が使用可能です。

項目　説明

F　**sm_cmd_opt** オブジェクトの **cmd_to_list_mode** フィールドの制御を可能にします。**cmd_to_list_mode** フィールドは、リストからどれだけの項目を使用するかを指定します。リストは、**cmd_to_list** フィールド・オブジェクトに指定されたコマンドによって作成されます。例えば、**cmd_to_list** フィールドが次のリストを生成したとします。

```
60 (6 行/インチ)
80 (8 行/インチ)
66
```

F オペレーターに指定できる値は、次のとおりです。

a　すべてのフィールドを取得します。これはデフォルト値です。

1　最初のフィールドを取得します。

2　2 番目のフィールドを取得します。

リストから最初のフィールドを取得するように SMIT に指示するには、次のように入力します。

```
:100:_l:F1:60
```

G　範囲リストを指定します。**G** オペレーターは、**cmd_to_list_mode** に r 値を指定します。r 値は、**cmd_to_list** フィールドによって表示される情報が、リストではなく情報の範囲であることを指定します。

範囲に対しては、妥当性検査が常に実行されます。範囲リストのデータの形式は、*x..y* (1..30) または *..y* (..30) または *x..* (1..) です。ここで、 *x* と *y* は整数で、範囲の上限と下限を指定します。妥当性検査により、属性値が指定の範囲内にあることが確認されます。負の整数も指定できますが、上限 (*y* 値) は下限 (*x* 値) より大か等しいことが必要です。50 から 100 の範囲でフィールド・リスト操作を行うように指定するには、次のように入力します。

```
:100:_l:G[50..100]:60
```

項目　説明

H　対応する属性のヘルプ・テキストにするメッセージ・カタログ仕様を指定します。メッセージ・カタログ仕様には、メッセージ・カタログ名、セット番号、およびメッセージ番号が含まれます。ヘルプ・テキストは、ヘルプが割り当てられた属性を使用する SMIT ダイアログ内で使用されます。

フラグ **-b** に **pioattrl cat** ダイアログからのヘルプを割り当てるには、次のように入力します。

```
:100:_b:H[pioattr1.cat,5,123]:60
```

I　対応する属性のヘルプ・テキストにする資料仕様を指定します。資料仕様には、**sm_cmd_opt** SMIT オブジェクト・クラスの **help_msg_id**、**help_msg_base**、および **help_msg_book** の各フィールドの値が含まれます。ヘルプ・テキストは、ヘルプが割り当てられた属性を使用する SMIT ダイアログ内で使用されます。

フラグ **-b** に資料仕様からのヘルプを割り当てるには、次のように入力します。

```
:100:_b:I[100145]:60
```

L　ユーザーが F4 を選択したときにポップアップ・リストが表示されるように指定します。ポップアップ・リストにより、ユーザーは与えられたオプションのリストからただ 1 つのオプションを選択できます。ポップアップ・リストは、**cmd_to_list** フィールド値から構成されます。ポップアップ・メニューの **op_type** フィールド値は、**l** (小文字の L) です。

タイプ入力によるユーザー入力が禁止されている場合のみ、妥当性検査が行われます。直接ユーザー入力を許可しないフィールドのエントリー・タイプは、**n** です。 **cmd_to_list** フィールドは、改行で区切られたリストを返します。このリストからの値が、属性値と比較されます。

L オペレーターに指定できる値は、**cmd_to_list** フィールドのシェル・コマンド・ストリングです。コマンドから生成されるリストは、改行文字で区切られた出力値のリストです。次に例を示します。

```
:100:_l:L[print "50\n55\n60\n65"]:60
```

M　与えられたオプションのリストからユーザーが複数の値を選択できる、複数選択リストを指定します。**multi-select** フィールドを **m** の値に設定する必要がある点を除いては、**M** オペレーターの動作は **L** オペレーター・リストとまったく同じです。

複数選択リストのオペレーター・エントリーの例を次に示します。

```
:100:_l:M[print "50\n55\n60\n65"]:60
```

R　リストのオプション・リング・タイプを指定します。**op_type** フィールドは **r** に設定されます。リング・リストが通常のリストと異なる点は、ユーザーがタブ (順方向) キーまたは後退タブ (逆方向) キーを押すことによって、リスト・オプションの表示を継続できることです。リング・リストがオプションの末尾に到達すると、リストの先頭に循環します。リング・リストは、順方向または逆方向に循環します。 F4 キーを押すと、リング・リストは通常のリストになります。

オプションのリング・オペレーターは、**disp_values**、**aix_values**、**values_msg_file**、**values_msg_set**、および **value_smg_id** の各フィールドを制御できます。メッセージ ID なし、メッセージ ID のみ、メッセージ・セットと ID、またはメッセージ・セット、カタログ、および ID が、リング・オプション・リスト内で有効です。

入力タイプ値が n に設定されて、ユーザーによる直接入力が禁止されている場合には、妥当性検査が実行されます。リングの値はハードコーディングされており、独立の値であるか、または基本オペレーティング・システム値にマップされます。

独立の値の例としては、可能なボー・レートのリストがあります (**'1200,2400,9600,19200'**)。この場合、レート値そのものがフラグ引数として使用されます。

マップされた値の例は、プリンター上で使用される用紙ドロワーを指定する属性です。この例では、可能な 3 つの表示値は下段ドロワー、上段ドロワー、および封筒給紙機構です。これらの可能な値は、基本オペレーティング・システムのフラグ・オペランド **'0,1,2'** にマップされます。実行されるコマンドには、基本オペレーティング・システムの値が渡されます。

妥当性検査により、属性値がハードコーディングされた値のセット内にあることが検証されます。次の例では、数種類のオプション・リング・リストを示します。

```
:100:_l:R[0,1,2]:0
:100:_l:R[none,full,emulator=0,1,2]:0
:100:_l:R[;none,full,emulator=0,1,2]:0
:100:_l:R[21,none,full,emulator=0,1,2]:0
:100:_l:R[1,21;none,full,emulator-0,1,2]:0
:100:_l:R[pioattr9.cat,1,21;none,full,emulator=0,1,2]:0
```

項目 **説明**

T ポップアップ・リストからの複数選択を許可し、**R** オペレーターと同じ動作をします。複数選択フィールドは、**m** と同じです。

ポップアップ・メニューからの複数選択を許可するには、次のように入力します。

```
:100:_l:T[none,full,emulator=0,1,2]:0
```

V 属性の追加検証を指定します。**V** オペレーターは、属性に対する ODM スタンザの作成方法に影響しません。**V** オペレーターによって指定されるデータは、コロン・ファイル・タイプ・コード (**%** オペレーター) です。**%** オペレーターは妥当性検査を実行します。コロン・ファイル・コードは 1 つの値に解決されます。値は 0 またはゼロ以外のどちらかです。解決された値が 0 ならば、属性値は有効です。値がゼロ以外ならば、属性値は無効です。

_l の値が 0 から 100 の範囲内にあることを検証するには、次のように入力します。

```
:100:_l:V[%?%G_l%{100}%>%t1%e%?%G_l%{0}%<%t1%e0%;%;]:60
```

プリンター・コロン・ファイルを使用したプリンターの追加:

プリンター・コロン・ファイルを使用してプリンターを追加できます。

プリンター・コロン・ファイルを使用してプリンターを追加するには、その前に次の前提条件が満たされていることを確認してください。

- プリンターはシステムに物理的に接続されている必要があります。
- 追加するプリンターと、現在サポートされているプリンターの類似点と相違点を比較します。サポートされるプリンターのリストを確認するには、**lsdev** (デバイスのリスト) コマンドを使用します。
- プリンター・コロン・ファイルとそのフォーマットを理解している必要があります。 79 ページの『プリンター・コロン・ファイルの規則』 は、コロン・ファイル内のプリンターと属性の名前と値に関する規則のリストです。

1. 新しいプリンターが最も近くエミュレートしている、サポートされるプリンターを選択します。必要に応じて、プリンターの資料を参照してください。
2. **mkvirprt** コマンドを使用し、mkvirprt と入力して仮想プリンター定義を作成します。 プロンプトに応答して、選択したプリンター・タイプを指定します。すべてのデバイス名とキュー名は英字から始める必要があるので注意してください。
3. **lsvirprt** コマンドを使用して、属性値と説明を検討します。後のステップでこれらの値を比較する必要があるので、次のように入力して、出力を一時ファイルにリダイレクトします。

   ```
   lsvirprt -q QueueName -d QueueDeviceName > tempfile
   ```

4. **lsvirprt** コマンドからの出力を表示して (別のウィンドウに、またはハードコピー印刷出力として)、属性の説明と値を、追加するプリンターのものと比較します。行う変更の内容を決定します。
5. 事前定義データベース・ディレクトリー (/usr/lib/lpd/pio/predef) からカスタマイズ・データベース・ディレクトリー (/var/spool/lpd/pio/@local/custom) に、プリンター・コロン・ファイルをコピーします。
6. 『Adding a New Printer Type"』 (「 *Kernel Extensions and Device Support Programming Concepts* の」) の説明に従って、コロン・ファイルの属性値を変更します。この属性値には、プリンター・タイプ (**mt** 属性)、プリンター記述 (**mL** 属性)、プリンター・エミュレーション・モード (**ep** 属性) などがあります。
7. **chvirprt** コマンドを実行し、属性値を指定せずにキュー名とキュー・デバイス名を指定します。 このアクションにより、要約された仮想プリンター定義が作成されます。
8. 新しく定義したプリンターが正しく印刷を行うことを確認します。
9. 事前定義の仮想プリンター定義を作成する場合は、**piopredef** コマンドを使用して作成します。

項目	説明
%Sxx	**xx** 属性の現行文字列値へのポインターをスタックにプッシュします。文字列ポインターに対して実行できる唯一の演算は、**%=** を使用して、文字列をそのスタック上にポインターがある別の文字列と比較することです。
%Lxx	**xx** 固定文字列または可変文字列の長さをスタックにプッシュします。例えば、属性 **ss** の値が **IJKLMN** ならば、シーケンス **ABC%Lss%dDEFG** は文字列 **ABC6DEFG** を生成します。ただし、**xx** が **%Lxx** シーケンスを含む属性である場合、長さは **%Lxx** が検出されたときに既に構成されている部分の文字列の長さです。例えば、**st** 属性の値が **ABC%Lst%dDEFG** ならば、属性 **st** に対して構成される文字列は **ABC3DEFG** です。

追加したプリンターのサポートのインストール:

各プリンターのサポートは、別個にインストール可能なパッケージとして提供されます。

ご使用のマシンにサポートが既にインストールされているプリンターのリストを表示するには、次のように入力します。

```
smit lssprt
```

その他のプリンターのサポートをインストールするには、次のように入力します。

```
smit printerinst
```

ご使用のプリンターがサポートされていない場合は、ご使用のプリンターと類似した機能をもつサポートされるプリンターとして構成できます。そうでなければ、ご使用のプリンターを汎用プリンターとして構成できます。このためには、次のいずれかを行います。

- プリンターの印刷キューを追加する際には、プリンター製造メーカーまたはプリンター・モデルとして「**Other (その他)**」を選択します。
- プリンターのプリンター・デバイス定義を追加する際には、「**Other serial printer (他のシリアル接続プリンター)**」または「**Other parallel printer (他のパラレル・ポート接続プリンター)**」を選択します。

サポートされるプリンター:

次に、サポートされるプリンターのリストを示します。

- Bull Compuprint 4/51
- Bull Compuprint 4/54
- Bull Compuprint 914
- Bull Compuprint 914 N
- Bull Compuprint 922
- Bull Compuprint 923
- Bull Compuprint 924
- Bull Compuprint 924 N
- Bull Compuprint 956
- Bull Compuprint 970
- Bull Compuprint 1070
- Bull Compuprint PageMaster 200
- Bull Compuprint PageMaster 201
- Bull Compuprint PageMaster 411
- Bull Compuprint PageMaster 413
- Bull Compuprint PageMaster 422

- Bull Compuprint PageMaster 721
- Bull Compuprint PageMaster 815
- Bull Compuprint PageMaster 825
- Bull Compuprint PageMaster 1015
- Bull Compuprint PageMaster 1021
- Bull Compuprint PageMaster 1025
- Bull Compuprint PageMaster 1625
- Bull PR-88
- Bull PR-88 VFU Handling
- Bull PR-90
- 165 ページの『Canon LASER SHOT LBP-B404PS/Lite』
- 165 ページの『Canon LASER SHOT LBP-B406S/D/E/G、A404/E、A304E』
- 165 ページの『Dataproducts LZR 2665 レーザー・プリンター』
- Dataproducts BP2000 ライン・プリンター
- HP 2500C カラー・プリンター
- 166 ページの『Hewlett-Packard LaserJets II、III、IIISi、3005、4、4Si、4Plus、 4V、4000、5200、5Si/5Si MX、5Si Mopier、4700 Color、8000 Color、および 8500 Color』
- HP LaserJet カラー
- HP LaserJet 5000 D640 プリンター
- HP LaserJet 8100 プリンター
- HP Color LaserJet 4500 プリンター
- 152 ページの『IBM パーソナル・プリンター II モデル 2380、2381、2390、2391、2380-2、2381-2、2390-2、2391-2』
- IBM 3112 ページ印刷装置
- IBM 3116 ページ印刷装置
- 152 ページの『IBM 3812 モデル 2 ページ印刷装置』
- 153 ページの『IBM 3816 ページ印刷装置』
- 153 ページの『IBM 4019 ページ印刷装置および 4029 ページ印刷装置』
- 154 ページの『IBM 4037 および IBM 4039 レーザー・プリンター』
- IBM 4070 インクジェット・プリンター
- 155 ページの『IBM 4072 ExecJet プリンター』
- IBM 4076 インクジェット・プリンター
- 155 ページの『IBM 4076 インクジェット・プリンター』
- IBM 4079 カラー・ジェットプリンター
- IBM 4201 モデル 2 プロプリンター II
- IBM 4202 モデル 2 プロプリンター II XL
- 155 ページの『IBM Proprinter モデル 4201-3、4202-3、4207-2、4208-2』
- 155 ページの『IBM 4208-502、IBM 5572-B02、IBM 5573-H02、および IBM 5579-H02/K02』
- IBM 4212 プロプリンター 24P
- 156 ページの『IBM 4216-031 型ページ印刷装置』

- 156 ページの『IBM 4216-510 および IBM 5327-011』
- IBM 4224 プリンター、モデル 301、302、3C2、3E3
- IBM 4226 プリンター
- 156 ページの『IBM 4234 印刷装置』
- 157 ページの『IBM 5202 クワイエット・ライター III』
- 157 ページの『IBM 5204 印刷装置』
- 157 ページの『IBM 5575-B02/F02/H02 および IBM 5577-B02/F02/FU2/G02/H02/J02/K02』
- 157 ページの『IBM 5584-G02/H02、IBM 5585-H01、IBM 5587-G01/H01、および IBM 5589-H01』
- IBM 6180 カラー・プロッター
- IBM 6182 オート・フィード・カラー・プロッター
- IBM 6184 カラー・プロッター
- IBM 6185-1 カラー・プロッター
- IBM 6185-2 カラー・プロッター
- IBM 6186 カラー・プロッター
- 157 ページの『IBM 6252 印刷装置』
- IBM 7372 カラー・プロッター
- IBM B02/F02/H02
- IBM B02/F02/FU2/G02/H02/J02/K02
- 161 ページの『IBM InfoPrint 20』
- 162 ページの『IBM InfoPrint 32 プリンター』
- 164 ページの『IBM InfoPrint 40 プリンター』
- 158 ページの『IBM ネットワーク・カラー・プリンター』
- 159 ページの『IBM ネットワーク・プリンター 12、17、および 24』
- Lexmark 4039 Plus レーザー・プリンター
- Lexmark 4079 Color JetPrinter Plus
- Lexmark 4227 フォーム・プリンター
- Lexmark ExecJet IIc
- 173 ページの『Lexmark Optra レーザー・プリンター』
- 175 ページの『Lexmark Optra Plus レーザー・プリンター』
- 177 ページの『Lexmark Optra Color 1200 プリンター』
- 179 ページの『Lexmark Optra Color 40 プリンター』
- 181 ページの『Lexmark Optra Color 45 プリンター』
- 183 ページの『Lexmark Optra K 1220 プリンター』
- 185 ページの『Lexmark Optra C カラー・レーザー・プリンター』
- 187 ページの『Lexmark Optra E レーザー・プリンター』
- 189 ページの『Lexmark Optra N レーザー・プリンター』
- 192 ページの『Lexmark Optra E310 レーザー・プリンター』
- 194 ページの『Lexmark Optra M410 レーザー・プリンター』
- 196 ページの『Lexmark Optra Se レーザー・プリンター』
- 199 ページの『Lexmark Optra T レーザー・プリンター・ファミリー』

- 202 ページの『Lexmark Optra W810 レーザー・プリンター』
- 205 ページの『Lexmark Plus プリンター・モデル 2380-3、2381-3、2390-3、2391-3』
- Lexmark ValueWriter 600
- 207 ページの『OKI MICROLINE 801PS/+F、801PSII/+F、800PSIILT』
- 207 ページの『Printronix P9012 ライン・プリンター』
- 207 ページの『QMS ColorScript 100 モデル 20 プリンター』
- 207 ページの『Texas Instruments OmniLaser 2115 ページ・プリンター』

パススルー・モード:

仮想プリンターとプリンター・デバイス・ドライバーは両方とも、パススルー・モードまたは非パススルー・モードのどちらかで動作 (または機能) できます。

パススルー・モードでは、データ・ストリームはバイトごとに、変更されずにプリンターに「パススルー」されます。ジョブに対して選択された操作のモードによって、データ・ストリームの処理方法、またデータ・ストリームが処理されるかどうかが決まります。2 つのモードの違い、それぞれのモードが有効になる条件、およびモードを変更できるかどうかを理解することが重要です。

プリンター・デバイス・ドライバーのパススルー・モード:

プリンター・デバイス・ドライバー自体、例えば `/dev/lp0` は、デフォルトでは非パススルー・モードで動作します。

ユーザーは、**splp** コマンドを実行して、`/dev/lp0` の操作規則を照会または変更できます。例えば、IBM 4029 ページ印刷装置が **lp0** として定義されているシステム上で、コマンド `splp lp0` を実行した結果を次に示します。結果は、**TERM** 環境変数に指定された表示エレメントに出力されます。

```
device = /dev/lp0      (+ yes     ! no)
CURRENT FORMATTING PARAMETERS (ignored by qprt, lpr, and lp commands)
Note: -p + causes the other formatting parameters to be ignored.

-p !    pass-through?                  -c +    send carriage returns?

-l 64   page length (lines)            -n +    send line feeds?

-w 80   page width (columns)           -r +    carriage rtn after line fee?

-i 0    indentation (columns)          -t +    suppress tab expansion?

-W !    wrap long lines?               -b +    send backspaces?

-C !    convert to upper case?         -f +    send form feeds?

CURRENT ERROR PROCESSING PARAMETERS
-T 600  timeout value (seconds)        -e !    return on error?

CURRENT SERIAL INTERFACE PARAMETERS
-B 19200baud rate                      -s 8    character size (bits)

-N !    enable parity?                 -S !    two stop bits?

-P !    odd parity?
```

-p パラメーターは、プリンター・デバイス・ドライバー `/dev/lp0` の操作のモードがデフォルトでパススルー・モードになるかどうかを決定します。操作のモードは、個々のデータ・ストリームごとに指定変更できます。デフォルトでは、**-p** パラメーターの値は **!** (または **no**) です。 **-p** パラメーターには、「操作のモードをパススルー・モードにするかどうか」という質問の答えを指定することに注意してください。

-p パラメーターの値が **!** ならば、デバイス・ドライバーはデータ・ストリームの処理時に、リストされている他のパラメーターをすべて反映します。同様に、**-p** パラメーターの値が **+** (または **yes**) ならば、データ・ストリームの処理時に、他のパラメーターはすべて無視されます。

splp コマンドを使用して、プリンター・デバイス・ドライバーのパラメーター値を変更しても、スプーラーの動作に影響はありません。 **splp** コマンドは、**cat** などのコマンドを使用してデバイス・ドライバーに直接アクセスする (スプーラーをバイパスして) 場合に、これらのコマンドに影響を及ぼします。例えば、次のコマンドを考えます。

```
cat /etc/motd > /dev/lp0
```

このコマンドは、/dev/lp0 を開き、「今日のメッセージ」の内容をプリンターに直接書き込みます。プリンターの出力のフォーマットは、次の例のようになります。

```
This is a test version of /etc/motd, used to demonstrate
what happens when a printer device driver, such as
/dev/lp0, is placed into or taken out of passthru mode.
Printers will print either exactly what they are sent,
if you set the job conditions up correctly, or, on the
most current printers, you may be able to direct the
printer to perform certain mappings for you.

There are no carriage returns in
this file, and the only blank line occurs
immediate before this one.
```

-r パラメーターの指示により、**-p** の値が **!** ならば、それぞれの改行 (LF) が改行と復帰にマップされることに注意してください。これが必要である理由は、ほとんどの UNIX ベースのオペレーティング・システムが改行のみを使用するためです。DOS や OS/2 などのオペレーティング・システムとは異なり、UNIX ベースのオペレーティング・システムでは、改行は復帰を暗黙に示しています。これはテキスト・エディターのような環境では正しく動作しますが、プリンター上では正しく動作しません。プリンターは送られたデータのみを印刷します。例えば、次の 2 つのコマンドを発行するとします。

```
splp -p+ lp0

cat /etc/motd > /dev/lp0
```

この結果、プリンター上では次のような出力が得られます。

```
This is a test version of /etc/motd, used to demonstrate
what happens when a printer device driver, such as
/dev/lp0, is placed into or taken out of passthru mode.
Printers will print either exactly what they are sent,
if you set the job conditions up correctly, or, on the
most current printers, you may be able to direct the
printer to perform certain mappings for you.

There are no carriage returns in
this file, and the only blank line occurs
immediately before this one.
```

最初の例では、デバイス・ドライバーの設定値がすべて反映されています。特に、改行から改行復帰へのマッピングがオンになっています。物理プリンターに文字を書き込む際に、デバイス・ドライバーはそれぞれの改行の後に復帰を送ります。また、ページ幅の設定値を反映します。

2 番目の例では、デバイス・ドライバーは単に /etc/motd の 1 バイト文字をそれぞれ物理プリンターに書き込む処理のみを行い、データ・ストリームのマッピングやその他の変更は行いません。/etc/motd の最初の文が終わると、改行によって印刷ヘッドは 1 行下にまっすぐ降ろされ、印刷ヘッドを左マージンに戻す復帰は行われません。「printer」という語の最初の 4 文字、**prin** が印刷されます。その時点で、デバイ

ス・ドライバーではなくプリンター自体が、右マージンに達したことを判定して復帰を印刷し、左マージンに印刷ヘッドを戻します。印刷はデータ・ストリーム内の次の文字から継続します。

2 番目の例では、プリンターのリセット・ボタンが押されるまで、ジョブの印刷自体が行われません。この理由は、プリンターがページを自動的に排出するために十分なデータ (文字) を受け取っておらず、さらにページの排出を引き起こす用紙送りがプリンターに送られていなかったからです。デバイス・ドライバーの **-f** パラメーターが無視されています。

フォーマッター・フィルターのパススルー・モード:

ジョブをスプーラーに実行依頼した後、ジョブは最終的にフォーマッター・フィルターに渡されて処理され、プリンター・デバイス・ドライバーに配送されます。フォーマッター・フィルターは、常にプリンター・デバイス・ドライバーをパススルー・モードで開きます。

ジョブをスプーラーに実行依頼した後、ジョブは最終的にフォーマッター・フィルターに渡されて処理され、プリンター・デバイス・ドライバーに配送されます。フォーマッター・フィルターは、常にプリンター・デバイス・ドライバーをパススルー・モードで開きます。スプーラーに対して実行依頼されたジョブは、プリンター・デバイス・ドライバーに直接送られたデータ・ストリームとは異なり、常にフォーマッター・フィルターによって処理または変更され、プリンター・デバイス・ドライバーによって処理されることはありません。

プリンター・デバイス・ドライバーと同様に、フォーマッター・フィルターにもパススルーと非パススルーの 2 つの操作モードがあります。同じく、ジョブに対して選択された操作のモードによって、データ・ストリームの処理方法、またデータ・ストリームが処理されるかどうかが決まります。

仮想プリンター定義 (要約されたコロン・ファイル) の **_d** 属性は、その仮想プリンターに関連したキューの入力データ・ストリーム・タイプを指定します。また、仮想プリンター定義は、その入力データ・ストリーム・タイプのフォーマッター・フィルターも指定します。ジョブを処理するためにフォーマッター・フィルターが呼び出されると、フォーマッター・フィルターを実行するプロセス (**pioformat**) が **_d** 属性の値を検査し、フォーマッター・フィルターをパススルー・モードで呼び出すかどうかを決定します。パススルー・モードが選択されると、フォーマッター・フィルターは passthru() サブルーチンを使用して入力ストリームを読み取り、変更せずにプリンター・デバイス・ドライバーに送ります。パススルー・モードが選択されなければ、フォーマッター・フィルターは lineout サブルーチンを使用して、入力データ・ストリームを 1 行ずつ処理します。どちらの場合も、プリンター・デバイス・ドライバーは書き込み用にパススルー・モードで開かれ、出力データ・ストリームに対して処理を行いません。

PostScript などの入力データ・ストリームは、本質的にパススルーであることに注意してください。つまり、処理はプリンター上の PostScript インタープリター・ハードウェアによって行われます。

splp コマンドを使用して表示または変更できるプリンター・デバイス・ドライバー・パラメーターのほとんどは、フォーマッター・フィルターにも存在します。これらのパラメーターは、仮想プリンター用のコロン・ファイルの要約バージョンに格納されています。例えば、IBM 4029 ページ印刷装置に対する ASCII キューの場合、プリンター・デバイス・ドライバーのパラメーターと、コロン・ファイル内の対応するパラメーターの間のマッピングは次のとおりです。

デバイス・ドライバー・パラメーター	コロン・ファイル内の対応パラメーター	
パススルー	-p	_d
ページ長 (行)	-l	_l
ページ幅 (列)	-w	_w
インデント (列)	-i	_i
長い行の折り返し	-W	_L
大文字への変換	-C	N/A
復帰文字を送信する	-c	_x
改行を送信する	-n	_x
改行後の復帰	-r	_x
タブ拡張の抑止	-t	N/A
後退文字を送信する	-b	N/A
用紙送りの送信	-f	_Z

右側の列のパラメーター値は、仮想プリンター定義に永続的に設定できます。また、**qprt** コマンドまたは **enq** コマンドに特定のフラグを使用することによって、ジョブの実行依頼時に指定変更することもできます。

仮想プリンターの定義:

IBM 4029 ページ印刷装置は、4 種類のデータ・ストリームをサポートします。

root ユーザーは、**mkvirprt** コマンドを使用して、4 つのデータ・ストリーム・タイプそれぞれに対してキューと仮想プリンター定義の両方を作成できます。さらに、root ユーザーは **lsvirprt** コマンドを使用して、仮想プリンター定義の基礎であるコロン・ファイルを表示し、変更できます。それぞれのタイプのキューが定義されているシステムに対して **lsvirprt** コマンドを実行すると、次のリストと照会が表示されます (キュー名とデバイスは、キューの作成時に root ユーザーによって選択されます)。

```
 No. Queue             Device   Description

  1   asc               lxx      4029  (IBM ASCII)
  2   gl                lxx      4029  (Plotter Emulation)
  3   pcl               lxx      4029  (HP LaserJet II Emulation)
  4   ps                lxx      4029  (PostScript)

Enter number from list above (press Enter to terminate):  ->
```

このリストから、root ユーザーは表示、フォーマット設定、または変更の対象である仮想プリンターに対応する番号を入力します。次のメッセージとプロンプトが表示されます。

```
To LIST attributes, enter AttributeName1 ... (* for all attributes)
To CHANGE an attribute value, enter AttributeName=NewValue
To FORMAT and EDIT an attribute value, enter AttributeName~v
To EDIT the attribute file, enter ~v
To terminate, press Enter:
```

この時点でいくつかのオプションを選択でき、その 1 つは Enter を押して **lsvirprt** コマンドを終了することです。その他のオプションは、次のとおりです。

- アスタリスク (*) を入力すると、コロン・ファイル内の属性すべてのリストが、メッセージ・カタログからのテキスト記述とともに表示されます。
- 属性の名前を入力すると、その属性のみがメッセージ・カタログからのテキスト記述とともに表示されます。
- 属性の名前、=、および値を入力すると、属性にその値が割り当てられます。

- **~v** を入力すると、未加工のコロン・ファイルを編集する **vi** セッションが開きます。
- 属性の名前を入力し、直後に (ブランク・スペースを空けずに) **~v** を入力すると、見やすいフォーマットでその属性を編集する **vi** セッションが開きます。

これら 5 つのオプションそれぞれについて、**asc** キューおよび関連した仮想プリンター定義のコンテキストで、基礎となるコロン・ファイルとともに説明します。

アスタリスク (*) を入力して Enter を押すと、次の内容が表示されます。

```
Name    Description                                          Value
__FLG   VALUES THAT MAY BE OVERRIDDEN WITH FLAGS ON THE
          COMMAND LINE
_0      (not used)
_1      (not used)
_2      (not used)
_3      (not used)
_4      (not used)
_5      (not used)
_6      (not used)
_7      (not used)
_8      (not used)
_9      (not used)
_A      stderr returned? 0: no;  1: yes, & pipelines;  2:    1
          yes, & values, pipelines
_E      Double-High Print. (!: no;  +: yes)
_F      (not used) Font file name
_G      Page format (!: use only printable page   +: use     !
          entire addressable area)
_H      Name To Replace Host Name On Burst Page
_I      Font ID (overrides pitch and type style)
_J      Restore the Printer at the End of the Print Job?     +
          (!: no;  +: yes)
_K      (not used)
_L      Wrap Long Lines (!: no;  +: yes)                     +
_O      Type of Input Paper Handling (1: manual, 3:          3
          sheetfeed)
_Q      Paper or Envelope Size For the Paper Source          %IwQ
                    Selected By the -O and -u Flag Values (Refer to
                    the s0, s1, s2, s3, and s4 attributes); Default
                    value: %IwQ
_S      High speed printing
_U      Unidirectional printing
_V      Vertical printing
_W      Double-Wide Print (!: no;  +: yes)                   !
_X      Code Page Name For Print Data Stream (file with      IBM-850
                    same name in dir. "d1")
_Y      Duplex Output (0: Simplex  1: Duplex Long-Edge  2:   0
           Duplex Short-Edge)
_Z      Issue Form Feed Between Copies & At Job End (!:      +
                    no;  +: yes)
:
```

出力は **pg** コマンドによってフォーマット済みなので、表示の最下部にフルコロン (**:**) があります。前に示した出力は、最初に表示されるフルスクリーンのみです。残りの出力は通常の **pg** サブコマンドを使用して表示できますが、ここでは簡潔にするために表示していません。この出力は表示専用で、属性を変更することはできません。

属性の名前、例えば **_w** (列数で表したページ幅) を入力して Enter を押すと、次のような内容が表示されます。

```
Name    Description                                          Value
_w      Page Width (characters); Default Value: %IwX         %IwX
                    (value based on paper size specified with s0 -
                    s5 attributes)

To LIST attributes, enter AttributeName1 ... (* for all attributes)
To CHANGE an attribute value, enter AttributeName=NewValue
To FORMAT and EDIT an attribute value, enter AttributeName~v
To EDIT the attribute file, enter ~v
To terminate, press Enter:
```

属性の名前が、メッセージ・カタログからのテキスト記述、および現行値とともに表示されます。プロンプトも再び表示されます。下線から始まる名前を持つ属性の下線は、入力する必要がないことに注意してください。例えば、前記の結果は **w** と入力すれば得られます。この出力は表示専用で、属性は変更できません。

その他の属性は、この形式では読みづらい場合があります。例えば、プロンプトに **ia** と入力して Enter を押すと、次のような出力が表示されます。

```
Name    Description                             Value
ia      ASCII                                    %Ide/pioformat -@%
                                                 Idd/%Imm -!%Idf/pi
                                                 of5202 -l%IwL -w%I
                                                 wW %f[begijpqstuvx
                                                 yzEGIJLOQWXZ] %Uh

To LIST attributes, enter AttributeName1 ..(* for all attributes)
To CHANGE an attribute value, enter AttributeName=NewValue
To FORMAT and EDIT an attribute value, enter AttributeName~v
To EDIT the attribute file, enter ~v
To terminate, press Enter:
```

属性の名前、**=**、および値を入力して Enter を押すと、属性にその値が割り当てられ、新しい値が表示されます。例えば、**_w=60** を入力して Enter を押すか、**w=60** を入力して Enter を押すと、次のような内容が表示されます。

```
To LIST attributes, enter AttributeName1 ..(* for all attributes)
To CHANGE an attribute value, enter AttributeName=NewValue
To FORMAT and EDIT an attribute value, enter AttributeName~v
To EDIT the attribute file, enter ~v
To terminate, press Enter:  w=60

Name    Description                                         Value
_w      COLUMNS per page                                     60

To LIST attributes, enter AttributeName1 ..(* for all attributes)
To CHANGE an attribute value, enter AttributeName=NewValue
To FORMAT and EDIT an attribute value, enter AttributeName~v
To EDIT the attribute file, enter ~v
To terminate, press Enter:
```

w の新しい値が表示されます。(この例の結果、このキューのページ幅は常に 60 列に設定されます)

~v を入力して Enter を押すと、次のような内容が表示されます。

```
:056:__FLG::
:625:CB:S[B]DyEn:
:626:CC:S[C]DyEn:
:627:CD:S[D]DyEn:
:628:CE:S[E]DyEn:
:629:CF:S[F]DyEn:
:630:CG:S[G]DyEn:
:622:Ca:DyS[G500]I[1810532]EnR[pioattr1.cat,1,631;(diag1) - do not print job; di
splay main pipeline and pre-processing filter,(diag2) - do not pr
int job; displa
y all pipelines and filters,(display) - print job; display all pi
```

```
pelines and fil
ters,(ignore) - print job; ignore stderr produced by filters,(nor
mal) - print jo
b; exit if filters produce stderr=-a1,-a0\x27 \x27-A3,-a0\x27 \x2
7-A2,-a0\x27 \x
27-A0,-a0\x27 \x27-A1]:%?%G_a%t-a%I_a%e-a%I_a\x27 \x27-A%I_A%;
:674:Cs:S[B005]I[1810500]EnC[_s,_p]R[%`W0]:-s%I_s\x27 \x27-p%I_p
:013:_A:DnEnR[0,1,2,3]:1
:789:_E:S[B020]I[1810501]%IWY:!
:790:_G:S[E025]I[1810502]%IWY:!
:621:_H:S[F350]I[1810503]Dy:
:024:_I:Dn:
:791:_J:S[C950]I[1810533]%IWY:+
:792:_K:Dn:
:793:_L:S[D020]I[1810504]%IWY:+
:697:_O:DnEnR[1,3]:3
:683:_Q:S[E020]I[1810505]En%IW6:%IwQ
:794:_W:S[B025]I[1810506]%IWY:!
:795:_X:S[D030]I[1810507]EtL[/usr/bin/ls -1 /usr/lib/lpd/pio/tran
s1 | /usr/bin/s
ed '/^850$/d']V[%`WX]:ISO8859-1
:808:_Y:Dn:
:614:_Z:Dn%IWY:+
:063:_a:DnEnR[0,1]:0
:635:_b:S[D010]I[1810508]E#G[0..%?%G_l%{0}%=%t%e%G_l%G_t%-%{1}%-%
d%;]:0
:658:_d:S[C925]I[1810509]EnL[%IW2]F1:a
:615:_e:S[B010]I[1810510]%IWY:!
:659:_f:S[C930]I[1810535]EtL[%IW3]F1V[%`W7]Dy:
:623:_g:S[C250]I[1810511]E#G[1..]:1
"/var/spool/lpd/pio/@local/custom/asc:lp1" 318 lines, 15318 chara
cters
```

この例の最終行に示されているとおり、これはこのキューの要約されていないプリンター・コロン・ファイルを未加工、未フォーマットの状態で編集する **vi** セッションです。**write** コマンドがこの **vi** セッション内で発行されると、定義が **piodigest** コマンドによって要約され、新しい要約バージョンのプリンター・コロン・ファイルが作成されます。

lsvirprt の最も強力なオプションは、属性名に続けて **~v** を入力すると使用できます。例えば、**ia~v** と入力すると、次のような内容が表示されます。

```
ASCII
ia = %Ide/pioformat -@%Idd/%Imm -!%Idf/piof5202 -l%IwL -w%IwW %f[
begijpqstuvxyzEGIJLOQWXZ] %Uh

  %Ide       INCLUDE: (Directory Containing Miscellaneous Modules)
 '/pioformat -@'
  %Idd       INCLUDE: (Directory Containing Digested Data Base Fil
es)
 '/'
 %Imm       INCLUDE: (File Name Of (Digested) Data Base; Init. By
            "piodigest" (mt.md.mn.mq:mv))
  ' -!'
  %Idf       INCLUDE: (Directory Containing Loadable Formatter Rou
tines)
 '/piof5202 -l'
  %IwL       INCLUDE: (Page Length In Chars, Using Length From Dat
a Base (used in pipelines))
 ' -w'
  %IwW       INCLUDE: (Page Width In Characters, Using Width From
Data Base (used in pipelines))
 ' '
  %f[begijpqstuvxyzEGIJLOQWXZ]  For Each Flag x on Command Line: "
```

```
-xArgument" -> OUTPUT
 ' '
  %Uh        Indicate to piobe: Pass the Following Attributes to s
ubsequent printer commands

/tmp/asc:lp1.ia" 24 lines, 1001 characters
```

この例の最終行に示されているとおり、これも **vi** セッションですが、今回は属性定義がフォーマット済みで注釈を付けて表示されています。ここでは、root ユーザーが属性定義を変更できます。**write** コマンドがこの **vi** セッション内で発行されると、定義が **piodigest** コマンドによって要約され、新しい要約バージョンのプリンター・コロン・ファイルが作成されます。

このフォーマット済みの例は、3 つの部分に分かれています。最初の部分は **ia=** で、属性定義が横に連なっています。2 番目の部分は、**vi** セッションの右側にある注釈で、これはプリンター・コロン・ファイルのエスケープ・シーケンスの機能を個別に説明するコメントです。3 番目の部分は、**vi** セッションの左マージンに整列した、フォーマット済みのプリンター・コロン・ファイルのエスケープ・シーケンスです。これらのエスケープ・シーケンスのフォーマット設定には、横方向の部分もあり、if-then-else ステートメント、ネストなどのフローを分かりやすく示すために、インデントが使用されます。

最初と 2 番目の部分は編集できますが、編集による変更には効果がないので、編集は行わないでください。属性の初期定義に対する変更、または注釈に対する変更は、ファイルを書き込んでも **piodigest** に無視されます。3 番目の部分 (フォーマット済みの属性定義) は、編集できます。この部分を編集し、書き込んだ場合に、構文エラーが検出されると、**piodigest** からエラー・メッセージが出されます。通常のプログラミング言語と同様に、論理エラーは起こる可能性がありますが、構文エラーはなくす必要があります。

プリンター・コロン・ファイルの実際の編集例については、『PostScript キューの mi、mp、および _d 属性の変更』を参照してください。

PostScript キューの mi、mp、および _d 属性の変更:

root ユーザーは、仮想プリンター定義の **mi**、**mp**、および **_d** 属性を変更できます。これにより、キュー・バックエンドがファイル・タイプ (PostScript または非 PostScript ASCII) を判別し、印刷環境を適切に設定できます。

入力データ・ストリーム属性は、さまざまな入力データ・ストリーム・タイプ用のパイプラインを格納します。詳しくは、68 ページの『仮想プリンターの入力データ・ストリーム属性』を参照してください。汎用 PostScript プリンターの定義には、**ia** (拡張 ASCII)、**in** (**troff**)、**ip** (パススルー)、および **is** (PostScript) の 4 つの入力データ・ストリーム・パイプラインがあります。コロン・ファイルの **_d** 属性は、4 つの入力データ・ストリーム処理パイプラインのうち、どれがデフォルトで使用されるかを制御します。汎用 PostScript キューの **_d** のデフォルト値は **s** (PostScript) なので、**is** によって定義されたパイプラインが使用されます。

mi 属性は、入力データ・ストリーム・タイプの名前を指定する、コンマ区切りの単一文字です。**mp** 属性は、コンマ区切りの文字列を使用して、入力データ・ストリーム・タイプを識別します。**mi** の文字と **mp** の文字列は、1 対 1 の対になっています。

汎用 PostScript 仮想プリンターの **mi** のデフォルト値は **s** です。**mp** のデフォルト値は **%%!** で、このため PostScript ファイルの先頭 2 文字は **%!** になります。(プリンター・コロン・ファイルのエスケープ・シーケンスはすべて **%** から始まるため、属性定義にリテラルの **%** を使用するには、エスケープのためにもう 1 つ **%** を付ける必要があります。) 仮想プリンターは、**%!** から始まるファイルすべてをデータ・ストリーム・タイプ **s** のものと解釈し、**is** パイプラインを使用します。非 PostScript ASCII ファイルは **%!** から始まらないので、このキューによって印刷されません。

汎用 PostScript 仮想プリンター定義を使用する PostScript キューに非 PostScript ASCII ジョブを実行依頼すると、ジョブが失われます。このキュー上で ASCII 印刷を使用可能にするには、root ユーザーが **lsvirprt** コマンドを使用して、参照される属性を次のように変更できます。

- **mi=a,s**
- **mp=,%%!**
- **_d=%mi**

lsvirprt コマンドを使用して、汎用 PostScript キューを選択します。次のプロンプトが表示されます。

```
To LIST attributes, enter AttributeName1 ..(* for all attributes)
To CHANGE an attribute value, enter AttributeName=NewValue
To FORMAT and EDIT an attribute value, enter AttributeName~v
To EDIT the attribute file, enter ~v
To terminate, press Enter:
```

このプロンプトで次のように入力します。

- **mi=a,s** を入力します。
- **mp=,%%!** を入力します。
- **d=%mi** を入力します。

属性の再定義をそれぞれ入力した後、その属性の新しい値が表示され、その後にプロンプトが表示されます。

この新しい値により、入力データ・ストリーム・タイプ **a** (拡張 ASCII) と任意の文字列の対、および入力データ・ストリーム・タイプ **s** (PostScript) と文字列 **%!** の対が設定されます。**%!** から始まらない入力データ・ストリームは **ia** パイプラインによって処理され、**%!** から始まる入力データ・ストリームは **is** パイプラインによって処理されます。

注: 汎用 PostScript 仮想プリンターに前述の変更を加えずに、コマンド・ラインから入力データ・ストリーム・タイプを指定変更することによって、非 PostScript ASCII ファイルを印刷することもできます。例えば、**qprt** の **d** フラグを次のように使用できます。

```
qprt -Pqueue_name -da /etc/motd
```

このコマンドは、/etc/motd という名前のファイルを **queue_name** という名前のキュー上で印刷し、入力データ・ストリームを ASCII として扱う (**ia** パイプラインが使用される) ように要求します。

プリンター・コロン・ファイルと piobe コマンド:

piobe コマンドは、印刷ジョブを処理するために **qdaemon** プログラムによって呼び出されるスプーラー・バックエンド・プログラムです。

piobe コマンドは、診断出力を生成できます。次の点について解説するために、この診断出力の具体例をこの後の説明に使用します。

- **piobe** がプリンター・コロン・ファイルをどのように使用するか
- パス名を解決するためにコロン・ファイルのエスケープ・シーケンスがどのように評価されるか
- ページ長を解決するためにコロン・ファイルのエスケープ・シーケンスがどのように評価されるか
- ページ幅を解決するためにコロン・ファイルのエスケープ・シーケンスがどのように評価されるか

この説明は、プリンター・コロン・ファイルのエスケープ・シーケンスを詳細に理解する必要がある読者を対象にしています。その理由としては、特有のサポートされないプリンター用に独自のコロン・ファイルを作成したい場合が考えられます。この説明をお読みになる前に、次のトピックについて理解しておく必要があります。

- 74 ページの『プリンター・コロン・ファイルのエスケープ・シーケンス』
- 114 ページの『仮想プリンターの定義』

次のコマンドは、**-a1** フラグ/引数を使用して、**piobe** バックエンドに診断データを要求します。このコマンドの残りの部分が指定する内容は、ジョブが **asc** という名前のキューによって処理され、/etc/motd という名前のファイルが 12 ポイント 90 度回転の Courier フォントで 3 部印刷され、**pr** フィルターによってジョブの前処理が行われ、ジョブから生成されるメッセージはジョブを実行依頼したユーザーにメールで送信されることです。

```
qprt -a1 -Pasc -fp -z1 -p12 -scourier -C -N3 /etc/motd
```

このコマンドを発行すると、コマンドを発行したユーザーに次のようなメールが送信されます。

```
Message from qdaemon:
=====> MESSAGE FROM PRINT JOB 31 (/etc/motd) <=====
0782-034 Below is the preview information requested with the -a1
flag.
         No files will be printed.

PRINTER:
[devices.cat,71,66;IBM 4029 LaserPrinter] (ASCII)

FLAG VALUES:
a=1, b=0, d=a, e=!, f=p, g=1, h=, i=0, j=1, l=48, p=12, q=, s=cou
rier, t=0,
u=1, v=6, w=128, x=2, y=!, z=1, A=1, B=nn, C=+, E=!, G=!, H=, I=,
 J=+, L=+,
N=3, O=3, P=ascx:lxx, Q=1, W=!, X=ISO8859-1, Z=+

PIPELINE OF FILTERS:
/usr/bin/pr
   -l48
   -w128  /etc/motd |
/usr/lib/lpd/pio/etc/pioformat
   -@/var/spool/lpd/pio/@local/ddi/ibm4029.asc.lp1.asc:lp1
   -!/usr/lib/lpd/pio/fmtrs/piof5202
   -l48
   -w128
   -p12
   -scourier
   -z1
```

このメールは、いくつかの項目を指定しています。

- 使用されることになっていた物理プリンター
- このスプーラー・キューに関連したフラグの値
- 実行されることになっていたフィルター・パイプライン

コマンド・ラインで使用されたフラグ値 **a1**、**Pasc**、**fp**、**z1**、**p12**、**scourier**、**C**、および **N3** は、メールの中で「FLAG VALUES」というラベルの付いたセクションに示されています。

興味深い個所は、メールの中で「PIPELINE OF FILTERS」というラベルの付いたセクションです。ここでは、**piobe** によって決定され、シェルによって作成されたフィルター・パイプラインを確認できます。**pr** フィルターは、印刷ジョブ (/etc/motd) の前処理を行い、デバイス独立のフォーマッター・ドライバー **pioformat** に出力を送ります。

この個所は、**piobe** が **asc** という名前のスプーラー・キューに関連した仮想プリンター定義を使用する方法を検討するために適しています。コロン・ファイル (このキューの仮想プリンター定義を格納している) は、属性 **ia** を使用して ASCII ジョブの入力データ・ストリーム・パイプライン (前記の「PIPELINE OF FILTERS」セクション) を指定します。このキューの **ia** の値は、次のとおりです。

```
%Ide/pioformat -@%Idd/%Imm -!%Idf/piof5202 -l%IwL -w%IwW
%f[begijpqstuvxyzEGIJLOQWXZ] %Uh
```

lsvirprt コマンドを使用して **ia** のフォーマットを整えると、次のように表示されます。

```
%Id         INCLUDE: (Directory Containing Miscellaneous Modules)
'/pioformat -@'
%Idd       INCLUDE: (Directory Containing Digested Data Base
Files)
'/'
%Imm     INCLUDE: (File Name Of (Digested) Data Base; Init. By
              "piodigest" (mt.md.mn.mq:mv))
' -!'
%Idf         INCLUDE: (Directory Containing Loadable Formatter
Routines)
'/piof5202 -l'
%IwL       INCLUDE: (Page Length In Chars, Using Length From Data
Base
               (used in pipelines))
' -w'
%IwW     INCLUDE: (Page Width In Characters, Using Width From
Data Base
               (used in pipelines))
' '
%f[begijpqstuvxyzEGIJLOQWXZ] For Each Flag x on Command
Line:"-xArgument" ->
                          OUTPUT
' '
%Uh    Indicate to piobe: Pass the Following Attributes to
subsequent
       printer commands
```

%Id は /usr/lib/lpd/pio/etc に解決されます。このディレクトリーには各種のモジュールがあります。'/pioformat -@' が単一引用符を付けずに前記の文字列に追加され、/usr/lib/lpd/pio/etc/pioformat になります。これは、フォーマッター・ドライバーの絶対パス名です。**pioformat** の後の **-@** は、**pioformat** コマンドのフラグで、この例ではアクセスされる要約データベース・ファイルの絶対パス名を指定します。

-@ フラグの値は、**%Idd**、**'/'**、および **%Imm** の連結によって指定されます。**%Idd** の値は、コロン・ファイル内で **%I@5/ddi** として定義されています。**@5** は自動変数で、値は /var/spool/lpd/pio/@local です。このため、**%Idd** は /var/spool/lpd/pio/@local/ddi に解決されます。そのパスに、**'/'** (単一引用符なし) が追加されます。**%Imm** は、コロン・ファイル内で **mt.md.mn.mq.mv** およびその他の仮想プリンター属性として定義されています。これらの属性が定義するものは、次のとおりです。

- **mt** - プリンター・タイプ
- **md** - 出力データ・ストリーム・タイプ
- **mn** - デバイス名
- **mq** - キュー名 (/etc/qconfig 内のキュー・スタンザの名前)
- **mv** - 仮想プリンター名 (/etc/qconfig 内の対応するデバイス・スタンザの名前)

これらのファイル仮想プリンター属性は、キューと仮想プリンターの作成時に、**piodigest** コマンドによって初期化されます。5 つの属性の組み合わせは、仮想プリンター・データベース内で固有です。

このキューに対する **mt.md.mn.mq.mv** の値は、ibm4029.asc.lp1.asc.lp1 です。このため、**pioformat** の **-@** フラグの値は、/var/spool/lpd/pio/@local/ddi/ibm4029.asc.lp1.asc.lp1 になります。これは、このキュー (**asc**) に関連した仮想プリンターを定義する要約データベース・ファイルの絶対パスです。

'-!' は、**pioformat** の 2 番目のフラグです。このフラグは、フォーマッター・ドライバー **pioformat** によって実行時にロード、リンク、および駆動される、デバイス依存のフォーマッターの絶対パス名を指定します。ここで、これら 2 つのモジュール間で実行時に接続が行われる方法と場所が分かります。

-! フラグの値は、**ia** 属性のフォーマット済み形式に示されているプリンター・コロン・ファイルのエスケープ・シーケンスの残り、つまり **%Idf** と **'/piof5202 -l'** から始まる部分を連結することによって指定されます。

%Idf の値は、コロン・ファイル内で **%I@4/fmtrs** として定義されています。 **@4** は自動変数で、値は /usr/lib/lpd/pio です。このため、**%Idf** は /usr/lib/lpd/pio/fmtrs に解決されます。**'piof5202 -l'** (単一引用符なし) がこの文字列に追加されるので、この時点までで **-!** フラグの値は **/usr/lib/lpd/pio/fmtrs/piof5202 -l** になります。**-l** は **piof5202** (IBM 4029 ページ印刷装置に対する ASCII データ・ストリームのデバイス依存のフォーマッター) のフラグで、ページ幅を文字数で指定します。

-l フラグの引数 **%IwL** の計算については、『プリンター・コロン・ファイルのエスケープ・シーケンスを使用したページ長の計算』を参照してください。

プリンター・コロン・ファイルのエスケープ・シーケンスを使用したページ長の計算:

プリンター・コロン・ファイルのエスケープ・シーケンスを使用して、ページ長を計算できます。

IBM 4029 ページ印刷装置の ASCII キューのプリンター・コロン・ファイルは、作業属性 **wL** によってページ長 (行) を定義します。**wL** の数値を取得するには、**wL** の定義にある組み込み参照を評価する必要があります。**lsvirprt** コマンドのフォーマット済み出力によって示されるとおり、**wL** は次のように定義されます。

```
Page Length In Chars, Using Length From Data Base (used in
 pipelines)
wL = %?%Cl%t%f!l%e%I_l%;

 %?            <IF>
     %Cl       PUSH: (1 If -l Flag on Command Line; Otherwise 0)
 %t            <THEN>
     %f!l      For Each Flag x on Command Line: "-xArgument" ->
OUTPUT
 %e            <ELSE>
     %I_l      INCLUDE: (LINES per page)
 %;            <END>
```

%Cl は、**l** フラグがコマンド・ラインで使用されたかどうかを検査します。使用されている場合は、**1** がスタックにプッシュされます。そうでなければ、**0** がスタックにプッシュされます。この例では、**l** フラグはコマンド・ラインで使用されていないので、**0** がスタックにプッシュされます。**%t** はスタックの真 (ゼロ以外) の値を検査し、見つからない場合は、**%e** (else) 構成体 **%I_l** を実行します。

lsvirprt コマンドのフォーマット済み出力によって示されるとおり、**_l** は **%IwY** として次のように定義されます。

```
Default Page Length (lines)
wY = %?%G_z%{1}%&%t%GwJ%e%GwK%;%G_v%*%{300}%/%d

 %?           <IF>
     %G_z     PUSH: (Page ORIENTATION)
     %{1}     PUSH: (Integer Constant 1)
     %&       PUSH: (pop2 & pop1) -- Bitwise AND
```

```
%t          <THEN>
    %GwJ    PUSH: (Primary Page Width (-z 0) or Secondary Page
                   Length (-z1), in pels)
%e          <ELSE>
    %GwK    PUSH: (Primary Page Length (-z 0) or Secondary Page
                   Width (-z1), in pels)
%;          <END>
%G_v        PUSH: (LINE DENSITY (lines per inch))
%*          PUSH: (pop2 * pop1)
%{300}      PUSH: (Integer Constant 300)
%/          PUSH: (pop2 / pop1)
%d          POP -> ASCII String -> OUTPUT
```

_l の計算では最初に、**_z** (ページの向き) の値をスタックにプッシュします。この例で使用されているジョブ実行依頼コマンド (qprt -a1 -Pasc -fp -z1 -p12 -scourier -C -N3 /etc/motd) は、**z** の値として **1** を指定しているので、**1** がスタックにプッシュされます。**%{1}** によってもう 1 つの **1** がスタックにプッシュされ、その後の **%&** によって先頭にある 2 つの値 (両方とも **1**) がスタックからポップされて、2 つの値のビット単位 AND が実行されます。ビット単位 AND の結果 **1** がスタックにプッシュされます。

> **注:** このテストが単純な等価テストではなくビット単位 AND である理由は、**z** フラグの有効な値が **0**、**1**、**2**、および **3** (印刷ページに適用できる 90 度回転の有効な数に対応する) であるからです。

次の **%t** は、スタック上で **1** を検出するので、**_l** を解決する作業がさらに行われる前に、**then** 節 (**%GwJ**) が解決されます。

lsvirprt コマンドのフォーマット済み出力によって示されるとおり、**wJ** は次のように定義されます。

```
Primary Page Width (-z 0) or Secondary Page Length (-z1), in pels
wJ = %G_Q%Pq%?%GWu%{3}%<%t%?%gq%{1}%=%t%{2400}%e%gq%{2}%=%t%{2400
}%e%gq%{3}%=%t%{1999}%e%gq%{4}%=%t%{2330}%e%{2025}%;%e%?%gq%{1}%=
%t%{1012}%e%gq%{2}%=%t%{1012}%e%gq%{3}%=%t%{1087}%e%gq%{4}%=%t%{1
149}%e%gq%{5}%=%t%{1763}%e%{1928}%;%;%d
```

```
%G_Q          PUSH: (PAPER SIZE override for input paper source)
%Pq           POP -> Internal Variable q
%?            <IF>
    %GWu      PUSH: (Calculate value for paper source based on _
                     0 and _u.)
    %{3}      PUSH: (Integer Constant 3)
    %<        PUSH: (pop2 < pop1 ?)
%t            <THEN>
    %?        <IF>
        %gq   PUSH: (Internal Variable q)
        %{1}  PUSH: (Integer Constant 1)
        %=    PUSH: (pop2 = pop1 ?)
    %t        <THEN>
        %{2400}  PUSH: (Integer Constant 2400)
    %e        <ELSE>
        %gq   PUSH: (Internal Variable q)
        %{2}  PUSH: (Integer Constant 2)
        %=    PUSH: (pop2 = pop1 ?)
    %t        <THEN>
        %{2400}  PUSH: (Integer Constant 2400)
    %e        <ELSE>
        %gq   PUSH: (Internal Variable q)
        %{3}  PUSH: (Integer Constant 3)
        %=    PUSH: (pop2 = pop1 ?)
    %t        <THEN>
        %{1999}  PUSH: (Integer Constant 1999)
    %e        <ELSE>
        %gq   PUSH: (Internal Variable q)
        %{4}  PUSH: (Integer Constant 4)
        %=    PUSH: (pop2 = pop1 ?)
    %t        <THEN>
```

```
        %{2330}  PUSH: (Integer Constant 2330)
    %e          <ELSE>
        %{2025}  PUSH: (Integer Constant 2025)
    %;          <END>
 %e             <ELSE>
    %?          <IF>
        %gq   PUSH: (Internal Variable q)
        %{1}  PUSH: (Integer Constant 1)
        %=    PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{1012}  PUSH: (Integer Constant 1012)
    %e          <ELSE>
        %gq   PUSH: (Internal Variable q)
        %{2}  PUSH: (Integer Constant 2)
        %=    PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{1012}  PUSH: (Integer Constant 1012)
    %e          <ELSE>
        %gq   PUSH: (Internal Variable q)
        %{3}  PUSH: (Integer Constant 3)
        %=    PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{1087}  PUSH: (Integer Constant 1087)
    %e          <ELSE>
        %gq   PUSH: (Internal Variable q)
        %{4}  PUSH: (Integer Constant 4)
        %=    PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{1149}  PUSH: (Integer Constant 1149)
    %e          <ELSE>
        %gq   PUSH: (Internal Variable q)
        %{5}  PUSH: (Integer Constant 5)
        %=    PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{1763}  PUSH: (Integer Constant 1763)
    %e          <ELSE>
        %{1928}  PUSH: (Integer Constant 1928)
    %;          <END>
 %;             <END>
 %d             POP -> ASCII String -> OUTPUT
```

wJ の計算では最初に、**_Q** (用紙入力ソースの用紙サイズの指定変更) の値をスタックにプッシュします。**_Q** の値は **%IwQ** として定義されています。**lsvirprt** コマンドのフォーマット済み出力によって示されるとおり、**wQ** は次のように定義されます。

```
Paper or Envelope Size For the Paper Source Selected By the -O
and -u Flag Values (Refer to the s0, s1, s2, s3, and s4
attributes)
wQ =
%?%GWu%{0}%=%t%Gs0%e%GWu%{1}%=%t%Gs1%e%GWu%{2}%=%t%Gs2%e%GWu%{3}%
=%t%Gs3%e%Gs4%;%d
```

```
 %?             <IF>
     %GWu       PUSH: (Calculate value for paper source based on
_O and _u.)
     %{0}       PUSH: (Integer Constant 0)
     %=         PUSH: (pop2 = pop1 ?)
 %t             <THEN>
     %Gs0       PUSH: (PAPER SIZE for manual paper feed)
 %e             <ELSE>
     %GWu       PUSH: (Calculate value for paper source based on
_O and _u.)
     %{1}       PUSH: (Integer Constant 1)
     %=         PUSH: (pop2 = pop1 ?)
 %t             <THEN>
     %Gs1       PUSH: (PAPER SIZE for tray 1 (upper))
 %e             <ELSE>
```

```
     %GWu      PUSH: (Calculate value for paper source based on
_O and _u.)
     %{2}      PUSH: (Integer Constant 2)
     %=        PUSH: (pop2 = pop1 ?)
 %t            <THEN>
     %Gs2      PUSH: (PAPER SIZE for tray 2 (lower))
 %e            <ELSE>
     %GWu      PUSH: (Calculate value for paper source based on
_O and _u.)
     %{3}      PUSH: (Integer Constant 3)
     %=        PUSH: (pop2 = pop1 ?)
 %t            <THEN>
     %Gs3      PUSH: (ENVELOPE SIZE for envelope feeder)
 %e            <ELSE>
     %Gs4      PUSH: (ENVELOPE SIZE for manual envelope feed)
 %;            <END>
 %d            POP -> ASCII String -> OUTPUT
```

wQ の計算では最初に、**Wu** の値をスタックにプッシュします。**lsvirprt** コマンドのフォーマット済み出力によって示されるとおり、**Wu** の値は次のように定義されます。

```
Calculate value for paper source based on _O and _u.
Wu =
%?%CO%t%?%G_O%{1}%=%t%?%Cu%t%?%G_u%{2}%>%t%{4}%e%{0}%;%e%{0}%;%e%
G_u%;%e%G_u%;%d
```

```
 %?            <IF>
     %CO       PUSH: (1 If -O Flag on Command Line; Otherwise 0)
 %t            <THEN>
     %?        <IF>
         %G_O  PUSH: (Type of INPUT PAPER HANDLING (backward
compatibility
               purpose only))
         %{1}  PUSH: (Integer Constant 1)
         %=    PUSH: (pop2 = pop1 ?)
     %t        <THEN>
         %?    <IF>
             %Cu  PUSH: (1 If -u Flag on Command Line; Otherwise 0)
         %t    <THEN>
             %?  <IF>
                 %G_u  PUSH: (Input PAPER SOURCE)
                 %{2}  PUSH: (Integer Constant 2)
                 %>  PUSH: (pop2 > pop1 ?)
             %t  <THEN>
                 %{4}  PUSH: (Integer Constant 4)
             %e  <ELSE>
                 %{0}  PUSH: (Integer Constant 0)
             %;  <END>
         %e    <ELSE>
             %{0}  PUSH: (Integer Constant 0)
         %;    <END>
     %e        <ELSE>
         %G_u  PUSH: (Input PAPER SOURCE)
     %;        <END>
 %e            <ELSE>
     %G_u      PUSH: (Input PAPER SOURCE)
 %;            <END>
 %d            POP -> ASCII String -> OUTPUT
```

Wu の値の計算では最初に、**%CO** を評価します。この評価により、コマンド・ラインに **O** フラグが指定されていた場合はスタックに **1** がプッシュされ、そうでなければスタックに **0** がプッシュされます。この例で使用されているジョブ実行依頼コマンドは **O** フラグを使用していないので、**0** がスタックにプッシュされます。次の **%t** (スタック上で **0** を検索する) は、続くプリンター・コロン・ファイルのエスケープ・シーケンスを 23 行スキップし、**Wu** 属性のフォーマット済み形式の下から 4 番目の行にある **%e** (else) 節を評価します。else 節は **%G_u** で、これは **_u** の値 (用紙入力ソース) をスタックにプッシュします。

この仮想プリンターの **_u** のデフォルト値は **1** なので、**1** がスタックにプッシュされます。次の **%;** は、元の **%?** を終了します。ただ 1 つ残ったエスケープ・シーケンス **%d** は、先頭の値 (**1**) をスタックからポップし、進行中の **wQ** の計算にこの値を ASCII フォーマットで戻します。

進行中の **wQ** の計算に戻される **1** は **Wu** の値で、スタックにプッシュされます。次の **%{0}** は、**0** をスタックにプッシュします。**%=** はスタックから先頭 2 つの値 (**0** と **1**) をポップし、同等かどうかの検査を行い、失敗します。**0** がスタックにプッシュされます。

次の **%t** は **0** を検出し、したがって **%Gs0** をスキップして、代わりに **%e** (else) 節を評価します。**Wu** (**1**) がスタックに再度プッシュされます。**%{1}** がスタックにもう一度 **1** をプッシュします。 **%=** が先頭 2 つの値 (2 つの **1**) をスタックから再びポップし、同等かどうかの検査を行い、成功します。**1** がスタックにプッシュされます。

次の **%t** は **1** を検出し、したがって **%Gs1** を評価します。 **s1** 属性は、用紙トレイ 1 (上段の用紙トレイ) の用紙サイズを示す数値で、この仮想プリンター定義のデフォルト値は **1** です。この **1** がスタックにプッシュされます。**wQ** の評価のうち、最後の 1 つを除く残りすべてのプリンター・コロン・エスケープ・シーケンスがスキップされます。**%d** は、先頭の値 (**1**) をスタックからポップし、進行中の **wJ** の計算にこの値を ASCII フォーマットで戻します。

進行中の **wJ** の計算に戻される **1** は **_Q** の値で、スタックにプッシュされます。この値はスタックから即時に再びポップされ、内部変数 **q** に格納されます。**Wu** (既に **1** と判定済み) がスタックに再びプッシュされます。**%{3}** はスタックに **3** をプッシュし、**%<** は先頭 2 つの値をスタックからポップして、2 番目にポップした値が最初にポップした値より小さいかどうか検査します。**1** は **3** より小さいので、**1** がスタックにプッシュされます。**%t** は **1** を検出し、したがって if-then-else-then-else-then-else... シーケンスに入って、**_Q** に対して計算された用紙サイズの値と対になる整数を探します。

%gq は、格納された **_Q** の値を内部変数 **q** から取り出し、スタックにプッシュします。**%{1}** がスタックにもう一度 **1** をプッシュします。 **%=** が先頭 2 つの値 (2 つの **1**) をスタックからポップし、同等かどうかの検査を行い、成功します。**1** がスタックにプッシュされます。**%t** は **1** を検出し、したがって **%{2400}** を評価して、スタックに **2400** をプッシュします。**wJ** の計算は、**wJ** を定義する残りのプリンター・コロン・ファイルのエスケープ・シーケンスのうち、最後の 1 行を除いて行われません。最後のエスケープ・シーケンス **%d** は、先頭の値 **2400** をスタックからポップし、進行中の **wY** の計算にこの値を ASCII フォーマットで戻します。

進行中の **wY** の計算に戻される **2400** は **wJ** の値で、スタックにプッシュされます。else 節の **%GwK** はスキップされ、**%;** が if-then-else シーケンスを終了します。 **%G_v** は行密度 (行/インチ) **6** を取り出し、スタックにプッシュします。 **%*** は先頭 2 つの値 (**6** と **2400**) をスタックからポップし、互いを乗算して、結果 (**14400**) をスタックにプッシュします。**%{300}** は **300** をスタックにプッシュします。**%/** は先頭 2 つの値 (**14000** と **300**) をスタックからポップし、スタックから 2 番目にポップした値をスタックから最初にポップした値で割り、結果 (**48**) をスタックにプッシュします。 **%d** は先頭の値 (**48**) をスタックからポップし、進行中の **wL** の計算に戻します。

進行中の **wL** の計算に戻される **48** は、**_l** の値です。 **wL** の値は、元は **ia** 属性 (ASCII ジョブの入力データ・ストリーム・パイプライン) の値の判定中に参照されたものです。その判定の際に、**%IwL** の代わりに数値 **48** が入るので、**pioformat** の **-!** フラグの値は `/usr/lib/lpd/pio/fmtrs/piof5202 -l48` になります。**-l48** は、この説明の基礎となっている **piobe** からの元の診断メッセージに示されています。これは、**piobe** に代わって **qdaemon** から送信されたメールの「**PIPELINE OF FILTERS**」セクションの一部です。

piof5202 の **-w** フラグに関連した値の計算については、131 ページの『プリンター・コロン・ファイルのエスケープ・シーケンスを使用したページ幅の計算』を参照してください。

次の『ページ長の計算』の図は、ページ長 (行) の最終的な数値を得るために使用されるスタック操作 (前述) を示しています。次に示す番号が振られた手順は、図の列の左側にある番号に対応しており、この特定のキュー (**asc**)、コロン・ファイル、およびコマンド・ラインに対するページ長 (行) を定義する、プリンター・コロン・ファイルのエスケープ・シーケンスの評価をステップバイステップで説明しています。

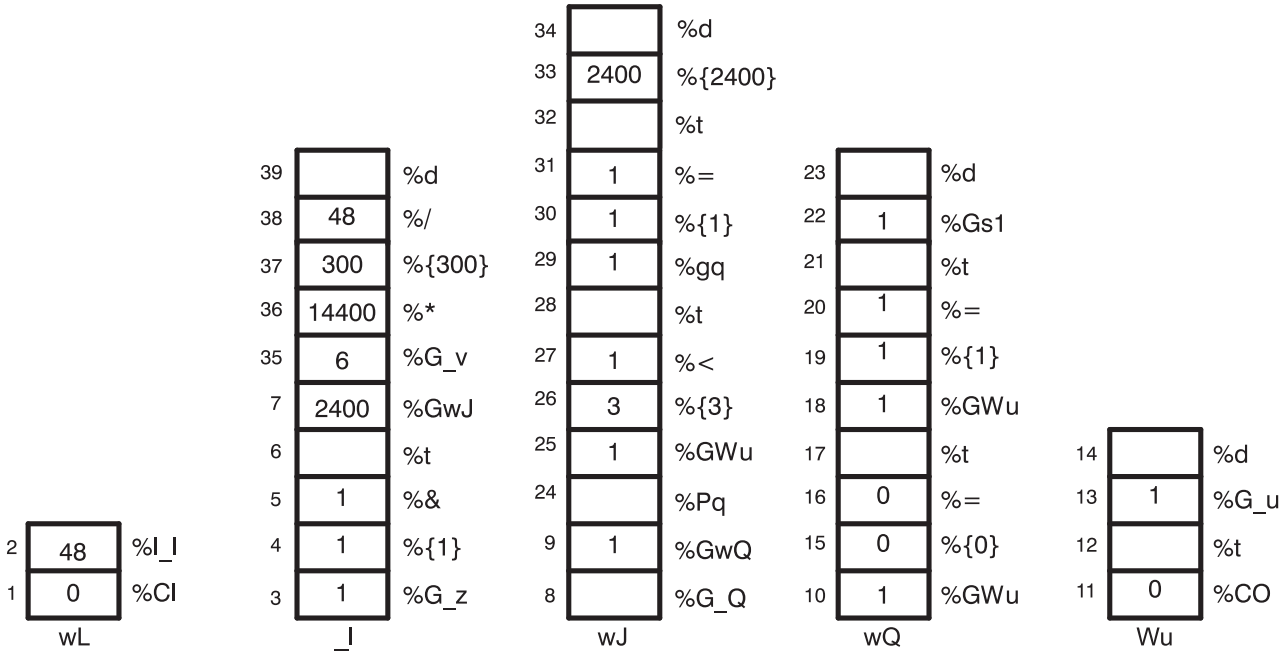

図 4. ページ長の計算

1. **%Cl** - **1** フラグがコマンド・ラインで使用されていないので、**0** をスタックにプッシュします。
2. **%I_l** - **_l** の評価を要求します。
3. **%G_z** - **1** をスタックにプッシュします。
4. **%{1}** - **1** をスタックにプッシュします。
5. **%&** - 先頭 2 つの値 (2 つの **1**) をスタックからポップし、2 つの値のビット単位 AND を実行して、結果の **1** をスタックにプッシュします。
6. **%t** - **1** をスタックからポップし、その値が TRUE (ゼロ以外) なので、**%GwJ** の評価を要求します。**_l** のラベルが付いたスタックは空になります。
7. **%GwJ** - **wJ** の評価を要求します。
8. **%G_Q** - **wQ** の評価を要求します。
9. **%GwQ** - **%GWu** の評価を要求します。
10. **%GWu** - **Wu** の評価を要求します。
11. **%CO** - **O** フラグがコマンド・ラインで使用されていないので、**0** をスタックにプッシュします。
12. **%t** - **0** をスタックからポップし、その値が FALSE (ゼロ) なので、**%G_u** の評価を要求します。**Wu** のラベルが付いたスタックが空になります。
13. **%G_u** - **1** をスタックにプッシュします。
14. **%d** - **1** をスタックからポップして、進行中の **wQ** の計算に ASCII フォーマットで戻します。

15. **%{0}** - **0** をスタックにプッシュします。
16. **%=** - **0** と **1** をスタックからポップし、同等かどうか比較して、結果の **0** をスタックにプッシュします。
17. **%t** - **0** をスタックからポップし、その値が FALSE (ゼロ) なので、**%GwU** の評価を要求します。
18. **%GWu** - この値は既知なので、**1** がスタックにプッシュされます。
19. **%{1}** - **1** をスタックにプッシュします。
20. **%=** - 2 つの **1** をスタックからポップし、同等かどうか比較して、結果の **1** をスタックにプッシュします。
21. **%t** - **1** をスタックからポップし、その値が TRUE (ゼロ以外) なので、**%Gs1** の評価を要求します。
22. **%Gs1** - **1** をスタックにプッシュします。
23. **%d** - **1** をスタックからポップして、進行中の **wJ** の計算に ASCII フォーマットで戻します。
24. **%Pq** - **1** をスタックからポップして、内部変数 **q** に格納します。
25. **%GWu** - この値は既知なので、**1** がスタックに再度プッシュされます。
26. **%{3}** - **3** をスタックにプッシュします。
27. **%<** - **3** と **1** をスタックからポップし、**1** が **3** より小さいので、**1** をスタックにプッシュします。
28. **%t** - **1** をスタックからポップし、その値が TRUE (ゼロ以外) なので、**%gq** の評価を要求します。
29. **%gq** - 内部変数 **q** の値 **1** をスタックにプッシュします。
30. **%{1}** - **1** をスタックにプッシュします。
31. **%=** - 2 つの **1** をスタックからポップし、同等かどうか比較して、結果の **1** をスタックにプッシュします。
32. **%t** - **1** をスタックからポップし、その値が TRUE (ゼロ以外) なので、**%{2400}** の評価を要求します。
33. **%{2400}** - **2400** をスタックにプッシュします。
34. **%d** - **2400** をスタックからポップして、進行中の **_l** の計算に ASCII フォーマットで戻します。
35. **%G_v** - **6** をスタックにプッシュします。
36. **%*** - **6** と **2400** をスタックからポップして、互いを乗算し、結果の **14400** をスタックにプッシュします。
37. **%{300}** - **300** をスタックにプッシュします。
38. **%/** - **300** と **14400** をスタックからポップし、**14400** を **300** で割って、結果の **48** をスタックにプッシュします。
39. **%d** - **48** をスタックからポップして、進行中の **ia** (ASCII ジョブの入力データ・ストリーム・パイプライン) の判定処理に ASCII フォーマットで戻します。

ページ長を記述するスタック言語の機能:

IBM ページ印刷装置 4029 シリーズのテクニカル・リファレンスには、ページの印刷可能領域と印刷不能領域、および標準の用紙と封筒のサイズに対応する用紙と封筒の寸法 (ピクセル) を説明する図と表の組み合わせが記載されています。

例えば、8.5 x 11 (幅 x 長さ) インチのページの印刷可能域は 2400 x 3200 ピクセル (幅 x 長さ) です。横長印刷のためにページを 90 度または 270 度回転する場合は、寸法が置き換わって 3200 x 2400 ピクセル (幅 x 長さ) になることに注意してください。

%IwL の評価は、**l** フラグがコマンド・ラインで使用されたかどうかの検査から始まります。使用されている場合は、計算は実行されません。要求された値が使用されます。(その値が正常に機能するという保証はなく、単に使用されるだけです。) コマンド・ラインで **l** フラグが使用されていない場合、**piobe** は現行ジョブ環境でのページ長を判別する必要があります。これは、他のコマンド・ライン・フラグ、およびコロン・ファイルのデフォルトによって決まります。

_l (ページ長) を評価する際に最初に検査される項目は、ページの向き (**_z**) です。前述のとおり、ページを 90 度の奇数倍だけ回転すると、ページの寸法が反転します。**wY** の定義の始まりである if-then-else ステートメントを見ると、**_z** の値によって、**wJ** と **wK** のどちらがページ長として使用されるかが制御されることが分かります。ページの向きが縦長ならば、**wK** が長さです。ページの向きが横長ならば、**wJ** が長さです。ピクセル単位のページ長が解決されたら、**wY** を定義するエスケープ・シーケンスの残りの部分では、ピクセル数を行数に変換する際に縦方向の行密度を計算に反映するだけです。

wJ 属性が選択された理由は、ページの向きが横長であるからです。現時点で分かっていることは寸法が反転されたことのみで、実際の寸法はまだ不明です。**wJ** の評価は、コマンド・ラインで使用された **Q** フラグの値 (存在する場合) を取り出すことから始まります。この値は、特定の用紙サイズを要求するプリンター依存の値です。 **Q** フラグがコマンド・ラインで使用されている場合は、その値を使用して用紙の長さ (ピクセル) が選択されます。そうでなければ、**Q** の値は **Wu** を評価することによって決まります。これは、属性 **_O** (用紙入力処理のタイプ) と **_u** (用紙入力ソース) に基づいた給紙機構の値です。**_Q** は **%IwQ** として定義され、その定義は **%IWu** で始まる点に注意してください。

Q がコマンド・ラインで使用されておらず、**Wu** の評価によって **O** フラグも使用されていないと判定されたので、**Wu** の定義内で外側にある if-then-else ステートメントの else 文節が実行され、**_u** のデフォルトのコロン・ファイル値 **1** が **wQ** の評価に戻されます。

_l を評価するためにエスケープ・シーケンスはこの深さまでネストするので、**Wu** を定義するロジックについて詳しく検討することは重要です。**O**、**u**、および **Q** の定義と有効な値に留意してください。これらは次のとおりです。

- **O** - 用紙入力処理のタイプ - **1** (手動)、**2** (連続用紙)、**3** (シート・フィード) - デフォルトはシート・フィードです。
- **u** - 用紙入力ソース - **1** (基本)、**2** (代替)、**3** (封筒) - デフォルトは基本です。
- **Q** - 用紙入力ソースの用紙サイズ。値はプリンター依存で、**O** と **u** の組み合わせによって定義されます。

Wu を定義するエスケープ・シーケンスの内容は、次のとおりです。

- ケース 1: **O** フラグがコマンド・ラインで使用されていて、その値が **1** でなかった場合は、コロン・ファイルにある **_u** のデフォルト値を戻します。例えば、ユーザーが用紙入力処理のタイプを指定しなかった場合は、用紙入力ソース (コマンド・ラインからの、またはコロン・ファイルのデフォルト) を **%IwQ** の評価に戻します。
- ケース 2: **O** フラグがコマンド・ラインで使用されていて、その値が **1** でなかった場合は、コロン・ファイルにある **_u** のデフォルト値を戻します。例えば、ユーザーが手動以外の用紙入力処理のタイプを指定した場合は、用紙入力ソース (コマンド・ラインからの、またはコロン・ファイルのデフォルト) を **%IwQ** の評価に戻します。
- ケース 3: **O** フラグがコマンド・ラインで使用され、その値が **1** であり、**u** フラグがコマンド・ラインで使用されていない場合は、**0** を戻します。例えば、ユーザーが手動の用紙処理を指定し、用紙入力ソースを指定しなかった場合は、**0** を **%IwQ** の評価に戻します。

- ケース 4: **O** フラグがコマンド・ラインで使用され、その値が **1** であり、**u** フラグがコマンド・ラインで使用され、その値が **2** より大きくない場合は、**0** を戻します。例えば、ユーザーが手動の用紙処理を指定し、基本または代替の用紙入力ソースを指定した場合は、**0** を **%IwQ** の評価に戻します。
- ケース 5 **O** フラグがコマンド・ラインで使用され、その値が **1** であり、**u** フラグがコマンド・ラインで使用され、その値が **2** より大きい場合は、**4** を戻します。例えば、ユーザーが手動の用紙処理を指定し、さらに用紙入力ソースとして封筒を指定した場合は、**4** を **%IwQ** の評価に戻します。

wQ の定義は、**Wu** の値を整数 **0**、**1**、**2**、および **3** と繰り返し比較して一致を探す if-then-else-then-else-then-else-then-else ステートメントです。この一致により、それぞれ属性 **s0**、**s1**、**s2**、**s3**、または **s4** のいずれかの値が選択されます (他に一致がない場合は **s4** が選択されます)。これらの属性が定義する項目は、次のとおりです。

- **s0** - 手動給紙の用紙サイズ
- **s1** - トレイ 1 (上段) の用紙サイズ
- **s2** - トレイ 2 (下段) の用紙サイズ
- **s3** - 封筒給紙ユニットの封筒サイズ
- **s4** - 手動封筒給紙の封筒サイズ

IBM 4029 ページ印刷装置の ASCII キューの仮想プリンター定義では、これら 5 つの属性に対して固有値は 2 つのみです。**s0**、**s1**、および **s2** はすべて **1** で、**s3** と **s4** は両方とも **3** です。

ネストされたエスケープ・シーケンスを再び検討すると、**wJ** の定義は外側の if-then-else ステートメントから作成されることが分かります。このステートメントの if と else の両部分に、if-then-else-then-else... ステートメントのチェーンが含まれています。**Wu** の値 (**O** と **u** に基づく、給紙機構の値) によって、外側のステートメントの if 部分または else 部分を実行するかどうかが決まります。 **Wu** が **1** または **2** (**3** より小さい) ならば、if 部分が実行され、そうでなければ else 部分が実行されます。**wJ** の最終的な判定の際に、ページ長 (ピクセル) が固定されます。

wJ を定義する外側の if-then-else ステートメントの if 部分は、封筒以外の用紙サイズの範囲からピクセル値を選択します。外側の if-then-else ステートメントの else 部分は、封筒用紙サイズの範囲からピクセル値を選択します。**Wu** は、if-then-else ステートメントのどの部分が実行されるかを制御します。ただし、if または else の部分が選択された後、ピクセル値の選択は **Q** の値によって決まります。前に示した 5 つのケースでは、次のように処理されます。

ケース 1: **u** のコマンド・ライン値、またはコロン・ファイルからのデフォルト (**1**、基本用紙トレイ) が **wQ** の評価に戻されます。**wQ** の定義にある残りのエスケープ・シーケンスは、**Wu** の値をテストし、**s0**、**s1**、**s2**、**s3**、または **s4** のいずれかの値を選択します。さらに、その値が **wJ** の評価に戻されます。 **u** が **1** または **2** ならば、**Q** は **1** (封筒以外の用紙サイズ) です。**u** が **3** ならば、**Q** は **3** (封筒用紙サイズ) です。**wJ** の評価が再開されたとき、**u** の値が **1** または **2** ならば、処理は外側の if-then-else ステートメントの if 部分に誘導され、**Q** の値は **1** なので 2400 ピクセルのページ長が選択されます。 **u** の値が **3** ならば、処理は外側の if-then-else ステートメントの else 部分に誘導され、**Q** の値は **3** なので 1087 ピクセルの封筒ページ長が選択されます。

ケース 2: ケース 1 と同じ。

ケース 3: コマンド・ラインでユーザーが手動用紙処理を指定しましたが、給紙機構は指定していないので、**Wu** には値 **0** が割り当てられ、その値が **wQ** の評価に戻されます。この **0** により、**wQ** には **s0** の値 (手動給紙の用紙サイズ、**1**) が割り当てられます。**wJ** の評価が再開されたとき、**u** の値が **0** なので、処理は外側の if-then-else ステートメントの if 部分に誘導され、**Q** の値は **1** (**s0**) なので 2400 ピクセルのページ長が選択されます。

ケース 4: コマンド・ラインでユーザーが手動用紙処理を指定し、また **u** フラグを使用して基本または代替の給紙機構 (ただし、封筒ではない) を指定しました。ケース 3 と同様に、2400 ピクセルのページ長が選択されます。

ケース 5: コマンド・ラインでユーザーが手動用紙処理を指定し、また **u** フラグを使用して封筒給紙機構を指定したので、**Wu** には値 **4** が割り当てられ、その値が **wQ** の評価に戻されます。この 4 により、**wQ** には **s4** の値 (手動封筒給紙の封筒サイズ、**3**) が割り当てられます。 **wJ** の評価が再開されたとき、**u** の値は **4** なので、処理は外側の if-then-else ステートメントの else 部分に誘導され、**Q** の値は **3** なので 1087 ピクセルの封筒の長さが選択されます。

ここでの例はケース 1 で、**O** または **u** のどちらのフラグもコマンド・ラインで使用されていないので、**Wu** には値 **1** が割り当てられます。これは、このコロン・ファイルの **_u** のデフォルト値です。**wQ** の評価が再開されると、**s1** に一致が検出され、**1** が **wJ** の評価に戻されます。**u** の値が **1** なので、処理は外側の if-then-else ステートメントの if 部分に誘導され、**Q** の値は **1** なので 2400 ピクセルのページ長が選択されます。この値が **_l** の評価に戻されます。

_l を定義する残りのプリンター・コロン・ファイルのエスケープ・シーケンスにより、もし 2400 ピクセル (縦方向) が使用可能であり、6 行/インチで印刷し、1 インチあたりのピクセル数が 300 (プリンターの解像度) ならば、1 ページに 48 行を印刷できることが推論されます。値 **48** が **ia** の評価に戻されます。つまり、これが「**PIPELINE OF FILTERS**」の **-l48** の根拠です。

プリンター・コロン・ファイルのエスケープ・シーケンスを使用したページ幅の計算:

プリンター・コロン・ファイルのエスケープ・シーケンスを使用して、ページ幅を計算できます。

IBM 4029 ページ印刷装置の ASCII キューのプリンター・コロン・ファイルは、作業属性 **wW** によってページ幅 (文字) を定義します。**lsvirprt** (詳しくは 114 ページの『仮想プリンターの定義』を参照) のフォーマット済み出力に示されるとおり、**wW** は次のように定義されます。

```
Page Width In Characters, Using Width From Data Base (used in
pipelines)
wW = %?%Cw%t%f!w%e%I_w%;

 %?             <IF>
     %Cw        PUSH: (1 If -w Flag on Command Line; Otherwise 0)
 %t             <THEN>
     %f!w       For Each Flag x on Command Line: "-xArgument" ->
OUTPUT
 %e             <ELSE>
     %I_w       INCLUDE: (COLUMNS per page)
 %;             <END>
```

%Cw は、**w** フラグがコマンド・ラインで使用されたかどうかを検査します。使用されている場合は、**1** がスタックにプッシュされます。そうでなければ、**0** がスタックにプッシュされます。この例では、**w** フラグはコマンド・ラインで使用されていないので、**0** がスタックにプッシュされます。**%t** はスタックの真 (ゼロ以外) の値を検査し、見つからない場合は、**%e** (else) 構成体 **%I_w** を実行します。

lsvirprt コマンドのフォーマット済み出力によって示されるとおり、**_w** 属性は **%IwX** として次のように定義されます。

```
Default Page Width (characters)
wX =
%?%G_z%{1}%&%t%GwK%e%GwJ%;%?%G_p%{17}%=%t%{171}%e%G_p%{10}%*%;%*%
?%G_W%t%{6000}%e%{3000}%;%/%d
```

```
 %?             <IF>
     %G_z       PUSH: (Page ORIENTATION)
     %{1}       PUSH: (Integer Constant 1)
     %&         PUSH: (pop2 & pop1) -- Bitwise AND
 %t             <THEN>
     %GwK       PUSH: (Primary Page Length (-z 0) or Secondary
Page Width (-z
                1), in pels)
 %e             <ELSE>
     %GwJ       PUSH: (Primary Page Width (-z 0) or Secondary Page
Length (-z
                1), in pels)
 %;             <END>
 %?             <IF>
     %G_p       PUSH: (PITCH (characters per inch))
     %{17}      PUSH: (Integer Constant 17)
     %=         PUSH: (pop2 = pop1 ?)
 %t             <THEN>
     %{171}     PUSH: (Integer Constant 171)
 %e             <ELSE>
     %G_p       PUSH: (PITCH (characters per inch))
     %{10}      PUSH: (Integer Constant 10)
     %*         PUSH: (pop2 * pop1)
 %;             <END>
 %*             PUSH: (pop2 * pop1)
 %?             <IF>
     %G_W       PUSH: (DOUBLE-WIDE print?)
 %t             <THEN>
     %{6000}    PUSH: (Integer Constant 6000)
 %e             <ELSE>
     %{3000}    PUSH: (Integer Constant 3000)
 %;             <END>
 %/             PUSH: (pop2 / pop1)
 %d             POP -> ASCII String -> OUTPUT
```

_w の計算では最初に、**_z** (ページの向き) の値をスタックにプッシュします。この例で使用されているジョブ実行依頼コマンド (qprt -a1 -Pasc -fp -p12 -scourier -C -N3 /etc/motd) は、**z** の値として **1** を指定しているので、**1** がスタックにプッシュされます。**%{1}** によってもう 1 つの **1** がスタックにプッシュされ、その後の **%&** によって先頭にある 2 つの値 (両方とも **1**) がスタックからポップされて、2 つの値のビット単位 AND が実行されます。ビット単位 AND の結果 **1** がスタックにプッシュされます。

> **注:** このテストが単純な等価テストではなくビット単位 AND である理由は、**z** フラグの有効な値が **0**、**1**、**2**、および **3** (印刷ページに適用できる 90 度回転の有効な数に対応する) であるからです。

次の **%t** は、スタック上で真 (ゼロ以外) の値を検出するので、**_w** を解決する作業がさらに行われる前に、then 節 (**%GwK**) が解決されます。

lsvirprt コマンドのフォーマット済み出力によって示されるとおり、**wK** は次のように定義されます。

```
Primary Page Length (-z 0) or Secondary Page Width (-z 1), in pels
wK =
%G_Q%Pq%?%GWu%{3}%<%t%?%gq%{1}%=%t%{3200}%e%gq%{2}%=%t%{4100}%e%g
q%{3}%=%t%{2935}%e%gq%{4}%=%t%{3407}%e%{3050}%;%e%?%gq%{1}%=%t%{2
150}%e%gq%{2}%=%t%{2562}%e%gq%{3}%=%t%{2750}%e%gq%{4}%=%t%{2498}%
e%gq%{5}%=%t%{2604}%e%{2852}%;%;%d
```

```
 %G_Q           PUSH: (PAPER SIZE override for input paper source)
 %Pq            POP -> Internal Variable q
 %?             <IF>
     %GWu       PUSH: (Calculate value for paper source based on
_O and _u.)
     %{3}       PUSH: (Integer Constant 3)
     %<         PUSH: (pop2 < pop1 ?)
```

```
%t              <THEN>
    %?          <IF>
        %gq     PUSH: (Internal Variable q)
        %{1}    PUSH: (Integer Constant 1)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{3200}  PUSH: (Integer Constant 3200)
    %e          <ELSE>
        %gq     PUSH: (Internal Variable q)
        %{2}    PUSH: (Integer Constant 2)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{4100}  PUSH: (Integer Constant 4100)
    %e          <ELSE>
        %gq     PUSH: (Internal Variable q)
        %{3}    PUSH: (Integer Constant 3)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{2935}  PUSH: (Integer Constant 2935)
    %e          <ELSE>
        %gq     PUSH: (Internal Variable q)
        %{4}    PUSH: (Integer Constant 4)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{3407}  PUSH: (Integer Constant 3407)
    %e          <ELSE>
        %{3050}  PUSH: (Integer Constant 3050)
    %;          <END>
%e              <ELSE>
    %?          <IF>
        %gq     PUSH: (Internal Variable q)
        %{1}    PUSH: (Integer Constant 1)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{2150}  PUSH: (Integer Constant 2150)
    %e          <ELSE>
        %gq     PUSH: (Internal Variable q)
        %{2}    PUSH: (Integer Constant 2)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{2562}  PUSH: (Integer Constant 2562)
    %e          <ELSE>
        %gq     PUSH: (Internal Variable q)
        %{3}    PUSH: (Integer Constant 3)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{2750}  PUSH: (Integer Constant 2750)
    %e          <ELSE>
        %gq     PUSH: (Internal Variable q)
        %{4}    PUSH: (Integer Constant 4)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{2498}  PUSH: (Integer Constant 2498)
    %e          <ELSE>
        %gq     PUSH: (Internal Variable q)
        %{5}    PUSH: (Integer Constant 5)
        %=      PUSH: (pop2 = pop1 ?)
    %t          <THEN>
        %{2604}  PUSH: (Integer Constant 2604)
    %e          <ELSE>
        %{2852}  PUSH: (Integer Constant 2852)
    %;          <END>
%;              <END>
%d              POP -> ASCII String -> OUTPUT
```

wK の計算では最初に、**_Q** (用紙入力ソースの用紙サイズの指定変更) の値をスタックにプッシュします。**_Q** の値は **%IwQ** として定義されています。**Wk** の計算中、この時点の状況は **wJ** を計算していたときとまったく同じです。つまり、**wQ** と **Wu** の値を判定しようとしている個所です。単一ジョブの実行依頼コマンドのコンテキストでは、最終的な値が別の属性の計算から要求されたからという理由で、**wQ** と **Wu** の最終的な値が変わることはありません。このため、ここでは既に計算した **wQ** の値 **1** と、**Wu** の値 **1** を使用します。

進行中の **wK** の計算に戻される **1** は **_Q** の値で、スタックにプッシュされます。この値はスタックから即時に再びポップされ、内部変数 **q** に格納されます。**Wu** (既に **1** と判定済み) がスタックに再びプッシュされます。**%{3}** はスタックに **3** をプッシュし、**%<** は先頭 2 つの値 (**3** と **1**) をスタックからポップして、2 番目にポップした値が最初にポップした値より小さいかどうか検査します。**1** は **3** より小さいので、**1** がスタックにプッシュされます。**%t** は **1** を検出し、したがって if-then-else-then-else-then-else... シーケンスに入って、**_Q** に対して計算された用紙サイズの値と対になる整数を探します。

%gq は、格納された **_Q** の値を内部変数 **q** から取り出し、スタックにプッシュします。**%{1}** は **1** をスタックにプッシュします。**%=** が先頭 2 つの値 (2 つの **1**) をスタックからポップし、同等かどうかの検査を行い、成功します。**1** がスタックにプッシュされます。**%t** は **1** を検出し、したがって **%{3200}** を評価して、スタックに **3200** をプッシュします。**wK** の計算は、**wK** を定義する残りのプリンター・コロン・ファイルのエスケープ・シーケンスのうち、最後の 1 行を除いて行われません。最後のエスケープ・シーケンス **%d** は、先頭の値 **3200** をスタックからポップし、進行中の **wX** の計算にこの値を ASCII フォーマットで戻します。

進行中の **wX** の計算に戻される **3200** は **wK** の値で、スタックにプッシュされます。else 節の **%GwJ** はスキップされ、**%;** が if-then-else シーケンスを終了します。 **wJ** の計算中、この時点から属性定義の残りの部分では、ページ長 (行) に影響を及ぼす要素 (縦方向の行密度など) を処理しました。一方、ページ幅の計算で考慮することは、ピッチと、横倍角印刷が選択されたかどうかです。

次に評価されるエスケープ・シーケンスは **%G_p** です。これは、**_p** 属性の値を取り出します。この属性は、このキューのピッチ (字/インチ) を定義します。このキューのデフォルト値は **10** ですが、この例で使用されるコマンド・ラインはピッチとして **12** を指定しているので (**-p12**)、**12** がスタックにプッシュされます。**%{17}** は **17** をスタックにプッシュします。**%=** はスタックから先頭 2 つの値 (**17** と **12**) をポップし、同等かどうかの検査を行い、失敗します。**0** がスタックにプッシュされます。**%t** は **0** (偽の値) を検出し、後続の else 節が評価されます。**%G_p** は **12** をスタックに再びプッシュします。 **%{10}** は **10** をスタックにプッシュします。**%*** は先頭 2 つの値 (**12** と **10**) をスタックからポップし、互いを乗算します。結果 **120** がスタックにプッシュされます。**%;** はこの if-then-else シーケンスを終了します。

後続の **%*** は先頭 2 つの値 (**120** と **3200**) をスタックからポップし、互いを乗算します。結果 **384000** がスタックにプッシュされます。**%G_W** は **_W** の値を取り出し、スタックにプッシュします。**_W** は、横倍角印刷が必要かどうかを「はい」(**1**) または「いいえ」(**0**) で示します。デフォルト値は **0** で、この値をコマンド・ラインで指定変更していないので、**0** がスタックにプッシュされます。**%t** は **0** を検出し、したがって else 節を実行します。**%{3000}** は **3000** をスタックにプッシュします。**%;** はこの if-then-else シーケンスを終了します。後続の **%/** は先頭 2 つの値 (**3000** と **384000**) をスタックからポップし、2 番目にポップした値を最初にポップした値で割ります。結果 **128** がスタックにプッシュされます。**%d** は、先頭の値 **128** をスタックからポップし、進行中の **wW** の計算にこの値を ASCII フォーマットで戻します。

進行中の **wW** の計算に戻される **128** は、**_w** の値です。 **wW** の値は、元は **ia** 属性 (ASCII ジョブの入力データ・ストリーム・パイプライン) の値の判定中に参照されたものです。その判定の際に、**%IwW** の代わりに数値 **128** が入るので、**pioformat** の **-!** フラグの値は `/usr/lib/lpd/pio/fmtrs/piof5202 -l48`

-w128 になります。**-w128** は、この説明の基礎となっている **piobe** からの元の診断メッセージに示されています。これは、**piobe** に代わって **qdaemon** から送信されたメールの「**PIPELINE OF FILTERS**」セクションの一部です。

次の『ページ幅の計算』の図は、ページ幅 (文字) の最終的な数値を得るために使用されるスタック操作 (前述) を示しています。次に示す番号が振られた手順は、図の列の左側にある番号に対応しており、この特定のキュー (**asc**)、コロン・ファイル、およびコマンド・ラインに対するページ幅 (文字) を定義する、プリンター・コロン・ファイルのエスケープ・シーケンスの評価をステップバイステップで説明しています。

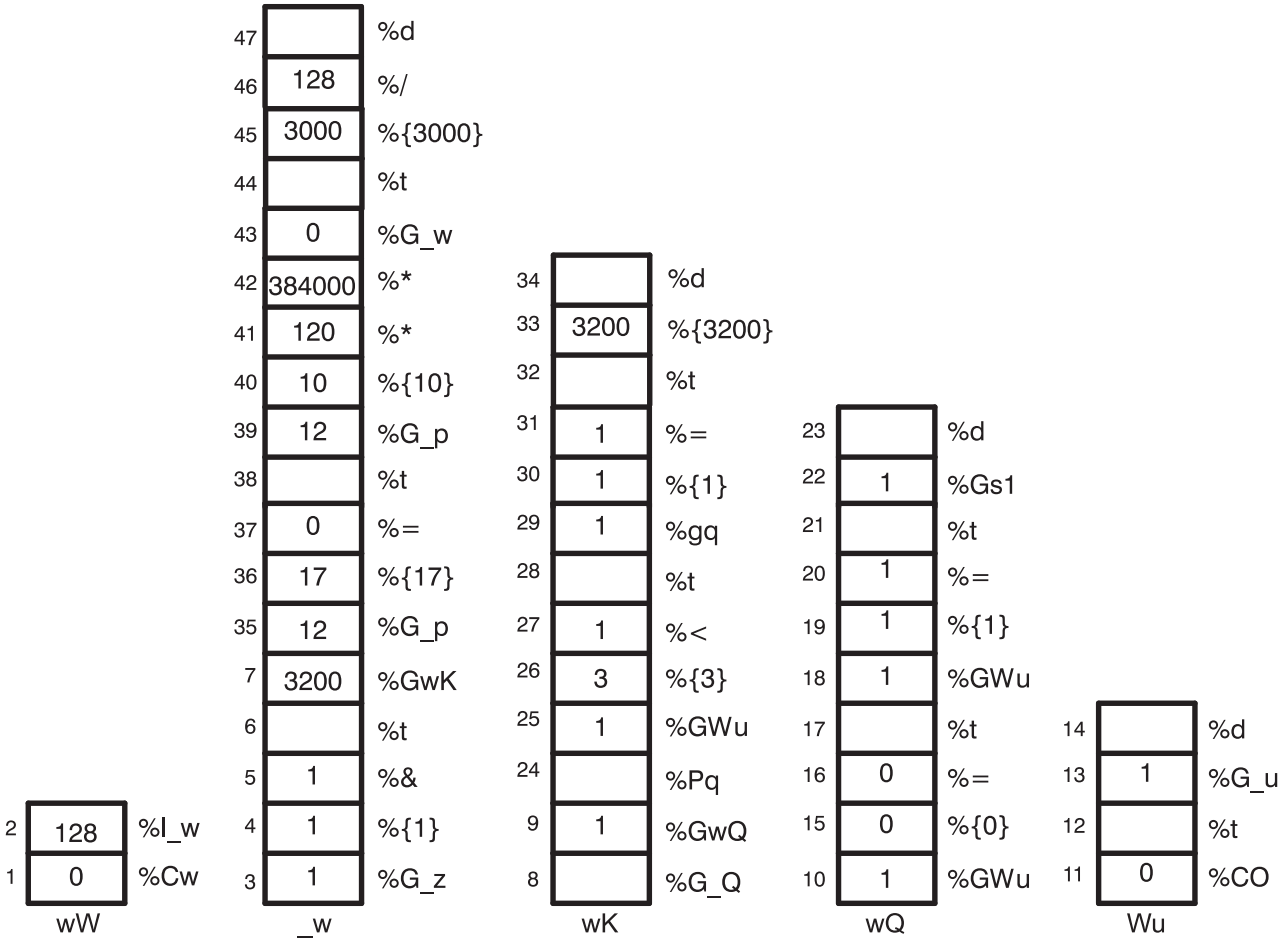

図 5. ページ幅の計算

1. **%Cw** - **w** フラグがコマンド・ラインで使用されていないので、**0** をスタックにプッシュします。
2. **%I_w** - **_w** の評価を要求します。
3. **%G_z** - **1** をスタックにプッシュします。
4. **%{1}** - **1** をスタックにプッシュします。
5. **%&** - 先頭 2 つの値 (2 つの **1**) をスタックからポップし、2 つの値のビット単位 AND を実行して、結果の **1** をスタックにプッシュします。
6. **%t** - **1** をスタックからポップし、その値が TRUE (ゼロ以外) なので、**%GwK** の評価を要求します。
7. **%GwK** - **wK** の評価を要求します。

8. **%G_Q** - **_Q** の評価を要求します。
9. **%GwQ** - **wQ** の評価を要求します。
10. **%GWu** - **Wu** の評価を要求します。
11. **%CO** - **O** フラグがコマンド・ラインで使用されていないので、**0** をスタックにプッシュします。
12. **%t** - **0** をスタックからポップし、その値が FALSE (ゼロ) なので、**%G_u** の評価を要求します。**Wu** のラベルが付いたスタックが空になります。
13. **%G_u** - **1** をスタックにプッシュします。
14. **%d** - **1** をスタックからポップして、進行中の **wQ** の計算に ASCII フォーマットで戻します。
15. **%{0}** - **0** をスタックにプッシュします。
16. **%=** - **0** と **1** をスタックからポップし、同等かどうか比較して、結果の **0** をスタックにプッシュします。
17. **%t** - **0** をスタックからポップし、その値が FALSE (ゼロ) なので、**%GWu** の評価を要求します。
18. **%GWu** - この値は既知なので、**1** がスタックにプッシュされます。
19. **%{1}** - **1** をスタックにプッシュします。
20. **%=** - 2 つの **1** をスタックからポップし、同等かどうか比較して、結果の **1** をスタックにプッシュします。
21. **%t** - **1** をスタックからポップし、その値が TRUE (ゼロ以外) なので、**%Gs1** の評価を要求します。
22. **%Gs1** - **1** をスタックにプッシュします。
23. **%d** - **1** をスタックからポップして、進行中の **wK** の計算に ASCII フォーマットで戻します。
24. **%Pq** - **1** をスタックからポップして、内部変数 **q** に格納します。
25. **%GWu** - この値は既知なので、**1** がスタックにプッシュされます。
26. **%{3}** - **3** をスタックにプッシュします。
27. **%<** - 先頭 2 つの値をスタックからポップし (**3** と **1**)、**1** が **3** より小さいので、**1** をスタックにプッシュします。
28. **%t** - **1** をスタックからポップし、その値が TRUE (ゼロ以外) なので、**%pq** の評価を要求します。
29. **%pq** - 内部変数 **q** の値 **1** をスタックにプッシュします。
30. **%{1}** - **1** をスタックにプッシュします。
31. **%=** - 先頭 2 つの値 (2 つの **1**) をスタックからポップし、同等かどうか比較して、結果の **1** をスタックにプッシュします。
32. **%t** - **1** をスタックからポップし、その値が TRUE (ゼロ以外) なので、**%{3200}** の評価を要求します。
33. **%{3200}** - **3200** をスタックにプッシュします。
34. **%d** - **3200** をスタックからポップして、進行中の **_w** の計算に戻します。
35. **%G_p** - **12** をスタックにプッシュします。
36. **%{17}** - **17** をスタックにプッシュします。
37. **%=** - 先頭 2 つの値 (**17** と **12**) をスタックからポップし、同等かどうか比較して、結果の **0** をスタックにプッシュします。
38. **%t** - **0** をスタックからポップし、その値が FALSE (ゼロ) なので、**%G_p** の評価を要求します。
39. **%G_p** - **12** をスタックにプッシュします。
40. **%{10}** - **10** をスタックにプッシュします。

41. **%*** - 先頭 2 つの値 (**10** と **12**) をスタックからポップして、互いに乗算し、結果の **120** をスタックにプッシュします。
42. **%*** - 先頭 2 つの値 (**120** と **3200**) をスタックからポップして、互いに乗算し、結果の **384000** をスタックにプッシュします。
43. **%G_w** - **0** をスタックにプッシュします。
44. **%t** - **0** をスタックからポップし、その値が FALSE (ゼロ) なので、**%{3000}** の評価を要求します。
45. **%{3000}** - **3000** をスタックにプッシュします。
46. **%/** - 先頭 2 つの値 (**3000** と **384000**) をスタックからポップし、2 番目にポップした値を最初にポップした値で割って、結果 **128** をスタックにプッシュします。
47. **%d** - **128** をスタックからポップして、進行中の **ia** (ASCII ジョブの入力データ・ストリーム・パイプライン) の計算に ASCII フォーマットで戻します。

ページ幅を記述するスタック言語の機能:

IBM ページ印刷装置 4029 シリーズのテクニカル・リファレンスには、ページの印刷可能領域と印刷不能領域、および標準の用紙と封筒のサイズに対応する用紙と封筒の寸法 (ピクセル) を説明する図と表の組み合わせが記載されています。

例えば、8.5 x 11 (幅 x 長さ) インチのページの印刷可能域は 2400 x 3200 ピクセル (幅 x 長さ) です。横長印刷のためにページを 90 度または 270 度回転する場合は、寸法が置き換わって 3200 x 2400 ピクセル (幅 x 長さ) になることに注意してください。

%IwW の評価は、**w** フラグがコマンド・ラインで使用されたかどうかの検査から始まります。使用されている場合は、計算は実行されません。要求された値が使用されます。(その値が正常に機能するという保証はなく、単に使用されるだけです。)コマンド・ラインで **w** フラグが使用されていない場合、**piobe** は現行ジョブ環境でのページ幅を判別する必要があります。これは、他のコマンド・ライン・フラグ、およびコロン・ファイルのデフォルトによって決まります。

_w (ページ幅) を評価する際に最初に検査される項目は、ページの向き (**_z**) です。前述のとおり、ページを 90 度の奇数倍だけ回転すると、ページの寸法が反転します。**wK** の定義の始まりである if-then-else ステートメントを見ると、**_z** の値によって、**wJ** と **wK** のどちらがページ幅として使用されるかが制御されることが分かります。ページの向きが縦長ならば、**wJ** が幅です。ページの向きが横長ならば、**wK** が幅です。ピクセル単位のページ幅が解決されたら、**wK** を定義するエスケープ・シーケンスの残りの部分では、ピクセル数を文字数に変換する際にピッチと文字幅 (横倍角かそうでないか) を計算に反映するだけです。

wK 属性が選択された理由は、ページの向きが横長であるからです。現時点で分かっていることは寸法が反転されたことのみで、実際の寸法はまだ不明です。**wK** の評価は、コマンド・ラインで使用された **Q** フラグの値 (存在する場合) を取り出すことから始まります。この値は、特定の用紙サイズを要求するプリンター依存の値です。 **Q** フラグがコマンド・ラインで使用されている場合は、その値を使用して用紙の幅 (ピクセル) が選択されます。そうでなければ、**Q** の値は **Wu** を評価することによって決まります。これは、属性 **_O** (用紙入力処理のタイプ) と **_u** (用紙入力ソース) に基づいた給紙機構の値です。**_Q** は **%IwQ** として定義され、その定義は **%IWu** で始まる点に注意してください。

Q がコマンド・ラインで使用されておらず、**Wu** の評価によって **O** フラグも使用されていないと判定されたので、**Wu** の定義内で外側にある if-then-else ステートメントの else 文節が実行され、**_u** のデフォルトのコロン・ファイル値 **1** が **wQ** の評価に戻されます。

_w を評価するためにエスケープ・シーケンスはこの深さまでネストするので、**Wu** を定義するロジックについて詳しく検討することは重要です。**O**、**u**、および **Q** の定義と有効な値に留意してください。これらは次のとおりです。

- **O** - 用紙入力処理のタイプ - **1** (手動)、**2** (連続用紙)、**3** (シート・フィード) - デフォルトはシート・フィードです。
- **u** - 用紙入力ソース - **1** (基本)、**2** (代替)、**3** (封筒) - デフォルトは基本です。
- **Q** - 用紙入力ソースの用紙サイズ。値はプリンター依存で、**O** と **u** の組み合わせによって定義されます。

Wu を定義するエスケープ・シーケンスの内容は、次のとおりです。

- ケース 1: **O** フラグがコマンド・ラインで使用されていて、その値が **1** でなかった場合は、コロン・ファイルにある **_u** のデフォルト値を戻します。例えば、ユーザーが用紙入力処理のタイプを指定しなかった場合は、用紙入力ソース (コマンド・ラインからの、またはコロン・ファイルのデフォルト) を **%IwQ** の評価に戻します。
- ケース 2: **O** フラグがコマンド・ラインで使用されていて、その値が **1** でなかった場合は、コロン・ファイルにある **_u** のデフォルト値を戻します。例えば、ユーザーが手動以外の用紙入力処理のタイプを指定した場合は、用紙入力ソース (コマンド・ラインからの、またはコロン・ファイルのデフォルト) を **%IwQ** の評価に戻します。
- ケース 3: **O** フラグがコマンド・ラインで使用され、その値が **1** であり、**u** フラグがコマンド・ラインで使用されていない場合は、**0** を戻します。例えば、ユーザーが手動の用紙処理を指定し、用紙入力ソースを指定しなかった場合は、**0** を **%IwQ** の評価に戻します。
- ケース 4: **O** フラグがコマンド・ラインで使用され、その値が **1** であり、**u** フラグがコマンド・ラインで使用され、その値が **2** より大きくない場合は、**0** を戻します。例えば、ユーザーが手動の用紙処理を指定し、さらに基本または代替の用紙入力ソースを指定した場合は、**0** を **%IwQ** の評価に戻します。
- ケース 5: **O** フラグがコマンド・ラインで使用され、その値が **1** であり、**u** フラグがコマンド・ラインで使用され、その値が **2** より大きい場合は、**4** を戻します。例えば、ユーザーが手動の用紙処理を指定し、さらに用紙入力ソースとして封筒を指定した場合は、**4** を **%IwQ** の評価に戻します。

wQ の定義は、**Wu** の値を整数 **0**、**1**、**2**、および **3** と繰り返し比較して一致を探す if-then-else-then-else-then-else-then-else ステートメントです。この一致により、それぞれ属性 **s0**、**s1**、**s2**、**s3**、または **s4** のいずれかの値が選択されます (他に一致がない場合は **s4** が選択されます)。これらの属性が定義する項目は、次のとおりです。

- **s0** - 手動給紙の用紙サイズ
- **s1** - トレイ 1 (上段) の用紙サイズ
- **s2** - トレイ 2 (下段) の用紙サイズ
- **s3** - 封筒給紙ユニットの封筒サイズ
- **s4** - 手動封筒給紙の封筒サイズ

IBM 4029 ページ印刷装置の ASCII キューの仮想プリンター定義では、これら 5 つの属性に対して固有値は 2 つのみです。**s0**、**s1**、および **s2** はすべて **1** で、**s3** と **s4** は両方とも **3** です。

ネストされたエスケープ・シーケンスを再び検討すると、**wK** の定義は外側の if-then-else ステートメントから作成されることが分かります。このステートメントの if と else の両部分に、if-then-else-then-else... ステートメントのチェーンが含まれています。**Wu** の値 (**O** と **u** に基づく、給紙機構の値) によって、外側

のステートメントの if 部分または else 部分を実行するかどうかが決まります。 **Wu** が **1** または **2** (**3** より小さい) ならば、if 部分が実行され、そうでなければ else 部分が実行されます。**wK** の最終的な判定の際に、ページ幅 (ピクセル) が固定されます。

ケース 1: **u** のコマンド・ライン値、またはコロン・ファイルからのデフォルト (**1**、基本用紙トレイ) が **wQ** の評価に戻されます。**wQ** の定義にある残りのエスケープ・シーケンスは、**Wu** の値をテストし、**s0**、**s1**、**s2**、**s3**、または **s4** のいずれかの値を選択します。さらに、その値が **wK** の評価に戻されます。 **u** が **1** または **2** ならば、**Q** は **1** (封筒以外の用紙サイズ) です。**u** が **3** ならば、**Q** は **3** (封筒用紙サイズ) です。**wK** の評価が再開されたとき、**u** の値が **1** または **2** ならば、処理は外側の if-then-else ステートメントの if 部分に誘導され、**Q** の値は **1** なので 3200 ピクセルのページ幅が選択されます。 **u** の値が **3** ならば、処理は外側の if-then-else ステートメントの else 部分に誘導され、**Q** の値は **3** なので 2750 ピクセルの封筒ページ幅が選択されます。

ケース 2: ケース 1 と同じ。

ケース 3: コマンド・ラインでユーザーが手動用紙処理を指定しましたが、給紙機構は指定していないので、**Wu** には値 **0** が割り当てられ、その値が **wQ** の評価に戻されます。この **0** により、**wQ** には **s0** の値 (手動給紙の用紙サイズ、**1**) が割り当てられます。**wK** の評価が再開されたとき、**u** の値が **0** なので、処理は外側の if-then-else ステートメントの if 部分に誘導され、**Q** の値は **1** (**s0**) なので 3200 ピクセルのページ幅が選択されます。

ケース 4: コマンド・ラインでユーザーが手動用紙処理を指定し、また **u** フラグを使用して基本または代替の給紙機構 (ただし、封筒ではない) を指定しました。ケース 3 と同様に、3200 ピクセルのページ幅が選択されます。

ケース 5: コマンド・ラインでユーザーが手動用紙処理を指定し、また **u** フラグを使用して封筒給紙機構を指定したので、**Wu** には値 **4** が割り当てられ、その値が **wQ** の評価に戻されます。この 4 により、**wQ** には **s4** の値 (手動封筒給紙の封筒サイズ、**3**) が割り当てられます。 **wK** の評価が再開されたとき、**u** の値は **4** なので、処理は外側の if-then-else ステートメントの else 部分に誘導され、**Q** の値は **3** なので 2498 ピクセルの封筒の幅が選択されます。

- ここでの例はケース 1 で、**O** または **u** のどちらのフラグもコマンド・ラインで使用されていないので、**Wu** には値 **1** が割り当てられます。これは、このコロン・ファイルの **_u** のデフォルト値です。**wQ** の評価が再開されると、**s1** に一致が検出され、**1** が **wK** の評価に戻されます。**u** の値が **1** なので、処理は外側の if-then-else ステートメントの if 部分に誘導され、**Q** の値は **1** なので 3200 ピクセルのページ幅が選択されます。この値が **_w** の評価に戻されます。

_w を定義する残りのプリンター・コロン・ファイルのエスケープ・シーケンスにより、もし 3200 ピクセル (横方向) が使用可能であり、12 字/インチで印刷し、プリンターの解像度が 300 ピクセル/インチならば、1 ページに 128 文字を印刷できることが推論されます。17 ピッチが指定される可能性を考慮して、ピッチとプリンター解像度の両方に 10 が掛けられます。17 ピッチは実際には 17.1 なので、分子と分母の両方に 10 を掛けることによって、ページ幅の最終的な計算に .1 が反映されるようになります。値 **128** が **ia** の評価に戻されます。つまり、これが「**PIPELINE OF FILTERS**」の **-128** の根拠です。

スプーラー・ジョブのヘッダー・ページとトレーラー・ページ:

ヘッダー・ページとトレーラー・ページを生成するためのパイプラインは、システム管理属性 **sh** (ヘッダー・ページ) と **st** (トレーラー・ページ) によって定義されます。

ヘッダー・ページとトレーラー・ページの印刷は、キュー状況照会の出力には表示されませんが、これらのページが付随するスプーラー印刷ジョブとは別個のプロセスです。

ヘッダー・ページとトレーラー・ページのパイプライン:

sh 属性は、ヘッダー・ページとトレーラー・ページのパイプラインを定義するために使用されます。

次に、IBM 4029 ページ印刷装置で拡張 ASCII キューのヘッダー・ページを生成し、印刷するためのパイプラインの定義に使用される **sh** 属性を示します。ここでは、**lsvirprt** コマンドによるこの属性のフォーマット済み出力を示します。詳しくは、 114 ページの『仮想プリンターの定義』を参照してください。

```
Pipeline for Header Page
sh = %Ide/pioburst %F[H] %Idb/H.ascii | %Ide/pioformat
-@%Idd/%Imm -!%Idf/piof52
02 -L! -J! %IsH -u%IuH

 %Ide          INCLUDE: (Directory Containing Miscellaneous
Modules)
 '/pioburst '
 %F[H]         If "-H] Argument" on Command Line, "-# Argument"
-> OUTPUT
 ' '
 %Idb          INCLUDE: (Directory Containing Header and Trailer
Text Files)
 '/H.ascii | '
 %Ide          INCLUDE: (Directory Containing Miscellaneous
Modules)
 '/pioformat -@'
 %Idd          INCLUDE: (Directory Containing Digested Data Base
Files)
 '/'
 %Imm          INCLUDE: (File Name Of (Digested) Data Base; Init.
By
               "piodigest" (mt.md.mn.mq:mv))
 ' -!'
 %Idf          INCLUDE: (Directory Containing Loadable Formatter
Routines)
 '/piof5202 -L! -J! '
 %IsH          INCLUDE: (FORMATTING FLAGS for header page)
 ' -u'
 %IuH          INCLUDE: (Input PAPER TRAY for header page)
```

スプーラー・ジョブの処理中に、**sh** 属性の値は次のように決定されます。

```
/usr/lib/lpd/pio/etc/pioburst /usr/lib/lpd/pio/burst/H.ascii |
/usr/lib/lpd/pio/etc/pioformat
-@/var/spool/lpd/pio/@local/ddi/ibm4029.asc.lp1.asc:lp1
-!/usr/lib/lpd/pio/fmtrs/piof5202 -L! -J! -u1
```

pioburst コマンドは、ヘッダー・ページ・テンプレートを処理し、その出力をデバイス独立のフォーマッター **pioformat** にパイピングします。このフォーマッターは、この仮想プリンター用のコロン・ファイルの要約バージョン (**-@** フラグの引数)、およびデバイス依存のフォーマッター **piof5202** (**-!** フラグの引数) をロードします。 **piof5202** には次の 3 つのフラグがあります。

1. **-L!** - 長い行を折り返しません。
2. **-J!** - ヘッダー・ページを印刷する前の状態にプリンターをリストアします。
3. **-u1** - ヘッダー・ページを用紙トレイ 1 から給紙します。

st 定義の値は、**sh** 定義の値と同様です。

カスタム・ヘッダー・ページ:

root ユーザーは、**sh** 属性の定義を変更して、ユーザー用のカスタム・ヘッダー・ページを作成できます。

スプーラー・プロセスはスプーラーにジョブを実行依頼したユーザーの環境にアクセスできるので、root ユーザーは **sh** 属性定義のうち、処理するヘッダー・ページ・テンプレートを指定する部分を変更できます。

例えば、**H.ascii** は処理および印刷するヘッダー・ページ・テンプレートを指定します。これは次に示すように、任意のユーザー環境変数 (**$MYHEADER** など) に置き換えることができます。

```
 %Ide          INCLUDE: (Directory Containing Miscellaneous
Modules)
 '/pioburst '
 %F[H]         If "-H] Argument" on Command Line, "-# Argument"
-> OUTPUT
 ' '
 %Idb          INCLUDE: (Directory Containing Header and Trailer
Text Files)
 '/$MYHEADER | '
 %Ide          INCLUDE: (Directory Containing Miscellaneous
Modules)
 '/pioformat -@'
 %Idd          INCLUDE: (Directory Containing Digested Data Base
Files)
 '/'
 %Imm          INCLUDE: (File Name Of (Digested) Data Base; Init.
 By
               "piodigest" (mt.md.mn.mq:mv))
 ' -!'
 %Idf          INCLUDE: (Directory Containing Loadable Formatter
Routines)
 '/piof5202 -L! -J! '
 %IsH          INCLUDE: (FORMATTING FLAGS for header page)
 ' -u'
 %IuH          INCLUDE: (Input PAPER TRAY for header page)
```

ユーザー **susan** がこのキューによってカスタム・ヘッダー・ページを使用できるようにするには、root ユーザーが次の手順で行います。

- `cp /usr/lib/lpd/pio/burst/H.ascii /usr/lib/lpd/pio/burst/H.susan` と入力します。
- Susan のヘッダー・ページに関する希望に合わせて、`H.susan` を編集します。
- Susan の環境の環境変数 **MYHEADER** を **H.susan** に設定します。 (例えば Korn シェルでは、`export MYHEADER=H.susan` を使用します。)

ユーザー **susan** がこのキューにジョブを実行依頼すると、**sh** 属性のヘッダー・ページ・テンプレートの参照が `/usr/lib/lpd/pio/burst/H.susan` に解決され、ユーザー **susan** はカスタム・ヘッダー・ページを受け取ります。このシナリオの問題は、この仮想プリンターに関連したキューを使用するすべてのユーザーに対して、環境変数 **MYHEADER** を定義する必要があることです。定義しなければ、仮想プリンターは `/usr/lib/lpd/pio/burst/$MYHEADER` の参照を解決できません。**$MYHEADER** が未定義の場合はエラーが出されます。ジョブは印刷されますが、ヘッダー・ページはリサイクル可能になるだけです。

このキューを使用するユーザー全員に **MYHEADER** を定義する必要があるという問題を避けるために、**sh** 属性の定義にシェル・コードを統合して、ヘッダー・ページ・パイプラインの作成前にユーザー環境を検査するようにできます。これを実現する方法の 1 つを次に示します。

```
Pipeline for Header Page
sh = { if test X"$MYHEADER" = X ; then %Ide/pioburst %F[H]
%Idb/H.ascii | %Ide/pioformat -@%Idd/%Imm -!%Idf/piof5202 -L! -J!
%IsH -u%IuH; else %Ide/pioburst %F[H] %Idb/$MYHEADER |
%Ide/pioformat -@%Idd/%Imm -!%Idf/piof5202 -L! -J! %IsH -u%IuH;
fi; }
```

```
 '{ if test X"$MYHEADER" = X ; then '
 %Ide          INCLUDE: (Directory Containing Miscellaneous
Modules)
 '/pioburst '
 %F[H]         If "-H] Argument" on Command Line, "-# Argument"
-> OUTPUT
 ' '
 %Idb          INCLUDE: (Directory Containing Header and Trailer
Text Files)
 '/H.ascii | '
 %Ide          INCLUDE: (Directory Containing Miscellaneous
Modules)
 '/pioformat -@'
 %Idd          INCLUDE: (Directory Containing Digested Data Base
Files)
 '/'
 %Imm          INCLUDE: (File Name Of (Digested) Data Base; Init.
By
               "piodigest" (mt.md.mn.mq:mv))
 ' -!'
 %Idf          INCLUDE: (Directory Containing Loadable Formatter
Routines)
 '/piof5202 -L! -J! '
 %IsH          INCLUDE: (FORMATTING FLAGS for header page)
 ' -u'
 %IuH          INCLUDE: (Input PAPER TRAY for header page)
 '; else '
 %Ide          INCLUDE: (Directory Containing Miscellaneous
Modules)
 '/pioburst '
 %F[H]         If "-H] Argument" on Command Line, "-# Argument"
-> OUTPUT
 ' '
 %Idb          INCLUDE: (Directory Containing Header and Trailer
Text Files)
 '/$MYHEADER | '
 %Ide          INCLUDE: (Directory Containing Miscellaneous
Modules)
 '/pioformat -@'
 %Idd          INCLUDE: (Directory Containing Digested Data Base
Files)
 '/'
 %Imm          INCLUDE: (File Name Of (Digested) Data Base; Init.
By
               "piodigest" (mt.md.mn.mq:mv))
 ' -!'
 %Idf          INCLUDE: (Directory Containing Loadable Formatter
Routines)
 '/piof5202 -L! -J! '
 %IsH          INCLUDE: (FORMATTING FLAGS for header page)
 ' -u'
 %IuH          INCLUDE: (Input PAPER TRAY for header page)
 '; fi; } '
```

元の **st** の定義が新しい **st** の定義では 2 回繰り返されています。シェル・コードは **MYHEADER** が定義されているかどうかを検査し、**MYHEADER** が定義されていない場合はヘッダー・ページ・テンプレート `H.ascii` が使用され、定義されていればヘッダー・ページ・テンプレート **$MYHEADER** が使用されます。

mo 仮想プリンター属性の変更:

すべての仮想プリンター定義に、**mo** という名前の属性が含まれています。

mo 属性は、デバイス・ドライバー・インターフェース・プログラムを呼び出すためのコマンド・ストリングを指定します。デバイス・ドライバー・インターフェース・プログラムは、入力データ・ストリーム処理パイプラインの最後のプロセスで、**piobe** をバックエンドとするローカル・スプーラー・キューの場合、通常は **pioout** です。デバイス・ドライバー・インターフェース・プログラムと呼ばれている理由は、パイプラインの最後のプロセスとして、このプログラムが通常はデバイス・ドライバーを書き込み用に開き、処理された入力データ・ストリームをデバイス・ドライバーに書き込むからです。詳細については、56 ページの『一般的な印刷ジョブのデータ・ストリーム・フロー』を参照してください。

基本オペレーティング・システム・スプーラーの設計により、root ユーザーは処理パイプラインの入力データ・ストリームを、ユーザー作成のコードに部分的に置き換えることができます。 **pioout** の絶対パスをデフォルト値とする **mo** 属性を、ユーザー作成のデリバリー・プログラムの絶対パスに再定義する例を説明します。詳細については、55 ページの『バックエンド処理』を参照してください。

サポートされない IP アドレッシング可能端末サーバー:

プリンターのモードと入力データ・ストリームを操作するフォーマッター・フィルターの機能を使用して、ユーザーが ASCII ジョブを実行依頼できる印刷サーバーへのキューを作成できます。

ご使用のイーサネット・ネットワークに、IP アドレッシング可能な端末サーバーが接続されているとします。端末サーバーには、ASCII 端末、モデム、プリンター、その他の非同期デバイスを接続できる非同期ポートがいくつか存在します。また、端末サーバーのベンダーが **ts_print** という名前のプログラムを提供していて、このプログラムには次のような特性があるとします。

- 標準入力から読み取る。
- IP アドレスを指定する **-A** フラグを受け入れる。
- ポート番号を指定する **-P** フラグを受け入れる。

この具体的な例として、IBM 4029 ページ印刷装置を端末サーバーのポート 11 に接続する場合を考え、端末サーバーの IP アドレスは 9.19.129.101 であるとします。目標は、ユーザーが ASCII ジョブを実行依頼できる印刷サーバーへのキューを作成し、端末サーバー上の 4029 でジョブを印刷することです。コマンド・ラインから **ts_print** を使用することもできますが、フォーマッター・フィルターの機能を利用して、プリンターのモードと入力データ・ストリームの両方を高度に操作することにします。プリンターへの真のシリアル・アクセスを提供することも目標の 1 つです。

この目標を達成する方法は複数あります。最も容易な方法は、/dev ディレクトリー内の文字スペシャル・ファイルではなく、通常のファイルにローカル ASCII キューを作成することです。キューおよび関連した仮想プリンターを作成した後、**ts_print** を使用するように仮想プリンターを変更できます。

キュー作成プロセスを開始するには、SMIT 高速パス **smit mkquedev** を入力します。次のようなメニューが表示されます。

```
Add a Print Queue

Move cursor to desired item and press Enter. Use arrow keys to scroll.

# ATTACHMENT TYPE        DESCRIPTION
    local                Printer Attached to Local Host
    remote               Printer Attached to Remote Host
    ascii                Printer Attached to ASCII Terminal
    hpJetDirect          Network Printer (HP JetDirect)
    file                 File (in /dev directory)
    other                User Defined Backend
```

「**file (ファイル)**」オプションを選択し、プリンター・タイプを選択します。 IBM 4029 ページ印刷装置 (または状況に応じた適切な項目) を選択した後、/dev ディレクトリー内の既存ファイルの名前を指定しま

す。これは、作成しているキューに対して実行依頼され、処理されたジョブが書き込まれる先のファイルです。ファイル名は、基本オペレーティング・システムの命名規則に準拠する任意の名前にすることができます。合理的な処置として、ファイル・キューのターゲットにすることのみを目的としたファイルを作成します。例えば、root ユーザーはコマンド `touch /dev/l`*xx* を発行して、l*xx* という名前のファイルを `/dev` ディレクトリー内に作成できます。

`/dev` ディレクトリーのファイルの名前を指定した後、前に選択したプリンター・タイプによってサポートされるそれぞれの入力データ・ストリームごとに、キュー名を選択します。この例では、ASCII キューに対して **asc** という名前を選択したとします。次のようなエントリーが `/etc/qconfig` 内に作成されます。

```
asc:
        device = lxx
lxx:
        file = /dev/lxx
        header = never
        trailer = never
        access = both
        backend = /usr/lib/lpd/piobe
```

スプーラー・キュー **asc** に対して実行依頼された印刷ジョブは、**piobe** によってセットアップされたパイプラインによって処理されます。処理されたデータ・ストリームは、`/dev/l`*xx* に書き込まれます。これは望ましい動作ではありません。目標は **ts_print** が出力を端末サーバーのポート 11 に書き込むようにすることなので、実際のところはこのキューに関連したファイルが存在していてはなりません。このために、次に示すように `/etc/qconfig` の新規スタンザ・ペアを編集し、**file** パラメーターの値を **FALSE** に変更します。

```
asc:
        device = lxx
lxx:
        file = FALSE
        header = never
        trailer = never
        access = both
        backend = /usr/lib/lpd/piobe
```

この状態でこのキューを使用すると、エラー・メッセージが出された場合を除いては、何かがファイルに書き込まれたり、どこかで印刷が行われたりすることはありません。**qdaemon** は、バックエンド **piobe** の実行準備をする際に、`/etc/qconfig` の **file** パラメーターの値に応じて、オープン・ファイル・ディスクリプターを **piobe** に渡します。その値が **FALSE** に設定されている場合、ファイル・ディスクリプターは渡されません。ファイル・ディスクリプターを最終的に受け取って使用するものは、**mo** 属性によって指示されたいずれかのプログラムです。 **mo** 属性が指定するデフォルト・プログラムは **pioout** であり、キューがこの状態にあるときにジョブがキューに入れられると、**pioout** の値は stdout に対して有効でなく、処理されたジョブは単に消滅します。

この時点で、**lsvirprt** を使用して **asc** 仮想プリンター定義を選択し、変更できます (詳しくは、 114 ページの『仮想プリンターの定義』を参照)。次のようなプロンプトが表示されます。

```
To LIST attributes, enter AttributeName1 ... (* for all attributes)
To CHANGE an attribute value, enter AttributeName=NewValue
To FORMAT and EDIT an attribute value, enter AttributeName~v
To EDIT the attribute file, enter ~v
To terminate, press Enter:
```

ts_print プログラムが `/usr/bin` にインストールされているとして、プロンプトに次のように入力します。

```
mo=/usr/bin/ts_print -A 9.19.129.101 -P 11
```

これで、**asc** キューに対して実行依頼されたジョブがローカル・ジョブと同様に処理されるようになりますが、パイプラインの最後に達すると、**pioout** がデバイス・ドライバーに出力データ・ストリームを配送するのでなく、**ts_print** プログラムが端末サーバーのポート 11 に出力データ・ストリームを配送します。

一般に、**piobe** をバックエンドとするキューの仮想プリンター定義の **mo** 属性を再定義すれば、ユーザーが選択した任意のファイルまたはデバイスに処理済みデータ・ストリームを配送できます (そのためのコードを作成できれば)。

フィルター:

仮想プリンター定義には、事前定義およびオープン (未定義) のフィルター属性が含まれています。

例えば、IBM 4029 ページ印刷装置の AIX バージョン 4 ASCII キューには、次のフィルター属性が備わっています。

- f1、f2、f3、f4、および f5 - オープン、ユーザー定義フィルター
- fb - ヘブライ語/アラビア語用 BIDI フィルター
- fc - cifplot フィルター
- fd - TeX (DVI) フィルター
- ff - FORTRAN フィルター
- fg - プロット・フィルター
- fl - パススルー・フィルター
- fn - ditroff フィルター
- fp - **pr** フィルター
- fv ラスター・イメージ・フィルター
- fc、fd、ff、fg、fl、fn、ft、fv - オープン、ユーザー定義フィルター
- fp - **pr** フィルター

フィルターは、データ・ストリームを選択的に操作できる、**piobe** コマンドによってセットアップされた入力データ・ストリーム処理**パイプライン**内の最初のプログラムです。ジョブごとに特定のフィルターをコマンド・ラインで選択でき、また仮想プリンター定義を変更して永続的に選択することもできます。

ジョブごとに特定のフィルターを選択するには、**qprt** コマンドの **-f** フラグを使用します。**-f** フラグの引数は、仮想プリンター定義にある 2 文字のフィルター属性名の 2 文字目です。例えば、IBM 4029 ページ印刷装置への **asc** という名前の ASCII キューにあるジョブに対して **pr** フィルターを選択するには、次のコマンドを実行できます。

```
qprt -Pasc -fp /etc/motd
```

pr フィルターを選択する最初の属性の名前は **fp** なので、**-f** フラグの引数は 2 文字目の **p** のみです。

pr フィルターを永続的に選択するには、**lsvirprt** コマンドを使用して仮想プリンター定義を編集し、**_f** 属性の値を **p** に設定します。**_f** 属性は、この仮想プリンター定義に関連したキューに対して実行依頼された、すべてのジョブの前処理に使用されるフィルターを選択します。

lp、**lpr**、および **qprt** はすべて、スプーラーへの真の入り口点である **enq** コマンドの単なるフロントエンドなので、**enq** は **-f** フラグをサポートしているように思えます。しかし、**-f** フラグを付けて **enq** コマンドを発行すると、エラー・メッセージが出されます。つまり、**enq** は **-f** フラグをサポートしません。この状況では、前に説明したように、`/bin/enq` を介して `/bin/echo` をマウントする技法 (54 ページの『スプーラー・データ・フロー (enq コマンド)』を参照) が役に立ちます。

root ユーザーは、シェル・プロンプトから次のコマンドを実行できます。

1. `mount /bin/echo /bin/enq`
2. `qprt -Pasc -fp /etc/motd`
3. `umount /bin/enq`

2 番目のコマンドを発行した後、TERM 環境変数によって定義される表示エレメントの内容は次のようになります。

```
-P asc -o -f -o p /etc/motd
```

これらは、**qprt** が **enq** に渡そうとした引数です。この引数が反映されている理由は、**qprt** が **enq** でなく **echo** を検出したからです。次のコマンドは、前記のステップ 2 に示したコマンドと等価です。

```
enq -P asc -o -f -o p /etc/motd
```

-o オプションは、バックエンド固有のフラグをバックエンドに渡すことを指定します。**-o** オプションは、**enq** コマンドがジョブ記述ファイルを作成し、**qdaemon** に新規ジョブの存在を通知する前に行われる構文検査を通過させるフリー・パスと考えることができます。

ASCII ファイルから一連の行を印刷するキューをセットアップしたいとします。例えば、/usr/lpp/bos/README を読んで、ファックスで送るか参照用に壁に貼るために印刷したい個所が 35 行見つかったとします。この場合は、/etc/qconfig を編集して、次の行を追加できます。

```
partial:
        device = partial
partial:
        file = FALSE
        backend = /usr/bin/partial
```

ファイル /usr/bin/partial は、所有権 root.printq とアクセス権 755 を指定したシェル・スクリプトです。内容は次のようなものです。

```
#!/bin/ksh
BEGIN=$1
END=$2
let DIFF=END-BEGIN+1
FILE=$3
/usr/bin/head -${END} ${FILE} | tail -${DIFF} | /usr/bin/qprt -Pasc
```

/usr/lpp/bos/README の行 189 から 223 を印刷したい場合は、**partial** キューを次のように使用できます。

```
qprt -Ppartial -o 189 -o 223 /usr/lpp/bos/README
```

バックエンドが実行されると、**BEGIN** には 189 が割り当てられ、**END** には 223 が割り当てられ、**DIFF** には 35 が割り当てられます。これが選択した行の数です。**FILE** には /usr/lpp/bos/README が割り当てられます。**head** コマンドは、要求された最後の行の直後で /usr/lpp/bos/README を切り捨てます。出力は **tail** コマンドにパイピングされ、このコマンドは切り捨てられたファイルの末尾 35 行を選択して、**qprt** コマンドにパイピングします。このコマンドは、入力を stdin から受け取ります。**qprt** コマンドは、**asc** という名前のキューにこれらの行を送信します。

改行を復帰改行にマップするためのフィルター:

多くのユーザーが、あらかじめ印刷済みの小切手、送り状、船荷証券などの用紙の空欄を埋めるデータ・ストリームを作成するアプリケーションを独自に作成したり、購入したりしています。これらのデータ・ストリームを印刷するには、物理プリンターの精密な制御が必要です。

piobe によって作成されたジョブ処理パイプラインが、元のデータ・ストリームに対してある程度の量のデータを挿入または削除すると、出力データが事前印刷用紙の正しい位置に印刷されなくなることがよくあります。

root ユーザーは **lsvirprt** コマンドを頻繁に使用して、仮想プリンター定義の **_d** 属性の値を **p** に設定することがあります。この結果、IBM 4029 ページ印刷装置の ASCII キュー上では、**piobe** はジョブの処理に **ip** パイプラインを選択します。**ip** パイプラインはパススルー印刷用なので、フォーマッター・フィルターは passthru() ルーチンを使用して、入力データ・ストリームを変更せずプリンターに単に渡します。

これにより、プリンターの制御に関して起こっていた問題がすべて解決する一方で、新しい問題が加わることがたびたびあります。フォーマッター・フィルターがパススルー・モードで動作しているときは、改行から復帰改行へのマッピングができなくなります。用紙は依然として正しく印刷されません。

アプリケーションがデータ・ストリームへの復帰の挿入を許可していない場合、次のような簡易フィルターを使用してこの問題を解決できます。

```
#include <stdio.h>

main(int argc, char **argv)
{
int ch ;

while (EOF != (ch = fgetc(stdin)))
 {
  switch (ch)
   {
    case 10: fputc(ch,stdout) ;
             fputc(0x0D,stdout) ;
             break ;
    default: fputc(ch,stdout) ;
             break ;
   }
 }
}
```

プログラムをコンパイルし、**cr_mapper** という名前を付けて、/usr/lib/lpd などのアクセス可能なロケーションにインストールします。所有権 **root.printq** とアクセス権 **555** を割り当てます。

IBM 4029 ページ印刷装置への **asc** という名前の ASCII キューがあるとすると、**lsvirprt** を使用して **asc** キューを選択し、**f1** フィルター属性のフォーマットを設定できます。次のような情報が表示されます。

```
User defined filter 1
f1 =
```

f1 属性のデフォルト値はヌルなので、定義は空です。

f1 属性を編集して、次のように定義します。

```
User defined filter 1
f1 =
 '/usr/lib/lpd/cr_mapper'
```

f1 の新しい定義を保管する際に、ここでも **lsvirprt** を使用してフォーマットを整えることができます。次のような内容が表示されます。

```
User defined filter 1
f1 = /usr/lib/lpd/cr_mapper
 '/usr/lib/lpd/cr_mapper'
```

次のようなコマンドを使用して、**f1** フィルターをコマンド・ラインから使用できるようになりました。

```
qprt -Pasc -f1 filename
```

```
enq -Pasc -o -f -o 1 filename
```

_d 属性が **p** に設定されていない場合は、**-dp** フラグと引数をコマンドに追加する必要があります。

```
qprt -Pasc -dp -f1 filename
enq -Pasc -o -d -o p -o -f -o 1 filename
```

cr_mapper プログラムは、**stdin** から文字を読み取り、**stdout** に書き込みます。改行 (16 進の A、または 10 進の 10) の読み取りおよび書き込みを行うたびに、このプログラムは復帰 (16 進の D) を書き込みます。

/etc/qconfig ファイル:

/etc/qconfig 構成ファイルは、任意のテキスト・エディターを使用して編集できます。

スプーラーの動作を停止させるなどして妨害することなく /etc/qconfig ファイルを編集できるかどうかについては、暗黙のルールがあります。

ジョブの処理中に /etc/qconfig ファイルを編集してはなりません。特に、ご使用のシステムに多数の (25 台を超える) プリンターが存在し、これらのプリンターの使用頻度が全般に高い場合に、このことが当てはまります。新規のジョブ記述ファイル (JDF) が存在するという通知を **qdaemon** が **enq** から受け取ると、**qdaemon** は /etc/qconfig と /etc/qconfig.bin (/etc/qconfig のバイナリー・バージョン) の両方の日付を検査します。/etc/qconfig が /etc/qconfig.bin より新しい場合は、現在実行中のジョブすべてが処理を完了するまで、**qdaemon** は新規ジョブ (前述のファイルが検査される原因となったジョブを含む) を受け入れません。ジョブが処理を完了すると、**qdaemon** は /etc/qconfig.bin の新規バージョンを作成します。

ジョブの処理中に **qdaemon** がこの状態になった場合は、スプーラーがハングしている可能性があります。この状態で /etc/qconfig を変更した場合に、いずれかのプリンターがまだ出力を生成している場合は、システムをそのままにしておいて、すべてのジョブが処理を完了した後でシステムの動作が元に戻るかどうか確認することが最善のオプションです。出力を生成しているプリンターがない場合、またはスプーラーがハングしていると考えられる場合は、274 ページの『クリーンアップと再始動』を参照してください。

注: ジョブの処理中に /etc/qconfig が変更されないようにしてください。テキスト・エディターを使用して /etc/qconfig を編集し、ファイルの新しいバージョンをディスクに書き込む方法のほかに、**smit** コマンドを使用してキューの属性またはパラメーター値を変更することによって、同じ効果が得られます。

テキスト・エディターによるキューの作成

root ユーザーは、テキスト・エディターを使用して /etc/qconfig を編集し、キューを定義できます。これを行ってはならない状況の 1 つとして、スプーラー・キューのバックエンドが **piobe** である場合があります。**piobe** をバックエンドとして使用するキューには、関連した仮想プリンター定義が必要です。この状況では、root ユーザーは **smit** コマンドを使用してキューを作成する必要があります。**smit** コマンドを使用すると、いくつかのプログラムが実行されて、仮想プリンター定義が作成されます。

透過印刷

ほとんどの端末装置には、シリアル・プリンターに接続できる補助ポートがあります。これらの端末装置は、補助と透過の 2 つの印刷モードをサポートします。

両方の印刷モードがオフならば、端末装置が受信したデータは単に画面に表示されます。補助印刷モードがオンならば、端末装置が受信したデータは画面に表示され、プリンターにも送信されます。透過印刷モードがオンならば、端末装置は受信したデータをプリンターに直接送信し、画面には表示しません。

透過印刷モードでは、端末装置を通常どおり使用できると同時に、ホストから端末の補助プリンター・ポートに接続されたプリンターに、同じシリアル接続を経由して情報が送信されます。これが透過印刷です。透過印刷ソフトウェアは、データのパケットが画面またはプリンターのどちらに向けられたものかを判別し、プリンターに向けられたデータの前に透過印刷モードをオンにするコマンドを発行し、そのデータより後に透過印刷モードをオフにするコマンドを発行します。

端末画面向けのデータには最も高い優先順位が付いており、画面に送信される情報に切れ目がある場合に限ってデータがプリンターに送信されます。連続したデータが端末デバイスに送信された場合は、プリンターには何も送信されません。

補助プリンター・ポートを使用する場合には、プリンターへのフロー制御が問題になります。プリンターが遅れてフロー制御を起動すると、プリンターと端末装置の両方への出力が停止します。透過印刷機能は、プリンター出力を制限してこの状況を回避するためのパラメーターを 3 つ用意しています。

SMIT の「Transparent Print Maximum Characters per Second (透過印刷の 1 秒あたりの最大文字数)」パラメーターは、プリンター・ポートの 1 秒あたりの最大文字数のデータ速度を制限します。この数は、標準的な使用時にプリンターが許容できる最小文字速度に設定する必要があります。

SMIT の「Transparent Print Maximum Character Packet Size (透過印刷の最大文字パケット・サイズ)」パラメーターは、端末装置の出力より前にプリンターへのキューに入れる文字数を制限します。数を小さくするとシステム・オーバーヘッドが増大し、数を大きくするとキー・ストロークのエコーが遅れます。9600 ボーの場合は値 **50** を指定してください。

SMIT の「Transparent Print Printer Buffer Size (透過印刷のプリンター・バッファー・サイズ)」パラメーターは、プリンターのバッファー・サイズより少し小さな値に設定する必要があります。一定期間アクティビティーがないと、ドライバーは maxcps 速度に遅れる前に、この数までの文字をプリンターにバースト送信して、印刷バッファーをいっぱいにします。

プリンターのオン/オフ文字列も SMIT を使用して設定します。端末装置の補助ポートとプリンターの間にケーブルを接続する必要があります。端末装置の補助ポートとプリンターのボー・レートが同じであり、プリンターと端末装置の補助ポートが同じハンドシェーク・モードを使用する必要があります。また、補助ポートを使用可能に設定する必要があります。ご使用の端末装置が直接サポートされないものである場合は、端末装置のエスケープ・シーケンスに関する知識が必要です。

接続に関する情報、エスケープ・コード、およびサポートされるハンドシェーク・モード (例: busy/ready または RTS/CTS) については、ご使用の端末装置とプリンターのマニュアルを参照してください。プリンター・デバイス (例: **xtty1**) は、`/etc/inittab` ファイルまたは `/etc/ttys` ファイル内に存在していてはならず、使用可能に設定されていてはなりません。

透過印刷の活動化については、『端末接続プリンターの構成』を参照してください。

RAN に接続したプリンターまたはプロッターの構成

128 ポート非同期アダプター RAN に接続したプリンターまたはプロッターを定義し、構成する手順を示します。

1. root ユーザー権限が必要です。
2. 128 ポート非同期通信制御機構を取り付けて定義し、使用可能にする必要があります。

3. 少なくとも 1 つの RAN を接続する必要があります。
4. RAN のノード ID を設定します。

注: 印刷スプーラー・サブシステムを正しく操作するためには、8 ピンまたは 10 ピンの RJ-45 ケーブルを使用して、シリアル・プリンターとプロッターを RAN に接続する必要があります。

手順は次のとおりです。

1. プリンター/プロッター・デバイスを 128 ポート RAN に追加するには、**smit pdp** 高速パスを使用して、「**Printer/Plotter Devices (プリンター/プロッター・デバイス)**」メニューにアクセスします。
2. 「**Add a Printer/Plotter (プリンター/プロッターの追加)**」を選択します。
3. 画面に表示されるプリンターおよびプロッターのタイプのリストから適切なプリンター・デバイスを選択し、Enter キーを押します。この例では、次の選択を行いました。

   ```
   osp      Other serial printer (他のシリアル接続プリンター)
   ```

4. RAN タイプに応じて「**rs232**」または「**RS-422**」を選択します。
5. 画面に表示された使用可能な RAN から選択を行います。 RAN が表示されていない場合、または「defined (定義済み)」の状態で表示されている場合は、RAN の構成、ケーブル接続、およびセットアップを再度確認してください。この例では、次の選択を行いました。

   ```
   sa4 Available 00-03-21 16-Port RAN EIA-232 for 128-Port Adapter
   (sa4 使用可能 00-03-21 128 ポート非同期通信制御機構用の 16 ポート非同期通信ノード (EIA-232))
   ```

6. 表示されたダイアログ・フィールド内で、目的のプリンター/プロッター・デバイスの属性を追加または変更できます。
7. 完了したら、「**Do (実行)**」を選択します。

端末接続プリンターの構成

端末接続プリンターを構成できます。

1. 128 ポート・アダプターが正しく構成され、作動可能であること。
2. 3151 ディスプレイが 128 ポート・アダプターに接続され、作動可能であること。
3. ASCII ディスプレイが 128 ポート・アダプターに接続され、作動可能であること。
4. 端末接続印刷用のプリンター・ソフトウェアがインストールされていること。

ハードウェア要件

- IBM 8 ポートまたは 128 ポート・アダプター
- ASCII ディスプレイ (IBM 3151 ディスプレイがこのハードウェアとして使用されます)
- シリアル・プリンター
- EIA 232 シリアル・ケーブルおよびオス・メス変換器

現在使用されている ASCII 端末の多くは、プリンターを接続するための補助のシリアル・ポートまたはパラレル・ポートを備えています。このタイプの接続を使用すれば、管理者は貴重なコンピューター・リソースを共有でき、またプリンターを可能な限りユーザーの近くに移動できるので、ユーザーの生産性と効率を高めることができます。ここでは、このタイプの端末接続プリンターをこのオペレーティング・システム環境で構成するために必要なハードウェア要件、および前提条件について説明します。

端末接続プリンターのハードウェアのセットアップ

端末接続プリンターのハードウェアをセットアップする際には、いくつかのステップを行う必要があります。

システムにプリンターを追加する前に、次の手順を行います。

1. EIA 232 モデム・ケーブル (IBM ケーブル D) を使用して、端末装置の補助ポートにシリアル・プリンターを接続します。ヌル・モデム・ケーブルは、IBM 3151 端末装置の AUX ポートには必要ありません。
2. ご使用のプリンターの回線速度、ワード長 (または 1 文字あたりのビット数)、パリティー (奇数、偶数、なし、スペース、マーク)、およびストップ・ビット数の設定値をメモに取っておきます。

端末接続プリンター用の補助ポートの構成

補助ポートを端末接続プリンター用に構成できます。

IBM 3151 端末装置で、次の手順を行います。

1. Ctrl キーと Setup キーを同時に押しながら、端末装置の電源をオンにします。3151 の画面に「**SETUP (セットアップ)**」メニューが表示されます。
2. Send キーを使用して、「**KEYBOARD/PRINTER (キーボード/プリンター)**」オプションが表示されるまでメニュー画面間を移動します。「**PRINTER (プリンター)**」オプションに、前にメモを取っておいたプリンター設定値を入力します。
3. 「**FUNCTION (機能)**」画面が表示されるまで、Send キーを押します。「**Save (保管)**」オプションを選択し、Spacebar (スペース) を押して構成を保管します。

端末接続プリンターの印刷キューの追加

端末接続プリンターへの印刷キューを追加できます。

この時点で、プリンターを端末装置に物理的に接続し、正しいプリンター設定値を使用して補助ポートを構成しました。次の手順では、ホスト上で端末接続プリンターにアクセスするローカル印刷キューを作成する方法について説明します。

1. root または **printq** 管理者グループのメンバーとしてログオンします。
2. **smit mkpq** 高速パスを使用して「**Add a Print Queue (印刷キューの追加)**」メニューにアクセスします。

   注: この画面の内容は、ホストにインストールされているプリンター・ソフトウェアによって異なる場合があります。
3. 「**ascii**」オプションを選択します。「**Printer Type (プリンター・タイプ)**」画面が表示されます。
4. 該当するプリンター製造メーカー、または「**Other (その他)**」を選択します。この例では、「**IBM**」を選択しました。別の「**Printer Type (プリンター・タイプ)**」画面が表示されます。
5. 該当するプリンター・モデルを選択します。この例では、「**ibm2380-2**」を選択しました。「**TTY Name (TTY 名)**」画面が表示されます。
6. プリンターが接続されている端末装置の TTY を選択します。この例では、「**tty0**」を選択しました。「**Add a Print Queue (印刷キューの追加)**」画面が表示されます。
7. 「**Name of new PRINT QUEUE to add (追加する新しい印刷キュー名)**」フィールドに端末接続印刷キューの記述名を入力し (例: `tty0asc`)、Enter キーを押します。印刷キューは作成されません。

端末接続プリンターのテスト

端末接続プリンターをテストして、プリンターが正しく機能していることを確認できます。

1. プリンターが正しく機能していることを確認するには、**lpstat** コマンドを発行して、システム上の使用可能な印刷キューすべてをリストします。次のような出力が表示されます。

```
# lpstat
Queue    Dev    Status   Job Files   User   PP %   Blks   Cp  Rnk
-----    ---    ------   ---------   ----   ----   ----   --  ---
4019g1   lp0    READY
4019ps   lp1    READY
tty0asc  tty0   READY
```

2. 次の **enq** コマンドを使用して、プリンターに ASCII ファイルを送信します。

```
enq -P tty0asc /etc/qconfig
```

端末装置の表示や機能を変更せずに、ファイルが印刷されることが必要です。

プリンター固有の情報

ヘッダー・ページとトレーラー・ページのフォーマットと内容は、プロトタイプ・テキストを含むファイルを編集することによってカスタマイズできます。

プロトタイプ・テキストを含むファイルは、/usr/lib/lpd/pio/burst ディレクトリーにあります。ファイル名のフォーマットは *X.yyy* で、*X* はヘッダー・ページを示す **H**、またはトレーラー・ページを示す **T** です。*yyy* は、データ・ストリームのタイプを示します。**ascii** は ASCII、**ps** は PostScript、**gl** はプロッター・エミュレーションを表しています。例えば、H.ascii という名前のファイルは ASCII で印刷されるヘッダー・ページのプロトタイプ・テキスト、T.ps は PostScript で印刷されるトレーラー・ページのプロトタイプ・テキストです。テキスト・ファイル内で使用されるエスケープ・シーケンスは、**%** (パーセント) 文字で始まり、**pioburst** コマンドとともに記述されます。詳しくは、**pioburst** コマンドを参照してください。

次に、プリンターとキュー・システムを構成してセットアップするために必要になることがある、特定のプリンターに固有の情報を示します。

IBM パーソナル・プリンター II モデル 2380、2381、2390、2391、2380-2、2381-2、2390-2、2391-2

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

ギリシャまたはトルコで購入したプリンターには、ギリシャ語またはトルコ語のコード・ページが付属しています。ギリシャ語またはトルコ語の文字を印刷するには、コード・ページが使用可能であることをシステムに通知する必要があります。このためには、次の手順で行います。

1. root ユーザーとして smit chpq と入力します。
2. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。
3. 「**COUNTRY (国)**」を該当する国選択に変更します。

IBM 3812 モデル 2 ページ印刷装置

プリンターおよびキュー・システムの製品固有の情報を示します。

プリンターのディスケット・ドライブにフォント・ディスケット (フィーチャー #3155) が挿入されていることが前提です。ギリシャ語またはトルコ語の文字のサポートを使用する場合は、言語グループ 3 フォント・ディスケットがドライブに挿入されていることが前提です。

フォントは、プリンター内部のフォント・ディスケットからプリンター・メモリーにロードされます。どのフォントがロード済みか、またメモリー内のフォントが印刷ジョブによって破壊されたためにディスケット

からの再ロードが必要かどうかの記録が、システムによって維持されます。プリンターをオフにして再度オンにした場合は、**-F !** フラグとデバイス名を指定して **splp** コマンドを実行します。これにより、フォントをプリンターのメモリーにロードする必要があることをシステムに通知します。

3812 モデル 2 ページ印刷装置は、8-1/2 x 11 インチのデフォルト・サイズ以外の用紙に印刷できます。SMIT を使用して用紙サイズを変更できます。単一の印刷ジョブに対して用紙サイズを変更するには、**qprt** コマンドに **-Q** フラグを指定します。

ギリシャ語またはトルコ語の文字のサポートが必要な場合は、次の手順で行います。

1. 言語グループ 3 フォント・ディスケットをプリンターに挿入する必要があります。
2. root ユーザーとして `smit chpq` と入力します。
3. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。フォント・ディスケットについては、次のように指定します。
   - `CP851` (ギリシャ語文字を印刷する場合)
   - `CP853` (トルコ語文字を印刷する場合)

IBM 3816 ページ印刷装置

プリンターおよびキュー・システムの製品固有の情報を示します。

プリンターのディスケット・ドライブにフォント・ディスケット (フィーチャー #7652) が挿入されていることが前提です。

フォントは、プリンター内部のフォント・ディスケットからプリンター・メモリーにロードされます。どのフォントがロード済みか、またメモリー内のフォントが印刷ジョブによって破壊されたためにディスケットからの再ロードが必要かどうかの記録が、システムによって維持されます。プリンターをオフにして再度オンにした場合は、**-F !** フラグとデバイス名を指定して **splp** コマンドを実行します。これにより、フォントをプリンターのメモリーにロードする必要があることをシステムに通知します。

3816 ページ印刷装置は、8-1/2 x 11 インチのデフォルト・サイズ以外の用紙に印刷できます。SMIT を使用して用紙サイズを変更できます。単一の印刷ジョブに対して用紙サイズを変更するには、**qprt** コマンドに **-Q** フラグを指定します。

IBM 4019 ページ印刷装置および 4029 ページ印刷装置

それぞれのプリンターおよびキューごとに、製品固有の情報を示します。

システムは、手操作による介入を必要とせずに、IBM ASCII または PCL のエミュレーション・データ・ストリームを選択します。

ページ印刷装置は、8-1/2 x 11 インチのデフォルト・サイズ以外の用紙、およびデフォルトの #10 封筒以外の封筒に印刷できます。用紙または封筒のサイズは、SMIT を使用して変更できます。単一の印刷ジョブに対して用紙または封筒のサイズを変更するには、**qprt** コマンドに **-Q** フラグを指定します。

フォント・カードが取り付けられている場合は、システムにそのことを通知する必要があります。これを処理するには、次の手順で行います。

1. root ユーザーとして `smit chpq` と入力します。
2. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。

- 上部カード・スロットに取り付けられているフォント・カードを指定します。
- 下部カード・スロットに取り付けられているフォント・カードを指定します。

  注: nn 変数の値はフォント・カード ID で、フォント・カードの矢印のすぐ上にある 2 桁の番号です。

PostScript オプションが 4019 ページ印刷装置にインストールされていて、PostScript オプションが自動エミュレーション・モード切り替えをサポートしている場合は、PostScript モードとエミュレーション・モードを自動的に切り替えるようにプリンターを構成でき、またエミュレーション・モード間での切り替えも可能です。ご使用のプリンターにインストールされている PostScript オプションがこの機能をサポートしているかどうか調べるには、ご使用の PostScript オプションに付属の操作ガイド・マニュアルを参照してください。または、PostScript インタープリターがバージョン 52.3 以降かどうかを確認します。このためには、PostScript モードでプリンターの電源をオンにしたときに印刷される (使用不可に設定されていなければ) PostScript 始動ページを調べます。自動エミュレーション・モード切り替えが示されている場合は、次のようにしてシステムに通知します。

1. root ユーザーとして `smit chpq` と入力します。
2. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。「**AUTOMATIC MODE SWITCHING for PostScript (PostScript の自動モード切り替え)**」属性に「**yes (はい)**」を指定します。

   注:

   a. 日本語文字を印刷するには、日本語基本システム・ロケール・パッケージをインストールする必要があります。

   b. 日本語文字は PostScript オプションを使用して印刷できません。

   c. マルチバイト文字を印刷するには、**qprt** コマンドの **-F** オプションを使用して、16x16 ドットまたは 32x32 ドットのフォントを指定します。次に例を示します。

   ```
   qprt -Pkji -F'RomanKn23,Kanji23,IBM_JPN23' file
   ```

IBM 4037 および IBM 4039 レーザー・プリンター

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

システムは、手操作による介入を必要とせずに、IBM ASCII または PCL のエミュレーション・データ・ストリームを選択します。

これらのプリンターは、8-1/2 x 11 インチのデフォルト・サイズ以外の用紙に印刷できます。SMIT を使用して用紙サイズを変更できます。単一の印刷ジョブに対して用紙サイズを変更するには、**qprt** コマンドに **-Q** フラグを指定します。

注:

1. 日本語文字を印刷するには、日本語基本システム・ロケール・パッケージをインストールする必要があります。
2. 日本語文字は PostScript オプションを使用して印刷できません。
3. マルチバイト文字を印刷するには、**qprt** コマンドの **-F** オプションを使用して、16x16 ドットまたは 32x32 ドットのフォントを指定します。次に例を示します。

   ```
   qprt -Pkji -F'RomanKn23,Kanji23,IBM_JPN23' file
   ```

IBM 4072 ExecJet プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

ギリシャ語またはトルコ語文字のサポートが必要で、NLS I フォント・カードが取り付けられている場合は、フォント・カードが取り付けられていることをシステムに通知する必要があります。

1. root ユーザーとして `smit chpq` と入力します。
2. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。
   - 左側のカード・スロットに取り付けられているフォント・カードを指定します。
   - 右側のカード・スロットに取り付けられているフォント・カードを指定します。

   注: nn 変数の値は、フォント・カードのパーツ・ナンバーの末尾 2 桁です。

IBM 4076 インクジェット・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

システムは、手操作による介入を必要とせずに、IBM ASCII または PCL のエミュレーション・データ・ストリームを選択します。

インクジェット・プリンターは、8-1/2 x 11 インチのデフォルト・サイズ以外の用紙、およびデフォルトの #10 封筒以外の封筒に印刷できます。用紙および封筒のサイズは、SMIT を使用して変更できます。単一の印刷ジョブに対して用紙または封筒のサイズを変更するには、**qprt** コマンドに **-Q** フラグを指定します。

IBM Proprinter モデル 4201-3、4202-3、4207-2、4208-2

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

ギリシャ語またはトルコ語文字のサポートが必要で、該当するフォント・ディスケットが使用可能ならば、root ユーザーとして次のように入力してフォントをインストールします。

piofontin -t *PrinterType* **-c** *CodePage* [**-d** *DeviceName*]

具体的には、このコマンドを発行する際に、**PrinterType** パラメーターには 4201-3、4202-3 を指定し、**CodePage** パラメーターには 851 (ギリシャ語) または 853 (トルコ語) を指定します。**DeviceName** パラメーター (例えば `/dev/fd1`) は、ディスケット機構が `/dev/fd0` (標準 3.5 インチ・ディスケット・ドライブ) でない場合のみ必要です。

注: ギリシャ語またはトルコ語のフォントがインストールされている場合は、プリンターの電源をオフにしてオンにするたびに、次のように入力してフォントをダウンロードする必要があります。

```
splp -F! xxx
```

ここで、*xxx* はプリンターのデバイス名 (例: **lp0**) です。このコマンドは、ギリシャ語またはトルコ語のフォントをプリンターにダウンロードするようにシステムに指示します。

IBM 4208-502、IBM 5572-B02、IBM 5573-H02、および IBM 5579-H02/K02

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

日本語および Proprinter のデータ・ストリームがサポートされています。日本語文字を印刷するには、次のように、**qprt** コマンドの **-F** オプションに 24x24 ドットのフォントを指定します。

```
qprt -Pkji -F/usr/lib/X11/fonts/JP/IBM_JPN17.pcf.Z file
```

IBM 4216-031 型ページ印刷装置

プリンターおよびキュー・システムの製品固有の情報を示します。

「*Personal Page Printer II Model 031 Guide to Operations*」の付録 B に指示されているとおり、プリンターの後部にあるスイッチは、自動エミュレーション切り替えに設定する必要があります。

システムは、手操作による介入を必要とせずに、PostScript、Proprinter XL エミュレーション、PCL エミュレーション、または Diablo 630 エミュレーションを選択します。

PostScript データ・ストリームを使用して ASCII ファイルを印刷することもできます。

プリンターをシリアル・ポートに接続する場合は、特殊な PostScript ファイルをプリンターに送信して、プリンターのボー・レート、パリティー、およびプロトコルを設定する必要が生じることがあります。「*Personal Page Printer II Model 031 Guide to Operations*」にはサンプル・ファイルが記載されています。

プロプリンター XL、PCL エミュレーション、および Diablo 630 エミュレーションの印刷キューは、各印刷ファイルの先頭 2 文字を検査します。先頭 2 文字が **%!** ならば、ファイルは PostScript ファイルであると想定され、印刷キュー名によって暗黙に指定される仮想プリンターの代わりに、PostScript 用の仮想プリンターが使用されます。PostScript 用の印刷キューを構成していない場合、印刷ジョブは失敗します。

PostScript ファイルを PostScript ファイルとしてではなく ASCII ファイルとして印刷したい場合は、IBM プロプリンター XL、PCL エミュレーション、または Diablo 630 エミュレーションの印刷ジョブを実行依頼する前に、**-da** フラグとパラメーターを **qprt** コマンドに指定してください。

IBM 4216-510 および IBM 5327-011

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

日本語データ・ストリームがサポートされています。日本語文字を印刷するには、次のように、**qprt** コマンドの **-F** オプションに 24x24 ドットのフォントを指定します。

```
qprt -Pkji -F/usr/lib/X11/fonts/JP/IBM_JPN17.pcf.Z file
```

IBM 4234 印刷装置

プリンターおよびキュー・システムの製品固有の情報を示します。

プリンターの制御パネルで適切な文字セット (印刷される言語) が選択されていることが前提です。プリンターの操作指示マニュアルにある、印刷言語の設定に関する説明を参照してください。

選択した文字セットが 02 (PC マルチリンガル) でない場合は、次の手順で行います。

1. 次の SMIT 高速パスを入力します。

   ```
   smit chpq
   ```

2. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。
3. 「**PRINTED LANGUAGE (印刷される言語)**」属性の文字セットを選択します。

IBM 5202 クワイエット・ライター III

プリンターおよびキュー・システムの製品固有の情報を示します。

このプリンターはフォント・カートリッジの存在を検出しますが、ホスト・システムはその存在を検出できません。次のいずれかの場合には、フォント・カートリッジについてシステムに通知する必要があります。

- コード・ページ 850 のフォントを含むフォント・カートリッジを接続する。
- そのフォントを使用して印刷する。
- コード・ページ 850 (ヨーロッパ文字) に固有の文字を印刷する。

このフォント・カートリッジについてシステムに通知するには、次の手順で行います。

1. 次の SMIT 高速パスを入力します。

   ```
   smit chpq
   ```

2. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。
3. 「**CODE PAGE 850 (コード・ページ 850)**」属性に対して「**yes (はい)**」を指定します。

IBM 5204 印刷装置

プリンターおよびキュー・システムの製品固有の情報を示します。

ギリシャ語またはトルコ語をサポートするフォント・カートリッジが取り付けられている場合は、次の手順で行います。

1. root ユーザーとして `smit chpq` を入力します。
2. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。
3. 「**Font Cartridge (フォント・カートリッジ)**」属性にフォント・カートリッジの部品番号を指定します。サポートされる値は 1301598 (ギリシャ語) と 1301614 (トルコ語) です。

IBM 5575-B02/F02/H02 および IBM 5577-B02/F02/FU2/G02/H02/J02/K02

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

日本語文字を印刷するには、次のように、**qprt** コマンドの **-F** オプションに 24x24 ドットのフォントを指定します。

```
qprt -Pkji -F/usr/lib/X11/fonts/JP/IBM_JPN17.pcf.Z file
```

IBM 5584-G02/H02、IBM 5585-H01、IBM 5587-G01/H01、および IBM 5589-H01

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

日本語文字を印刷するには、次のように、**qprt** コマンドの **-F** オプションに 24x24 ドットのフォントを指定します。

```
qprt -Pkji -F/usr/lib/X11/fonts/JP/IBM_JPN17.pcf.Z file
```

IBM 6252 印刷装置

プリンターおよびキュー・システムの製品固有の情報を示します。

取り付けられている印刷バンドのアクティブ・コード・ページがコード・ページ 850 でない場合は、次のようにしてシステムに通知する必要があります。

1. root ユーザーとして `smit chpq` を入力します。
2. 該当する印刷キューを選択し、「**Change/Show Characteristics (特性の変更/表示)**」メニューの「**Printer Setup (プリンターの設定)**」を選択します。
3. 「**ACTIVE CODE PAGE (アクティブ・コード・ページ)**」属性に対して、コード・ページを選択します。

IBM ネットワーク・カラー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

事前定義ファイルは、IBM ネットワーク・カラー・プリンター 2.0 以上のコントローラー・コード・レベルをサポートします。

ネットワーク・カラー・プリンター PS キューのみを使用する場合に有効な追加の **qprt** オプション・フラグは、次のとおりです。

項目	説明
-e #	輝度レベルを指定します。
	0 デフォルト・プリンター設定
	1 最も明るい
	2 2 番目に明るい
	3 3 番目に明るい
	4 通常
	5 3 番目に暗い
	6 2 番目に暗い
	7 最も暗い
-E #	仕上げを指定します。
	0 デフォルト・プリンター設定
	1 通常
	2 マット紙
	3 光沢紙
-k #	カラー型を指定します。
	0 デフォルト・プリンター設定
	1 CMYK
	2 グレースケール
-K #	カラー・レンダリング・ディクショナリーを指定します。
	0 デフォルト・プリンター設定
	1 スキャナー
	2 高輝度
	3 写真
	4 プレゼンテーション
	5 モニター
	6 一色
-S #	印刷モードを指定します。
	0 デフォルト・プリンター設定
	1 写真品質
	2 ビジネス・グラフィックス

IBM ネットワーク・カラー・プリンターには、次の有効なキュー名が存在します。

項目	説明
ibmcolor_direct	1.03 または 1.1 コントローラー・コードおよび 16MB のメモリー。
ibmcolor_print	1.03 または 1.1 コントローラー・コードおよび 32MB または 48MB のメモリー。
ibmncp_direct	2.0 以上のコントローラー・コードおよび 16MB のメモリー。
ibmncp_print	2.0 以上のコントローラー・コードおよび 32MB または 48MB のメモリー。

注: IBM ネットワーク・カラー・プリンターのコントローラー・コード・レベルとメモリー容量を判別するには、プリンター・オペレーター・パネルから構成ページを印刷します。「GENERAL INFO」の下にある構成ページを参照してください。メモリーは 2 番目の項目で、3 番目の項目はコントローラー・コードのバージョンです。AIX 上の事前定義ファイルは、IBM ネットワーク・カラー・プリンター 2.0 以上のコントローラー・コード・レベルのみをサポートします。

IBM ネットワーク・プリンター 12、17、および 24

プリンターおよびキュー・システムの製品固有の情報を示します。

IBM ネットワーク・プリンターは、8 1/2 x 11 インチのデフォルト・サイズ以外の用紙に印刷できます。用紙のサイズは、SMIT を使用して変更できます。これについて詳しくは、 158 ページの『IBM ネットワーク・カラー・プリンター』を参照してください。単一の印刷ジョブに対して用紙サイズを変更するには、**qprt** コマンドに **-Q** フラグを指定します。

行スペースに収まる数 (6 または 8 行/インチ) より多くの行を 1 ページに印刷するには、ページあたりの行数を指定します。より多くの行がページに収まるように、行送りが圧縮されます。例えば、行送りが 6 行/インチならば、コマンド `qprt -l 66 FileName` を入力すると、ファイル `FileName` がデフォルトの 60 行/ページではなく 66 行/ページで印刷されます。

IBM ネットワーク・プリンター 12、17、および 24 は、次のフォントとピッチをサポートします。

許可されるピッチ値およびフォント

フォント	許可されるピッチ値
courier	10、12、または 17 ピッチ
courier-bold	10、12、または 17 ピッチ
courier-italic-bold	10、12、または 17 ピッチ
lettergothic	10、12、または 17 ピッチ
lettergothic-bold	10、12、または 17 ピッチ
lettergothic-italic	10、12、または 17 ピッチ
lineprinter	17 ピッチ

例えば、コマンド `qprt -s Lineprinter -p 17 FileName` を入力すると、ファイル `FileName` が 17 ピッチ (17 字/インチ) でライン・プリンターのフォントを使用して印刷されます。

次に、アラビア語、ギリシャ語、およびヘブライ語の書体を示します。

アラビア語、ギリシャ語、およびヘブライ語の書体のタイプ

アラビア語の書体	ギリシャ語の書体	ヘブライ語の書体
typing typing-italic typing-bold typing-bold-italic	grcour grcour-oblique grcour-bold grcour-bold-oblique	shalom shalom-bold shalom-italic shalom-bold-italic
rokaa rokaa-italic rokaa-bold rokaa-bold-italic	grhelvet grhelvet-bold grhelvet-oblique grhelvet-bold-oblique	narkisstam narkisstam-bold narkisstam-italic narkisstam-bold-italic
setting setting-italic setting-bold setting-bold-italic	grtimesnr grtimesnr-bold grtimesnr-oblique grtimesnr-bold-oblique	narkissim narkissim-bold narkissim-italic narkissim-bold-italic

注: アラビア語、ギリシャ語、またはヘブライ語の書体を使用する際には、フラッシュ SIM またはプリンターのハード・ディスクにフォントを必ずダウンロードしてください。

IBM ネットワーク・プリンター 12、17、および 24 は、次の出力ビンをサポートします。出力ビンには、**qprt** コマンドの **(-=)** フラグを使用してアクセスできます。次の表に、可能な値および対応する出力先の出力ビンを示します。

IBMネットワーク・プリンター 12 の場合の可能な値および対応する出力先の出力ビンのリスト

-= 値 (#)	**出力先の出力ビン**
0	メイン出力トレイ
1	フェースアップ (後部) トレイ

-= 値 (#)	**出力先の出力ビン**
0	メイン出力トレイ
1	メールボックス・ビン 1
2	メールボックス・ビン 2
3	メールボックス・ビン 3
4	メールボックス・ビン 4
5	メールボックス・ビン 5
6	メールボックス・ビン 6
7	メールボックス・ビン 7
8	メールボックス・ビン 8
9	メールボックス・ビン 9
10	メールボックス・ビン 10
50	オフセット・ビン

IBMネットワーク・プリンター 24 の場合の可能な値および対応する出力先の出力ビンのリスト

-= 値 (#)	**出力先の出力ビン**
0	自動出力ビン
1	メイン出力トレイ
2	フェースアップ (後部ビン)
3	上段フィニッシャー・ビン・フェースダウン
4	中段フィニッシャー・ビン・フェースダウン
5	下段フィニッシャー・ビン・フェースダウン
6	上段フィニッシャー・ビン・フェースアップ
7	中段フィニッシャー・ビン・フェースアップ
8	下段フィニッシャー・ビン・フェースアップ
9	自動フィニッシャー・ビン・フェースダウン

次の追加 **qprt** オプション・フラグは、ネットワーク・プリンター 24 の PS キューまたは PCL キューを使用する場合にのみ有効です。

ネットワーク・プリンター 24 を使用することによって使用できる追加フラグのリスト

項目	説明
-e #	ステープルおよび丁合いを指定します。**-e #** オプション・フラグは、フェースダウン・フィニッシャー・ビンが選択された場合のみ有効になります。詳しくは、**-=** オプション・フラグを参照してください。 **0** デフォルト・プリンター設定 **1** 縦長ステープル **2** 横長ステープル **3** 縦長 2 カ所ステープル **4** 横長 2 カ所ステープル **5** ジョブ終了時にオフセット用紙揃え **6** セット終了時にオフセット用紙揃え **7** ステープルまたは丁合いなし

IBM ネットワーク・プリンター 12、17、および 24 上には、次のような有効なキュー名が存在します。

ネットワーク・プリンターでの有効なキュー名のリスト

項目	説明
TEXT	改行と復帰の処理を必要とするデータ。
PASS	さらに処理する必要のないデータ。

IBM InfoPrint 20

プリンターおよびキュー・システムの製品固有の情報を示します。

プリンターの製品固有の情報

項目	説明
-q #	印刷品質オプションを指定します。InfoPrint 20 用の **-q** オプションは、次のとおりです。 **600** 600 dpi を指定します。 **1200** 1200dpi を指定します。これにより、実際の 2 倍のプリンター解像度があるように見える印刷結果が得られます。 1200dpi の印刷品質は、1200dpi で作成されたイメージなどのデータを印刷する場合に推奨されます。
-u #	給紙機構を設定します。InfoPrint 20 用の **-u** オプションは、次のとおりです。 **0** プリンター上で選択された現行の給紙機構を使用 **1** トレイ 1 **2** 補助トレイ - 用紙用の手差し **3** 補助トレイ - 封筒用の手差し **4** 補助トレイ - 自動 **5** トレイ 2 **6** 封筒トレイ **7** トレイ 3 **9** 2000 枚給紙ドロワー
-= #	出力用紙処理のタイプを設定します。InfoPrint 20 用の **-=** オプションは、次のとおりです。 **0** メイン **1** オフセット・オプションを設定したメイン

プリンターの製品固有の情報

項目	説明
-Q #	印刷ジョブの用紙サイズを指定します。InfoPrint 20 用の **-Q** オプションは、次のとおりです。 **1** レター **2** リーガル **3** フォリオ **4** 11 x 17 **5** A4 **6** B4 **7** A3 **8** 汎用用紙サイズ **9** B5-JIS **10** A5 **11** エグゼクティブ **12** ステートメント **13** ハガキ **14** Monarch 封筒 **15** COM10 封筒 **16** C5 封筒 **17** DL 封筒 **18** 汎用封筒サイズ
-s *Name*	*Name* 変数によって書体を指定します。 InfoPrint 20 用の書体は、次のとおりです。 • courier • courier-bold • courier-italic • courier-bold-italic • gothic • lineprinter • gothic-bold • gothic-italic • prestige elite • prestige elite-bold • prestige elite-italic • prestige elite-bold-italic

IBM InfoPrint 32 プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

プリンターの製品固有の情報:

項目	説明
-= #	出力処理のタイプ。 **0** デフォルト・プリンター設定 **1** メイン (フェースダウン) **2** フェースアップ (後部ビン) **3** フィニッシャー・ビン 1 (フェースダウン上段) **4** フィニッシャー・ビン 2 (フェースダウン中段) **5** フィニッシャー・ビン 3 (フェースダウン下段) **9** 任意のフィニッシャー・ビン (フェースダウン)

プリンターの製品固有の情報:

項目 **説明**

-e # ステープル/丁合いを指定します。このオプション・フラグは、フェースダウン・フィニッシャー・ビンが選択された場合のみ有効になります (**-=** オプション・フラグを参照)。

- **0** デフォルト・プリンター設定
- **1** 縦長ステープル
- **2** 横長ステープル
- **3** 縦長 2 カ所ステープル
- **4** 横長 2 カ所ステープル
- **5** ジョブ終了時にオフセット用紙揃え
- **6** セット終了時にオフセット用紙揃え
- **7** ステープルまたは丁合いなし

-k # RePro 丁合いコピーの印刷部数を指定します。この機能を実行するには、プリンターにハード・ディスクが備わっている必要があります。

-s # Name 変数によって書体を指定します。例えば、courier、courier-bold、courier-italic courier-bold-italic、lettergothic、lineprinter、lettergothic-bold、lettergothic-italic、prestigeelite、prestigeelite-bold、prestigeelite-italic、prestigeelite-bold-italic などがあります。

-u # 給紙機構を次のいずれかに設定します。

- **0** プリンター上で選択された現行の給紙機構を使用
- **1** トレイ 1
- **2** 補助トレイ - 手差し (用紙)
- **3** 補助トレイ - 手差し (封筒)
- **4** 補助トレイ (自動)
- **5** トレイ 2
- **6** 封筒トレイ
- **7** トレイ 3
- **8** トレイ 4
- **9** トレイ 5

-z # 右回りに 4 分の 1 回転の単位で、*Value* 変数の指定に従ってページ・プリンターの出力を回転します。長さ (-l) と幅 (-w) の値は、自動的に正しく調整されます。

- **0** 縦長
- **1** 横長
- **2** 縦長 (逆方向)
- **3** 横長 (逆方向)

-Q # 印刷ジョブの用紙サイズを指定します。

- **1** レター
- **2** リーガル
- **3** フォリオ
- **4** 11x17
- **5** A4
- **6** B4
- **7** A3
- **8** 汎用用紙サイズ
- **9** B5-JIS
- **10** A5
- **11** エグゼクティブ
- **12** ステートメント
- **13** ハガキ
- **14** Monarch 封筒
- **15** COM10 封筒
- **16** C5 封筒
- **17** DL 封筒
- **18** 汎用封筒サイズ

IBM InfoPrint 40 プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

プリンターの製品固有の情報:

項目	説明
-= #	出力処理のタイプ。 **0** デフォルト・プリンター設定 **1** メイン (フェースダウン) **2** フェースアップ (後部ビン) **3** フィニッシャー・ビン 1 (フェースダウン上段) **4** フィニッシャー・ビン 2 (フェースダウン中段) **5** フィニッシャー・ビン 3 (フェースダウン下段) **9** 任意のフィニッシャー・ビン (フェースダウン)
-e #	ステープル/丁合いを指定します。このオプション・フラグは、フェースダウン・フィニッシャー・ビンが選択された場合のみ有効になります (**-=** オプション・フラグを参照)。 **0** デフォルト・プリンター設定 **1** 縦長ステープル **2** 横長ステープル **3** 縦長 2 カ所ステープル **4** 横長 2 カ所ステープル **5** ジョブ終了時にオフセット用紙揃え **6** セット終了時にオフセット用紙揃え **7** ステープルまたは丁合いなし
-k #	RePro 丁合いコピーの印刷部数を指定します。この機能を実行するには、プリンターにハード・ディスクが備わっている必要があります。
-s #	Name 変数によって書体を指定します。例えば、courier、courier-bold、courier-italic、courier-bold-italic、lettergothic、lineprinter、lettergothic-bold、lettergothic-italic、prestigeelite、prestigeelite-bold、prestigeelite-italic、prestigeelite-bold-italic などがあります。
-u #	給紙機構を次のいずれかに設定します。 **0** プリンター上で選択された現行の給紙機構を使用 **1** トレイ 1 **2** 補助トレイ - 手差し (用紙) **3** 補助トレイ - 手差し (封筒) **4** 補助トレイ (自動) **5** トレイ 2 **6** 封筒トレイ **7** トレイ 3 **8** トレイ 4 **9** トレイ 5

プリンターの製品固有の情報:

項目	説明	
-Q #	印刷ジョブの用紙サイズを指定します。	
	1	レター
	2	リーガル
	3	フォリオ
	4	11x17
	5	A4
	6	B4
	7	A3
	8	汎用用紙サイズ
	9	BJ-JIS
	10	A5
	11	エグゼクティブ
	12	ステートメント
	13	ハガキ
	14	Monarch 封筒
	15	COM10 封筒
	16	C5 封筒
	17	DL 封筒
	18	汎用封筒サイズ
Queue Names	IBM ネットワーク・プリンター 12、17、24、および InfoPrint 20、32、および 40 上には、次のような有効なキュー名が存在します。	
	TEXT	改行と復帰の処理を必要とするデータ。
	PASS	さらに処理する必要のないデータ。

Canon LASER SHOT LBP-B404PS/Lite

プリンターおよびキュー・システムの製品固有の情報を示します。

Canon LASER SHOT LBP-B404PS/Lite プリンターでは、日本語 PostScript および ASCII データ・ストリームがサポートされます。日本語のテキスト・ファイルは印刷できません。

Canon LASER SHOT LBP-B406S/D/E/G、A404/E、A304E

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

Canon LASER SHOT LBP-B406S/D/E/G、A404/E、および A304E プリンターでは、日本語コード・セットがサポートされます。IBM 5575 エミュレーション・カードは使用しないでください。**lips3** キューは、LIPS II+ モードのモデル LBP-B406S/D,A404 には使用できません。

Dataproducts LZR 2665 レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

Dataproducts LZR 2665 レーザー・プリンター用のデータ・ストリーム (PostScript、Diablo 630) は、コントロール・パネルを使用して手動で選択する必要があります。 PostScript データ・ストリームを使用して ASCII ファイルを印刷することもできます。

Hewlett-Packard LaserJets II、III、IIISi、3005、4、4Si、4Plus、4V、4000、5200、5Si/5Si MX、5Si Mopier、4700 Color、8000 Color、および 8500 Color

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

Hewlett-Packard LaserJet プリンターは、8-1/2 x 11 インチのデフォルト・サイズ以外の用紙に印刷できます。SMIT を使用して用紙サイズを変更できます。単一の印刷ジョブに対して用紙サイズを変更するには、**qprt** コマンドに **-Q** フラグを指定します。

行スペースに収まる数 (6 または 8 行/インチ) より多くの行を 1 ページに印刷するには、ページあたりの行数を指定します。より多くの行がページに収まるように、行送りが圧縮されます。例えば、行送りが 6 行/インチならば、コマンド `qprt -l 66 FileName` を入力すると、`FileName` パラメーターにより指定されたファイルがデフォルトの 60 行/ページではなく 66 行/ページで印刷されます。

HP LaserJet III、IIISi、および 4 は、次のフォントとピッチをサポートします。

フォント	許可されるピッチ
クーリエ・フォント	10、12、または 17 ピッチ
クーリエ-ボールド・フォント	10 または 12 ピッチ
クーリエ-イタリック・フォント	10 または 12 ピッチ
ライン・プリンター・フォント	17 ピッチ

例えば、コマンド `qprt -s Lineprinter -p 17 FileName` を入力すると、ファイル `FileName` が 17 ピッチ (17 字/インチ) でライン・プリンターのフォントを使用して印刷されます。

Hewlett-Packard LaserJet プリンターを RS-422A シリアル・ポートに接続するには、特殊なケーブルが必要です。次のピン配列情報を使用して、ケーブルを組み立てることができます。

ソケット・コネクター (システム側)	信号	ピン・コネクター (デバイス側)
シェル	シールド用接地	1
2	TxA	3
3	RxA	9
4	TxB	18
5	RxB	10
7	信号用接地	7

Hewlett-Packard LaserJet 3005、5200 および 4700 カラー・プリンター

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

フォント名およびピッチ

以下のフォントとピッチ・サイズがサポートされます。

フォント名	許可されるピッチ値
-s クーリエ	-p (1 から 99)
-s クーリエ-太字	-p (1 から 99)
-s クーリエ-イタリック体	-p (1 から 99)
-s ゴシック体	-p (1 から 99)
-s ゴシック体-太字	-p (1 から 99)
-s ゴシック体-イタリック体	-p (1 から 99)
-s ライン・プリンター	-p 17

注: ライン・プリンター・フォントに対してはピッチ 17 のみがサポートされています。

用紙タイプ

以下の用紙タイプ選択は、**qprt** コマンドの **-y** フラグを指定してサポートされます (HP4700, HP5200)。

項目	用紙タイプ	プリンター・タイプ
-y 0	any	
-y 1	普通紙	
-y 2	事前印刷	
-y 3	レターヘッド	
-y 4	透明	
-y 5	事前せん孔	
-y 6	ラベル	
-y 7	ボンド紙	
-y 8	再生紙	
-y 9	カラー紙	
-y 10	ライト	
-y 11	ヘビー	
-y 12	カード・ストック (cardstock)	
-y 13	ラフ紙	
-y 14	ベラム (vellum)	
-y 16	光沢紙	(HP 4700 のみ)
-y 17	重量光沢紙	(HP 4700 のみ)

エンベロープ・サイズ

以下のエンベロープ・サイズは、エンベロープ・トレイが給紙機構 (-u 3 または -u 4 または -u 5) として選択されたとき、サポートされます (HP4700, HP5200)。

項目	説明
-Q 1	Monarch
-Q 2	B5
-Q 3	COM-10
-Q 4	DL
-Q 5	C5

丁合いと印刷部数

プリンターは、印刷ジョブの複数部印刷の丁合いを内部でサポートします。**qprt** コマンドのこの機能は、**-W** フラグと **-S** フラグによって、次に示すように制御されます。

項目	説明
-S !	丁合いオフ
-S +	丁合いオン
-W #	印刷部数

注:

1. この機能は、スプーラーがサポートする **-N** フラグとは異なります。**-N #** フラグを使用すると、印刷ジョブがプリンターに # 回送信されます。**-W #** フラグは印刷ジョブを 1 回送信し、このジョブが # 部印刷されます。
2. この機能は、プリンターに取り付けられているメモリーの容量、および印刷ジョブのサイズによって制限されます。

ジョブ・ストレージ

この機能は、**qprt** コマンドの **-U** コマンド・フラグおよび **-V** コマンド・フラグの指定によりサポートされます。**-U** フラグでは、以下に示すようにジョブ・ストレージ・モードを指定します。

項目	ジョブ・ストレージ・モード	プリンター・タイプ
-U 1	オフ	
-U 2	私用ジョブ	
-U 3	検査と保留	
-U 4	即時コピー	(HP4700 のみ)
-U 5	即時コピー	(HP4700 のみ)
-U 6	保管ジョブ (専用)	(HP4700 のみ)

-V フラグは、専用印刷ピン番号を制御します。

-V [0000-9999]

HP LaserJet 3005

以下の給紙機構選択は、**qprt** コマンドの **-u** フラグを指定してサポートされます。

項目	説明
-u 0	手動給紙多目的トレイ 1
-u 1	給紙トレイ 2
-u 2	給紙トレイ 3
-u 3	多目的トレイ 1 内の封筒給紙機構
-u 4	多目的トレイ 1 内の手動封筒給紙機構
-u 6	多目的トレイ 1 内の給紙機構
-u 7	自動選択

以下の用紙サイズ選択は、**qprt** コマンドの **-Q** フラグを指定してサポートされます。

項目	説明
-Q 1	レター
-Q 2	リーガル
-Q 5	A4
-Q 9	A5
-Q 6	エグゼクティブ
-Q 8	JISB5
-Q 11	Monarch
-Q 12	B5
-Q 13	COM10
-Q 14	DL
-Q 15	C5
-Q 3	カスタム

以下の用紙タイプ選択は、qprt コマンドの **-y** フラグを指定してサポートされます。

項目	説明
-y 0	不特定型
-y 1	ボンド紙
-y 2	CardStock
-y 4	エンベロープ
-y 5	ラベル
-y 6	レターヘッド
-y 7	3 番目に明るい
-y 8	普通紙
-y 9	事前印刷
-y 10	事前せん孔
-y 11	再生紙
-y 12	ラフ紙
-y 13	透明
-y 14	Vellum

HP LaserJet 4700 カラー

以下の給紙機構選択は、**qprt** コマンドの **-u** フラグを指定してサポートされます。

項目	説明
-u 0	手動給紙
-u 1	トレイ 2
-u 2	トレイ 3
-u 3	エンベロープ (トレイ 1)
-u 4	手動封筒給紙機構
-u 6	多目的トレイ 1
-u 7	トレイ 4
-u 8	トレイ 5
-u 9	トレイ 6
-u 10	自動選択

以下の用紙サイズ選択は、**qprt** コマンドの **-Q** フラグを指定してサポートされます。

項目	説明
-Q 1	レター
-Q 2	リーガル
-Q 5	A4
-Q 6	exec
-Q 8	B5 (JIS)
-Q 9	A5
-Q 10	exec (JIS)
-Q 11	8.5 x 13
-Q 12	ステートメント

HP LaserJet 5200

以下の給紙機構選択は、**qprt** コマンドの **-u** フラグを指定してサポートされます。

項目	説明
-u 0	手動給紙
-u 1	トレイ 2
-u 2	トレイ 3
-u 3	エンベロープ (トレイ 1)
-u 4	手動封筒給紙機構
-u 6	多目的トレイ 1
-u 7	自動選択

以下の用紙サイズ選択は、**qprt** コマンドの **-Q** フラグを指定してサポートされます。

項目	説明
-Q 1	レター
-Q 2	リーガル
-Q 3	レジャー
-Q 4	A3
-Q 5	A4
-Q 6	exec
-Q 7	B4(JIS)
-Q 8	B5(JIS)
-Q 9	A5
-Q 10	exec(JIS)
-Q 11	8.5 x 13
-Q 12	ステートメント

Hewlett-Packard LaserJet 5Si および 5Si Mopier プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

出力ビン:

基本の LaserJet 5Si および 5Si Mopier プリンターには、可能な出力先が 2 つあります。

- **上段出力ビン**。表面を下にして印刷します。
- **左側出力ビン**。表面を上にして逆順に印刷します。
- オプションの大容量出力 (HCO) デバイスが取り付けられている場合は、追加のトレイも使用可能です。基本オペレーティング・システムは、HP 5Si プリンターの場合は 8 つまでの HCO 出力ビンをサポートし、5Si Mopier の場合は 5 つまでの HCO 出力ビンと 1 つのステープラー・ビンをサポートします。

出力ビンには、**qprt** コマンドの (**-=**) フラグを使用してアクセスできます。次の表に、LaserJet 5Si の場合の可能な値および対応する出力先の出力ビンを示します。

LaserJet 5Si:

-= 値 (#)	**出力先の出力ビン**
0	プリンター上段/フェースダウン・ビン
1	HC0 フェースダウン・ビン 1
2	HC0 フェースダウン・ビン 2
3	HC0 フェースダウン・ビン 3
4	HC0 フェースダウン・ビン 4
5	HC0 フェースダウン・ビン 5
6	HC0 フェースダウン・ビン 6
7	HC0 フェースダウン・ビン 7
8	HC0 フェースダウン・ビン 8
50	HC0 フェースアップ
50	プリンター左側/フェースアップ・ビン (HC0 が取り付けられていない場合)

出力ビンには、**qprt** コマンドの (**-=**) フラグを使用してアクセスできます。次の表に、LaserJet 5Si Mopier の場合の可能な値および対応する出力先の出力ビンを示します。

LaserJet 5Si Mopier:

-= 値 (#)	**出力先の出力ビン**
0	プリンター上段/フェースダウン・ビン
1	HC0 フェースダウン・ビン 1
2	HC0 フェースダウン・ビン 2
3	HC0 フェースダウン・ビン 3
4	HC0 フェースダウン・ビン 4
5	HC0 フェースダウン・ビン 5
50	HC0 フェースアップ
50	プリンター左側/フェースアップ・ビン (HC0 が取り付けられていない場合)
51	ステープラー・ビン

印刷部数 (LaserJet 5Si Mopier):

LaserJet 5Si Mopier は、**-W** フラグによって内部での複数部印刷をサポートします。これは、スプーラーがサポートする **-N** フラグとは異なります。**-N** フラグを使用すると、コピーは基本オペレーティング・システム・マシン上で処理された後、プリンターに一度に 1 つずつ送信されます。一方、LaserJet 5Si Mopier の **-W** オプションは、プリンターに 1 部のみの印刷ジョブを送信し、その後プリンターによってコピーが作成されます。基本フォーマットは **-W #** です。

Hewlett-Packard LaserJet 8000 プリンターおよび 8500 カラー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

出力ビン:

基本の LaserJet 8500 カラー・プリンターおよび LaserJet 8000 プリンターには、可能な出力先が 2 つあります。

- **上段出力ビン**。表面を下にして印刷します。
- **左側出力ビン**。表面を上にして逆順に印刷します。

オプションの大容量出力 (HC0) デバイスが取り付けられている場合は、追加のトレイも使用可能です。

出力ビンには、**qprt** コマンドの **(-=)** フラグを使用してアクセスできます。次の表に、可能な値および対応する出力先の出力ビンを示します。

LaserJet 8500 カラー:

-= 値 (#)	**出力先の出力ビン**
0	プリンター上段/フェースダウン・ビン
1	HC0 フェースダウン・ビン 1
2	HC0 フェースダウン・ビン 2
3	HC0 フェースダウン・ビン 3
4	HC0 フェースダウン・ビン 4
5	HC0 フェースダウン・ビン 5
6	HC0 フェースダウン・ビン 6
7	HC0 フェースダウン・ビン 7
8	HC0 フェースダウン・ビン 8
50	HC0 フェースアップ
50	プリンター左側/フェースアップ・ビン (HC0 が取り付けられていない場合)

LaserJet 8000:

-= 値 (#)	**出力先の出力ビン**
0	プリンター上段/フェースダウン・ビン
1	HC0 フェースダウン・ビン 1
2	HC0 フェースダウン・ビン 2
3	HC0 フェースダウン・ビン 3
4	HC0 フェースダウン・ビン 4
5	HC0 フェースダウン・ビン 5
6	HC0 フェースダウン・ビン 6
7	HC0 フェースダウン・ビン 7
8	HC0 フェースダウン・ビン 8
50	HC0 フェースアップ
50	プリンター左側/フェースアップ・ビン (HC0 が取り付けられていない場合)
51	ステープラー・ビン

印刷部数:

LaserJet 8000 プリンターおよび 8500 カラー・プリンターは、複数部印刷を内部でサポートします。**-W** フラグを使用すると、プリンターには 1 部のみの印刷ジョブが送信され、その後プリンターによってコピーが作成されます。基本フォーマットは **-W #** です。

用紙サイズ:

印刷ジョブの用紙サイズを指定します。

印刷ジョブ	用紙サイズ
-Q 1	レター
-Q 2	リーガル
-Q 4	A4
-Q 5	エグゼクティブ
-Q 8	A3

Lexmark 4227 フォーム・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。 **-u 1** トラクター 1 **-u 2** トラクター 2 バナー・ページとトレーラー・ページは、印刷ジョブと同じ給紙機構を使用します。トラクターの切り替えは、プリンターが有人のときに行うことをお勧めします。
ピッチ、フォント、および品質	ピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名は **-s** フラグ、印刷品質は **-q** フラグを使用して選択できます。サポートされるデフォルト値には、次のものがあります。 **10** ピッチ **courier** フォント **品質** 1 (ドラフト) 有効なフォント値には、次のものがあります。 **フォント名** **-s** fast draft **-s** draft **-s** courier **-s** gothic 有効な品質値には、次のものがあります。 **品質 (-q フラグ)** **0** fast draft **1** draft **2** レター品質 有効なピッチ値は、10、12、17、および 20 です。 **注:** 1. draft または fast draft を選択すると、選択済みのフォントが指定変更されます。 2. **-e** フラグと強調印刷を使用すると、太字フォントがサポートされます。**-k** フラグとイタリック印刷を使用すると、イタリック・フォントがサポートされます。
ページ幅	**-w** フラグは、印刷可能ページの幅を文字数単位で制御します。デフォルトは **136** です。

Lexmark Optra レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	拡張 PCL (R) 5 エミュレーションと PostScript レベル 2 エミュレーションのどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。オプションの給紙機構はいくつかあります。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。 **-u 0** 手差し **-u 1** トレイ 1 **-u 2** トレイ 2 **-u 3** トレイ 3 **-u 4** フィーダーまたはフィーダー 1 **-u 5** フィーダー 2 デフォルトでは、バナー・ページとトレーラー・ページは上段トレイから給紙されます。デフォルトを変更するには、コロン・ファイル内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値 (s0 から s5) に変更します。 **lsvirprt** コマンドを使用してください。
用紙サイズ	**qprt** コマンドのフラグ **-O** および **-Q** のどちらか、または両方を使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。値 3 は用紙サイズを示し、4 は封筒サイズを示します。封筒は、手差し、フィーダー、またはフィーダー 2 の場合のみ有効です。**-O** のデフォルトは 3 (用紙) です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は Com 10 です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5 用紙** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7** その他の封筒
用紙タイプ	Optra プリンターは、**qprt** コマンドの **-y** パラメーター、またはコロン・ファイルの **_y** 属性を使用して、ラフ紙、普通紙 (デフォルト)、透明、ラベル、およびカード・ストックの用紙タイプをサポートします。 **-y 1** ラフ紙 **-y 2** 普通紙 (デフォルト) **-y 3** 透明 **-y 4** ラベル **-y 5** カード・ストック
印刷解像度	Optra Plus プリンターは、**qprt** コマンドの **-q** フラグを使用して、300、600、および 1200 dpi の印刷解像度をサポートします。デフォルトは 600 dpi です。 **-q** 300 **-q** 600 **-q** 1200

製品固有の情報

項目	説明
ピッチ	PCL 5 エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。 **フォント名** ピッチ **-s courier** **-p** (1 から 100) **-s courier-bold** **-p** (1 から 100) **-s courier-italic** **-p** (1 から 100) **-s courier-bold italic** **-p** (1 から 100) **-s gothic -p** (1 から 100) **-s gothic-bold** **-p** (1 から 100) **-sgothic-italic** **-p** (1 から 100) **-s lineprinter** **-p** 17 **注:** その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。PostScript キューの場合は、**-p** はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。また、lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。
両面印刷モード	**qprt** コマンドの **-Y** フラグを使用した、オプションの両面印刷機能がサポートされています。 **-Y 0** 片面印刷 **-Y 1** 両面印刷、長辺とじ **-Y 2** 両面印刷、短辺とじ

Lexmark Optra Plus レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	拡張 PCL (R) 5 エミュレーションと PostScript レベル 2 エミュレーションのどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。オプションの給紙機構はいくつかあります。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。 **-u 0** 手差し **-u 1** トレイ 1 **-u 2** トレイ 2 **-u 3** トレイ 3 **-u 4** フィーダーまたはフィーダー 1 **-u 5** フィーダー 2 デフォルトでは、バナー・ページとトレーラー・ページは上段トレイから給紙されます。デフォルトを変更するには、コロン・ファイル内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値 (**s0** から **S5**) に変更します。 **lsvirprt** コマンドを使用してください。

製品固有の情報

項目	説明
用紙サイズ	**qprt** コマンドのフラグ **-O** および **-Q** のどちらか、または両方を使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。値 3 は用紙サイズを示し、4 は封筒サイズを示します。封筒は、手差し、フィーダー、またはフィーダー 2 の場合のみ有効です。**-O** のデフォルトは 3 (用紙) です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は Com 10 です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5 用紙** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7** その他の封筒
用紙タイプ	Optra Plus プリンターは、**qprt** コマンドの **-y** パラメーター、またはコロン・ファイルの **_y** 属性を使用して、ラフ紙、普通紙 (デフォルト)、透明、ラベル、およびカード・ストックの用紙タイプをサポートします。 **-y 1** ラフ紙 **-y 2** 普通紙 (デフォルト) **-y 3** 透明 **-y 4** ラベル **-y 5** カード・ストック
印刷解像度	Optra Plus プリンターは、**qprt** コマンドの **-q** フラグを使用して、300、600、および 1200 dpi の印刷解像度をサポートします。デフォルトは 600 dpi です。 **-q** 300 **-q** 600 **-q** 1200

製品固有の情報

項目 **説明**

ピッチ

PCL 5 エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。

フォント名
ピッチ

-s courier
-p (1 から 100)

-s courier-bold
-p (1 から 100)

-scourier-italic
-p (1 から 100)

-s courier-bold italic
-p (1 から 100)

-s gothic -p (1 から 100)

-s gothic-bold
-p (1 から 100)

-sgothic-italic
-p (1 から 100)

-s lineprinter
-p 17

注: その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。PostScript キューの場合は、**-p** はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。また、lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。

両面印刷モード

qprt コマンドの **-Y** フラグを使用した、オプションの両面印刷機能がサポートされています。

-Y 0 片面印刷
-Y 1 両面印刷、長辺とじ
-Y 2 両面印刷、短辺とじ

丁合い

Optra Plus プリンターは、印刷ジョブの複数部印刷の丁合いを内部でサポートします。この機能は、**qprt** コマンドの **-W** フラグと **-S** フラグによって制御されます。

-S ! 丁合いオフ
-S + 丁合いオン
-S # 印刷部数

注: この機能は、**qprt** コマンドの **-N** フラグとは独立しています。**-N#** フラグを使用すると、プリンター・ジョブがプリンターに # 回送信されます。**-W#** は印刷ジョブを 1 回送信し、このジョブが # 部印刷されます。

区切りページ

Optra Plus プリンターは、内部で生成される区切りページをサポートします。この機能は、**qprt** コマンドの **-E** フラグによって制御されます。

-E 0 なし
-E コピー間
-E 2 ジョブ間
-E 3 ページ間

給紙機構のデフォルトはフィーダーです。デフォルトを変更するには、仮想プリンターの **uS** 属性を変更する必要があります。**uS** の有効な値は、手差しが有効な給紙機構でないことを除いては、給紙機構フラグ **-u** と同じです。

注: この機能は、**qprt** コマンドの **-B** フラグとは独立しています。

Lexmark Optra Color 1200 プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	PCL 5 エミュレーションと PostScript 言語のどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。 **-u 0** 手差し **-u 1** トレイ 1 **-u 2** トレイ 2 **-u 3** トレイ 3 **-u 4** 多目的フィーダー デフォルトでは、バナー・ページとトレーラー・ページはトレイ 1 から給紙されます。デフォルトを変更するには、コロン・ファイル内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値に変更します。有効な値は、**-u** フラグと同じです。変更するには、**chvirprt** コマンドを使用して、仮想プリンターのコロン・ファイルを編集します。
用紙サイズ	**qprt** コマンドのフラグ **-O** および **-Q** を使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。 **-O** の値 3 は用紙サイズを示し、4 は封筒サイズを示します。値 1 と 2 は、後方互換性のためにスキップされます。最初の 5 つの用紙サイズも、後方互換性のために番号が付けられています。給紙機構に無効値が選択された場合は、無視されます。 **注:** 封筒は、手差しと MP トレイを選択した場合のみ使用可能です。 **-O** のデフォルトは、**3** (用紙) です。**-Q** のデフォルトは、用紙サイズの場合は **1** (レター)、封筒サイズの場合は Monarch です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5** 10 (Com 10) **-Q 4 A4** DL **-Q 5 A5** C5 **-Q 6 B4** B5 封筒 **-Q 7 A3** その他の封筒 **-Q 8 11 X 17** - **注:** PCL 5 用のプリンター・ファイル (lexOptraC1200.pcl) のデフォルトの用紙サイズは、レターです。デフォルトのサイズを変更するには、**_u** (給紙機構) 属性に対して、このファイル内でそれぞれ **s0** から **s3** 属性の値を変更します。例えば、リーガルをトレイ 2 のデフォルト・サイズにするには、**s2** 属性の値を 2 に変更します。 **注:** PCL キューの場合、選択したサイズが選択した給紙機構に存在しなければ、要求したサイズが検索シーケンスを使用して検索されます。サイズが見つかった場合は、その給紙機構が使用されます。PostScript キューの場合は、選択したサイズが選択した給紙機構に存在しなければ、適切なサイズの用紙を給紙するようにプリンターからユーザーにプロンプトが出されます。このため、予期しない給紙機構が使用されたり、すぐには意味が分からないメッセージがオペレーター・パネルに出されたりすることがあります。適切な応答を判別するには、マニュアルを参照してください。

製品固有の情報

項目 **説明**

ピッチ

PCL エミュレーションの場合はピッチの選択がサポートされており、ピッチは **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。

フォント名

ピッチ

-s courier

-p (1 から 100)

-s courier-bold

-p (1 から 100)

-s courier-italic

-p (1 から 100)

-s courier-bold italic

-p (1 から 100)

-s gothic -p (1 から 100)

-s gothic-bold

-p (1 から 100)

-sgothic-italic

-p (1 から 100)

-s lineprinter

-p 17

注: その他のフォント・スタイル用に ASCII ファイルのフォーマットを設定する場合は、基本オペレーティング・システムの enscript ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。Postscript キューの場合は、-p はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。

lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。

丁合い

通常、必要な印刷部数の指定には **-N** コマンド・ライン・オプションが使用されます。この方式では、印刷ジョブ全体の指定された数のコピーが印刷システムに対して実行依頼されるか、キューに入れられます。Optra Color 1200 は丁合いを内部でサポートするので、丁合いと各ページの印刷部数を内部でサポートするオプションが追加されています。この機能は、プリンターに取り付けられているメモリーの容量、およびジョブのサイズによって制限されます。**-W#** オプションは、各ページの必要な印刷部数を決定します (**#** は印刷部数)。**-S [!/+]** オプションは、丁合いが必要かどうかを制御します。デフォルトは ! (必要ない) です。**-W** オプションと **-S** オプションを使用する主な利点は、プリンター・サブシステムの使用量を節約し、プリンターに **#** 部のコピーを送信せずにプリンター側で複数部印刷を処理できることです。**-S!** オプションと **-W#** を組み合わせて使用すると、各ページを **#** 部連続して印刷することもできます (必要な場合)。**-N** と **-W** を同時に使用できることに注意してください。この結果、**-N** 個の印刷ジョブが作成され、それぞれのジョブに各ページの **-W** 部のコピーが含まれます。

区切りページ

-E フラグは、区切りページを制御します。有効な値は **0**、**1**、**2**、および **3** で、これらはそれぞれ「**なし**」、「**コピー間**」、「**ジョブ間**」、および「**ページ間**」を表します。区切りページの給紙機構は、コロン・ファイル属性 **uS** によって設定され、デフォルトはトレイ 1 です。**uS** の有効な値は、手差しが区切りページの有効な給紙機構でないことを除いては、**uH** および **uT** と同じです。デフォルトを変更するには、仮想プリンターの **uS** 属性を、有効な値の 1 つに変更する必要があります (このプリンターの『給紙機構』の情報を参照してください)。

Lexmark Optra Color 40 プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	PCL 5 エミュレーションと PostScript 言語のどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。 **-u 0** 手差し **-u 1** トレイ 1
用紙サイズ	qprt コマンドのフラグ **-O** および **-Q** を使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。 **-O** の値 3 は用紙サイズを示し、4 は封筒サイズを示します。値 1 と 2 は、後方互換性のためにスキップされます。最初の 5 つの用紙サイズも、後方互換性のために番号が付けられています。給紙機構に無効値が選択された場合は、無視されます。 **注:** 封筒は、手差しおよびトレイ 1 から給紙できます。 **-O** のデフォルトは、**3** (用紙) です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は Monarch です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7 汎用** その他の封筒 **注:** PCL 5 用のプリンター・ファイル (`lexOptraC40.pcl`) のデフォルトの用紙サイズは、レターです。デフォルトのサイズを変更するには、**_u** (給紙機構) 属性に対して、このファイル内でそれぞれ **s0** から **s3** 属性の値を変更します。例えば、リーガルをトレイ 2 のデフォルト・サイズにするには、**s2** 属性の値を 2 に変更します。 **注:** PCL キューの場合、選択したサイズが選択した給紙機構に存在しなければ、要求したサイズが検索シーケンスを使用して検索されます。サイズが見つかった場合は、その給紙機構が使用されます。PostScript キューの場合は、選択したサイズが選択した給紙機構に存在しなければ、適切なサイズの用紙を給紙するようにプリンターからユーザーにプロンプトが出されます。このため、予期しない給紙機構が使用されたり、すぐには意味が分からないメッセージがオペレーター・パネルに出されたりすることがあります。適切な応答を判別するには、マニュアルを参照してください。

製品固有の情報

項目 **説明**

ピッチ

PCL エミュレーションの場合はピッチの選択がサポートされており、ピッチは **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。

フォント名
ピッチ

-s courier
-p (1 から 100)

-s courier-bold
-p (1 から 100)

-s courier-italic
-p (1 から 100)

-s courier-bold italic
-p (1 から 100)

-s gothic -p (1 から 100)

-s gothic-bold
-p (1 から 100)

-sgothic-italic
-p (1 から 100)

-s lineprinter
-p 17

注: その他のフォント・スタイル用に ASCII ファイルのフォーマットを設定する場合は、基本オペレーティング・システムの enscript ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。Postscript キューの場合は、**-p** はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。

lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。

丁合い

通常、必要な印刷部数の指定には **-N** コマンド・ライン・オプションが使用されます。この方式では、印刷ジョブ全体の複数のコピーが印刷システムに対して実行依頼されるか、キューに入れられます。Optra Color 40 は丁合いを内部でサポートするので、丁合いと各ページの印刷部数を内部でサポートするオプションが追加されています。この機能は、プリンターに取り付けられているメモリーの容量、およびジョブのサイズによって制限されます。**-W#** オプションは、各ページの必要な印刷部数を決定します (**#** は印刷部数)。**-S [!/+]** オプションは、丁合いが必要かどうかを制御します。デフォルトは ! (必要ない) です。**-W** オプションと **-S** オプションを使用する主な利点は、プリンター・サブシステムの使用量を節約し、プリンターに **#** 部のコピーを送信せずにプリンター側で複数部印刷を処理できることです。**-S!** オプションと **-W#** を組み合わせて使用すると、各ページを **#** 部連続して印刷することもできます (必要な場合)。**-N** と **-W** を同時に使用できることに注意してください。この結果、**-N** 個の印刷ジョブが作成され、それぞれのジョブに各ページの **-W** 部のコピーが含まれます。

区切りページ

-E フラグは、区切りページを制御します。有効な値は **0**、**1**、**2**、および **3** で、これらはそれぞれ「**なし**」、「**コピー間**」、「**ジョブ間**」、および「**ページ間**」を表します。区切りページの給紙機構は、コロン・ファイル属性 **uS** によって設定され、デフォルトはトレイ 1 です。**uS** の有効な値は、手差しが区切りページの有効な給紙機構でないことを除いては、**uH** および **uT** と同じです。デフォルトを変更するには、仮想プリンターの **uS** 属性を、有効な値の 1 つに変更する必要があります (前述の『給紙機構』を参照してください)。

Lexmark Optra Color 45 プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	PCL 5 エミュレーションと PostScript 言語のどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。 **-u 0** 手差し **-u 1** トレイ 1
用紙サイズ	qprt コマンドのフラグ **-O** および **-Q** を使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。 **-O** の値 3 は用紙サイズを示し、4 は封筒サイズを示します。値 1 と 2 は、後方互換性のためにスキップされます。最初の 5 つの用紙サイズも、後方互換性のために番号が付けられています。給紙機構に無効値が選択された場合は、無視されます。 **注:** 封筒は、手差しおよびトレイ 1 から給紙できます。 **-O** のデフォルトは、**3** (用紙) です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は Monarch です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5** 10 (Com 10) **-Q 4 A4** DL **-Q 5 A5** C5 **-Q 6 エグゼクティブ** B5 封筒 **-Q 7 A3** その他の封筒 **-Q 8 11 X 17** - **-Q 9 汎用** - **注:** PCL 5 用のプリンター・ファイル (lexOptraC45.pcl) のデフォルトの用紙サイズは、レターです。デフォルトのサイズを変更するには、**_u** (給紙機構) 属性に対して、このファイル内でそれぞれ **s0** から **s3** 属性の値を変更します。例えば、リーガルをトレイ 2 のデフォルト・サイズにするには、**s2** 属性の値を 2 に変更します。 **注:** PCL キューの場合、選択したサイズが選択した給紙機構に存在しなければ、要求したサイズが検索シーケンスを使用して検索されます。サイズが見つかった場合は、その給紙機構が使用されます。PostScript キューの場合は、選択したサイズが選択した給紙機構に存在しなければ、適切なサイズの用紙を給紙するようにプリンターからユーザーにプロンプトが出されます。このため、予期しない給紙機構が使用されたり、すぐには意味が分からないメッセージがオペレーター・パネルに出されたりすることがあります。適切な応答を判別するには、マニュアルを参照してください。

製品固有の情報

項目	説明
ピッチ	PCL エミュレーションの場合はピッチの選択がサポートされており、ピッチは **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。 **フォント名** ピッチ **-s courier** **-p** (1 から 100) **-s courier-bold** **-p** (1 から 100) **-s courier-italic** **-p** (1 から 100) **-s courier-bold italic** **-p** (1 から 100) **-s gothic -p** (1 から 100) **-s gothic-bold** **-p** (1 から 100) **-sgothic-italic** **-p** (1 から 100) **-s lineprinter** **-p** 17 **注:** その他のフォント・スタイル用に ASCII ファイルのフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。Postscript キューの場合は、-p はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。 lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。
丁合い	通常、必要な印刷部数の指定には **-N** コマンド・ライン・オプションが使用されます。この方式では、印刷ジョブ全体の複数のコピーが印刷システムに対して実行依頼されるか、キューに入れられます。Optra Color 45 は丁合いを内部でサポートするので、丁合いと各ページの印刷部数を内部でサポートするオプションが追加されています。この機能は、プリンターに取り付けられているメモリーの容量、およびジョブのサイズによって制限されます。**-W#** オプションは、各ページの必要な印刷部数を決定します (**#** は印刷部数)。**-S [!/+]** オプションは、丁合いが必要かどうかを制御します。デフォルトは ! (必要ない) です。**-W** オプションと **-S** オプションを使用する主な利点は、プリンター・サブシステムの使用量を節約し、プリンターに # 部のコピーを送信せずにプリンター側で複数部印刷を処理できることです。**-S!** オプションと **-W#** を組み合わせて使用すると、各ページを # 部連続して印刷することもできます (必要な場合)。**-N** と **-W** を同時に使用できることに注意してください。この結果、**-N** 個の印刷ジョブが作成され、それぞれのジョブに各ページの **-W** 部のコピーが含まれます。
区切りページ	**-E** フラグは、区切りページを制御します。有効な値は **0**、**1**、**2**、および **3** で、これらはそれぞれ「**なし**」、「**コピー間**」、「**ジョブ間**」、および「**ページ間**」を表します。区切りページの給紙機構は、コロン・ファイル属性 **uS** によって設定され、デフォルトはトレイ 1 です。**uS** の有効な値は、手差しが区切りページの有効な給紙機構でないことを除いては、**uH** および **uT** と同じです。デフォルトを変更するには、仮想プリンターの **uS** 属性を、有効な値の 1 つに変更する必要があります (前述の『給紙機構』を参照してください)。

Lexmark Optra K 1220 プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	「拡張 PCL (R) 5e エミュレーション」と「PostScript (tm) レベル 2 エミュレーション」のどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。オプションの給紙機構はいくつかあります (取り付けられているものを判別するには、マニュアルを参照してください)。これらの番号は、どの給紙機構が存在するかに関係なく適用されます。存在しないものを選択すると、ユーザーズ・マニュアルに記載されているとおりのデフォルトが使用されます。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。 **-u 0** 手差し **-u 1** トレイ 1 **-u 2** トレイ 2 **-u 3** 多目的トレイ デフォルトでは、バナー・ページとトレーラー・ページはトレイ 1 から給紙されます。デフォルトを変更するには、仮想プリンター内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値に変更します。これは、**chvirprt** コマンドを使用して行います。有効な値は、**-u** フラグと同じです。
用紙サイズ	**qprt** コマンドのフラグ **-O** および **-Q** のどちらか、または両方を使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。値 3 は用紙サイズを示し、4 は封筒サイズを示します。値 1 と 2 は、後方互換性のためにスキップされます。封筒は、手差し、封筒フィーダー、または多目的トレイの場合のみ有効です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は 3 (Com 10) です。デフォルトを変更するには、それぞれの有効な給紙機構ごとに、**s0** から **s7** 属性をそれぞれ変更します。手差しと多目的トレイは用紙と封筒を両方ともサポートするので、用紙のデフォルトは **s0** と **s7** の「else」部分 (%e1) で、封筒のデフォルトは「then」部分 (%t3) です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5 (JIS B5)** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7 カスタム (汎用)** その他の封筒 **注:** PCL キューの場合、選択したサイズが選択した給紙機構に存在しなければ、要求したサイズが検索シーケンスを使用して検索されます。サイズが見つかった場合は、その給紙機構が使用されます。PostScript キューの場合は、選択したサイズが選択した給紙機構に存在しなければ、適切なサイズの用紙を給紙するようにプリンターからユーザーにプロンプトが出されます。このため、予期しない給紙機構が使用されたり、すぐには意味が分からないメッセージがオペレーター・パネルに出されたりすることがあります。適切な応答を判別するには、マニュアルを参照してください。

製品固有の情報

項目 **説明**

ピッチ PCL エミュレーションの場合はピッチの選択がサポートされており、ピッチは **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。

フォント名
ピッチ

-s courier
-p (1 から 100)

-s courier-bold
-p (1 から 100)

-s courier-italic
-p (1 から 100)

-s courier-bold italic
-p (1 から 100)

-s gothic -p (1 から 100)

-s gothic-bold
-p (1 から 100)

-sgothic-italic
-p (1 から 100)

-s lineprinter
-p 17

注: その他のフォント・スタイル用に ASCII ファイルのフォーマットを設定する場合は、基本オペレーティング・システムの enscript ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da** フラグを使用します。また、lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。

丁合い 通常、必要な印刷部数の指定には **-N** コマンド・ライン・オプションが使用されます。この方式では、印刷ジョブ全体の指定された数のコピーが印刷システムに対して実行依頼されるか、キューに入れられます。Optra K 1220 は丁合いを内部でサポートするので、丁合いと各ページの印刷部数を内部でサポートするオプションが追加されています。この機能は、プリンターに取り付けられているメモリーの容量、およびジョブのサイズによって制限されます。**-W#** オプションは、各ページの必要な印刷部数を決定します (# は印刷部数)。**-S [!/+]** オプションは、丁合いが必要かどうかを制御します。デフォルトは ! (必要ない) です。**-W** オプションと **-S** オプションを使用する主な利点は、プリンター・サブシステムの使用量を節約し、プリンターに # 部のコピーを送信せずにプリンター側で複数部印刷を処理できることです。**-S!** オプションと **-W#** を組み合わせて使用すると、各ページを # 部連続して印刷することもできます (必要な場合)。**-N** と **-W** を同時に使用できることに注意してください。この結果、**-N** 個の印刷ジョブが作成され、それぞれのジョブに各ページの **-W** 部のコピーが含まれます。

区切りページ **-E** フラグは、区切りページを制御します。有効な値は **0**、**1**、**2**、および **3** で、これらはそれぞれ「**なし**」、「**コピー間**」、「**ジョブ間**」、および「**ページ間**」を表します。区切りページの給紙機構は、コロン・ファイル属性 **uS** によって設定され、デフォルトはトレイ 1 です。**uS** の有効な値は、手差しが区切りページの有効な給紙機構でないことを除いては、**uH** および **uT** と同じです。デフォルトを変更するには、仮想プリンターの **uS** 属性を、有効な値の 1 つに変更する必要があります (前述の『給紙機構』を参照してください)。

Lexmark Optra C カラー・レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
PCL 5 エミュレーション・モードでのカラー・ファイルの印刷	カラー・ファイル、または事前フォーマット設定済みの印刷ジョブを PCL 言語で印刷するには、**qprt** コマンドの **-dp** フラグを使用します。このフラグは、基本オペレーティング・システムのプリンター・バックエンドをパススルー・モードに設定します。アプリケーションから PCL エミュレーションによる印刷を行う際には、常に使用する必要があります。 コロン・ファイル内の _d 属性を変更することによって、印刷キューのデフォルトをパススルーに変更できます。**lsvirprt** コマンドについては、「*Commands Reference, Volume 3*」を参照してください。
給紙機構	PCL 5 エミュレーションと PostScript 言語のどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。 **PCL** PostScript **-u 1 上段トレイ** **-u** 1 上段トレイ **-u 2 下段トレイ** **-u** 2 下段トレイ **-u 3 フィーダー** **-u** 3 フィーダー
用紙サイズ	PCL 5 エミュレーションの場合は、**qprt** コマンドの **-Q** フラグを使用した用紙サイズの選択がサポートされています。 **用紙サイズ** サイズ **-Q 1** レター (デフォルト) **-Q 2** リーガル **-Q 3** B5 **-Q 4** A4 デフォルトのサイズを変更するには、このファイル内で **s1** から **s3** 属性の値を変更します。例えば、すべての給紙機構に対して A4 をデフォルト・サイズにするには、**s1**、**s2**、および **s3** を 4 に変更します。これにより、上段トレイ、下段トレイ、およびフィーダー・トレイのサイズがそれぞれ変更されます。

製品固有の情報

項目	説明
ピッチ	PCL 5 エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。 **フォント名** ピッチ **-s courier** **-p** (1 から 100) **-s courier-bold** **-p** (1 から 100) **-s courier-italic** **-p** (1 から 100) **-s courier-bold italic** **-p** (1 から 100) **-s gothic -p** (1 から 100) **-s gothic-bold** **-p** (1 から 100) **-s gothic-italic** **-p** (1 から 100) **-s lineprinter** **-p** 17 **注:** その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。PostScript キューの場合は、**-p** はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。 また、lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。
丁合い	Optra C プリンターは、印刷ジョブの複数部印刷の丁合いを内部でサポートします。この機能は、**qprt** コマンドの **-W** フラグと **-S** フラグによって制御されます。 **-S !** 丁合いオフ **-S+** 丁合いオン **-S+** 印刷部数 **注:** この機能は、**qprt** コマンドの **-N** フラグとは独立しています。**-N#** フラグを使用すると、印刷ジョブがプリンターに # 回送信されます。**-W#** フラグは印刷ジョブを 1 回送信し、このジョブが # 部印刷されます。
区切りページ	Optra C プリンターは、内部で生成される区切りページをサポートします。この機能は、**qprt** コマンドの **-E** フラグによって制御されます。 **-E 0** なし **-E 1** コピー間 **-E 2** ジョブ間 **-E 3** ページ間 給紙機構のデフォルトはフィーダーです。デフォルトを変更するには、仮想プリンターの **uS** 属性を変更する必要があります。**uS** の有効な値は、給紙機構フラグと同じです。 **注:** この機能は、**qprt** コマンドの **-B** フラグとは独立しています。

Lexmark Optra E レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	PCL 5 エミュレーションと PostScript 言語のどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。 **PCL** **-u 1** 手差し **-u 2** 上段トレイ **-u 3** 下段トレイ デフォルトでは、バナー・ページとトレーラー・ページは上段トレイから給紙されます。デフォルトを変更するには、コロン・ファイル内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値 (**s1** から **s3**) に変更します。 **lsvirprt** コマンドを使用してください。
用紙サイズ	PCL 5 エミュレーションの場合は、**qprt** コマンドの **-Q** フラグと **-O** フラグを使用した用紙サイズの選択がサポートされています。**-O** フラグの値 3 は用紙を指示し、値 4 は封筒を指示します。トレイ 2 では封筒は無効です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5 用紙** 10 (Com 10) **-Q 4 A4** DL **-Q 5** エグゼクティブ C5 **-Q 6 A5** B5 封筒 **-Q 7** その他の封筒 **注:** PCL 5 用のプリンター・ファイル (`optra_e.pcl`) のデフォルトの用紙サイズは、レターです。デフォルトのサイズを変更するには、このファイル内で **s1** から **s3** 属性の値を変更します。例えば、すべての給紙機構に対して A4 をデフォルト・サイズにするには、**s1**、**s2**、および **s3** を 4 に変更します。これにより、上段トレイ、下段トレイ、およびフィーダー・トレイのサイズがそれぞれ変更されます。
用紙タイプ	Optra E プリンターは、**qprt** コマンドの **-y** パラメーター、またはコロン・ファイルの **_y** 属性を使用して、ラフ紙、普通紙 (デフォルト)、透明、ラベル、およびカード・ストックの用紙タイプをサポートします。 **-y 1** ラフ紙 **-y 2** 普通紙 (デフォルト) **-y 3** 透明 **-y 4** ラベル **-y 5** カード・ストック **注:** これらの値は用紙のみに適用され、封筒には適用されません。トレイ 2 でサポートされる値は、ラフ紙と普通紙のみです。
印刷解像度	Optra E プリンターは、**qprt** コマンドの **-q** フラグを使用して、300 および 600 dpi の印刷解像度をサポートします。デフォルトは 300 dpi です。 **-q** 300 **-q** 600

製品固有の情報

項目 **説明**

ピッチ PCL 5 エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。

フォント名
ピッチ

-s courier
-p (1 から 100)

-s courier-bold
-p (1 から 100)

-s courier-italic
-p (1 から 100)

-s courier-bold italic
-p (1 から 100)

-s gothic -p (1 から 100)

-s gothic-bold
-p (1 から 100)

-s gothic-italic
-p (1 から 100)

-s lineprinter
-p 17

注: その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。PostScript キューの場合は、**-p** はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。

また、lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。

各ページの印刷部数: **-W** フラグを使用すると、プリンター自体が作成する各ページの印刷部数を制御できます。例えば、**-qprt** コマンドの **-W** フラグを使用して 3 ページのジョブを実行依頼すると、ページ 1 が 2 枚、続いてページ 2 が 2 枚、続いてページ 3 が 2 枚印刷されます。デフォルト値は 1 で、最大値は 999 です。

Lexmark Optra N レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目 **説明**

給紙機構 拡張 PCL (R) 5 エミュレーションと PostScript レベル 2 エミュレーションのどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。オプションの給紙機構はいくつかあります (取り付けられているものを判別するには、ご使用のプリンターの資料を参照してください)。オプションの給紙機構は、どの給紙機構が取り付けられているかに関係なく適用されます。存在しないものを選択すると、デフォルトが使用されます。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。

-u 0 手差し
-u 1 トレイ 1
-u 2 トレイ 2
-u 3 トレイ 3
-u 4 封筒フィーダー
-u 5 多目的トレイ

製品固有の情報

項目	説明
用紙サイズ	**qprt** コマンドの **-O** フラグと **-Q** フラグを使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。 **-O** の値 3 は用紙サイズを示し、4 は封筒サイズを示します。値 1 と 2 は、後方互換性のためにスキップされます。最初の 5 つの用紙サイズも、後方互換性のために番号が付けられています。給紙機構に無効値が選択された場合は、無視されます。 **-O** のデフォルトは、3 (用紙) です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は Monarch です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5 用紙** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7 B4** その他の封筒 (MPT のみ) **-Q 8 A3** **-Q 9 レジャー (11x17)** **-Q 10 カスタム (11.69x17.69)** デフォルトを変更するには、それぞれの有効な入力値ごとに、**s0** から **s5** の属性を変更します。手差し、封筒フィーダー、および多目的トレイは用紙と封筒を両方ともサポートするので、デフォルトを変更するには **s0**、**s4**、または **s5** を編集します。これら 3 種類の機構に対する用紙のデフォルトは「else」部分 (%e1) で、封筒のデフォルトは「then」部分 (%t3) です。 **注:** 1. 封筒は、手差し、封筒フィーダー、または多目的トレイの場合のみ有効です。 2. トレイ 1、2、および 3 は、用紙サイズのみをサポートします。 3. 多目的トレイ (MPT) は、用紙と封筒を両方ともサポートします。 4. トレイ 1 は、サイズ **-Q** 1、2、4、および 7 (レター、リーガル、A4、および B4) をサポートします。 5. トレイ 2 および 3 は、サイズ **-Q** 1、2、4、7、8、9 (レター、リーガル、A4、B4、A3、レジャー) をサポートします。 6. 多目的トレイは、すべてのサイズの用紙と封筒をサポートします。 7. その他の封筒サイズは、多目的トレイによってのみサポートされます。 8. プリンターとコロン・ファイルのデフォルトの用紙サイズは、米国の場合はレター、ヨーロッパの場合は A4 で、デフォルトの封筒サイズは、米国の場合は COM10、ヨーロッパの場合は DL です。 9. 給紙機構に無効値が選択された場合は、エラーが報告されます。 10. 選択したサイズが選択した給紙機構に存在しなければ (誤ったサイズ、または空)、要求したサイズが検索シーケンスを使用して検索されます。詳しくは、ご使用のプリンターの資料を参照してください。

製品固有の情報

項目	説明
ピッチ	PCL 5 エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。 **フォント名** ピッチ **-s courier** **-p** (1 から 100) **-s courier-bold** **-p** (1 から 100) **-s courier-italic** **-p** (1 から 100) **-s courier-bold italic** **-p** (1 から 100) **-s gothic -p** (1 から 100) **-s gothic-bold** **-p** (1 から 100) **-s gothic-italic** **-p** (1 から 100) **-s lineprinter** **-p** 17 **注:** その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの enscript ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。PostScript キューの場合は、**-p** はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。また、lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。
両面印刷モード	**qprt** コマンドの **-Y** フラグを使用した、オプションの両面印刷機能がサポートされています。 **-Y 0** 片面印刷 **-Y 1** 両面印刷、長辺とじ **-Y 2** 両面印刷、短辺とじ
丁合いと印刷部数	Optra N プリンターは、印刷ジョブの複数部印刷の丁合いを内部でサポートします。この機能は、**qprt** コマンドの **-W** フラグと **-S** フラグによって制御されます。 **-S !** 丁合いオフ **-S +** 丁合いオン **-W #** 印刷部数 **注:** 1. この機能は、**qprt** コマンドの **-N** フラグとは独立しています。**-N#** フラグを使用すると、プリンターはジョブをプリンターに複数回送信します。**-W#** は印刷ジョブを 1 回送信し、このジョブが # 部印刷されます。 2. この機能は、プリンターに取り付けられているメモリーの容量、および印刷ジョブのサイズによって制限されます。

製品固有の情報

項目	説明
区切りページ	Optra N プリンターは、内部で生成される区切りページをサポートします。この機能は、**qprt** コマンドの **-E** フラグによって制御されます。 **-E 0** なし **-E 1** コピー間 **-E 2** ジョブ間 **-E 3** ページ間 給紙機構のデフォルトはトレイ 1 です。デフォルトを変更するには、仮想プリンターの **uS** 属性を変更する必要があります。uS の有効な値は、次のとおりです。 **uS 1** トレイ 1 **uS 2** トレイ 2 **uS 3** トレイ 3 **uS 4** 封筒フィーダー **uS 5** 多目的トレイ **注:** この機能は、**qprt** コマンドの **-B** フラグとは独立しています。
出力ビン	等号 **(=)** は、出力先を指定するためのコマンド・ライン・オプションです。有効な値は、次のとおりです。 **0** プリンター上段ビン **1** フィニッシャー・ビン 1 **2** フィニッシャー・ビン 2 **3** フィニッシャー・ビン 3 **50** プリンター・サイド・ビン プリンター上段ビンまたは 0 が、出力ビンのデフォルト値です。 **注:** プリンター・サイド・ビンが選択された場合に、フィニッシャー・オプションが取り付けられていると、出力はアクティブ・ビンに送られます。
フェースアップまたはフェースダウン	**-U** オプションは、用紙がフィニッシャーに対してフェースアップまたはフェースダウンのどちらで出力されるかを制御します。 **注:** プリンター上段ビンは常にフェースアップです。値 **+** または true はフェースダウンを指示し、これがデフォルトです。値 **!** または false はフェースアップを指示します。フェースアップが選択されると、ステープル **(-y)** とジョブ・オフセット **(-e)** は無視されます。
ステープル	**-y** オプションは、ステープルが必要かどうかを制御します。このフラグの値それぞれに対して、特定の用紙サイズのみがサポートされます。また、出力の数量と出力先についていくつかの規則があります。可能なすべての設定について詳しくは、ご使用のプリンターの資料を参照してください。有効な値は次のとおりです。 **0** ステープルなし (デフォルト) **1** 1 カ所ステープル (左上) **2** 2 カ所ステープル (左側)
ジョブ・オフセット	**-e** フラグは、フィニッシャー・ビン内で各ジョブの先頭ページにオフセットが必要かどうかを制御します。先頭ページに、フィニッシャーの前部方向に 1.7 インチのオフセットが付けられます。ステープルがオフでない場合、オフセット機能は無視されます。セパレーター・シートは、オフセットとは独立して選択できます。有効な値は次のとおりです。 **+** ジョブ・オフセットはオン **!** ジョブ・オフセットはオフ (デフォルト)

Lexmark Optra E310 レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
ページ回転	PCL 5 エミュレーションの場合は、**qprt** コマンドの **-z** フラグを使用したページ回転の選択がサポートされています。 **-z 0** 縦長 **-z 1** 横長
給紙機構	拡張 PCL 5 エミュレーションと PostScript レベル 2 エミュレーションのどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。 **-u 0** 手差し **-u 1** トレイ 1 デフォルトでは、バナー・ページとトレーラー・ページは上段トレイから給紙されます。デフォルトを変更するには、コロン・ファイル内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値に変更します。有効な値は、**-u** フラグと同じです。変更するには、**chvirprt** コマンドを使用して、仮想プリンターのコロン・ファイルを編集します。
用紙サイズ	**qprt** コマンドの **-O** フラグと **-Q** フラグを使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。 **-O** の値 3 は用紙サイズを示し、4 は封筒サイズを示します。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7** その他の封筒 (MPT のみ) デフォルトを変更するには、`lexOptraE310.pcl` コロン・ファイルの **s1**、**s3** 属性の値を変更します。用紙サイズのデフォルトは 1 またはレターで、封筒のデフォルトは 3 または Com 10 です。レター値は **s1** 属性と **s3** 属性の *else* 部分 (%e1)、封筒は *then* 部分 (%t3) です。
用紙タイプ	Optra E310 プリンターは、**qprt** コマンドの **-y** パラメーター、またはコロン・ファイルの **_y** 属性を使用して、ラフ紙、普通紙 (デフォルト)、透明、ラベル、およびカード・ストックの用紙タイプをサポートします。**-y** オプションの値は、前述のタイプに対応してそれぞれ 1 から 5 です。 **-y 1** ボンド紙 **-y 2** 普通紙 **-y 3** 透明 **-y 4** ラベル **-y 5** カード・ストック **注:** これらの値は、封筒には適用されません。

製品固有の情報

項目 **説明**

ピッチ

PCL エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。

フォント名
ピッチ

-s courier
-p (1 から 100)

-s courier-bold
-p (1 から 100)

-s courier-italic
-p (1 から 100)

-s courier-bolditalic
-p (1 から 100)

-s gothic **-p** (1 から 100)

-s gothic-bold
-p (1 から 100)

-s gothic-italic
-p (1 から 100)

-s lineprinter
-p 17

注: その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。PostScript キューの場合は、**-p** はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。

lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。

各ページの印刷部数:

qprt コマンドの **-W** フラグは、各ページの印刷部数を制御します。デフォルトは 1 部で、最大値は 999 です。

-w # 印刷部数

例: qprtコマンドに **-W2** を指定して 3 ページのジョブを実行依頼すると、ページ 1 が 2 枚、続いてページ 2 が 2 枚、さらに続いてページ 3 が 2 枚、この順序で印刷されます。

Lexmark Optra M410 レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目 **説明**

ページ回転

PCL 5e エミュレーションの場合は、**qprt** コマンドの **-z** フラグを使用したページ回転の選択がサポートされています。

-z 0 縦長

-z 1 横長

製品固有の情報

項目	説明
給紙機構	拡張 PCL (R) 5e エミュレーションと PostScript レベル 2 エミュレーションのどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。オプションの給紙機構はいくつかあります (取り付けられているものを判別するには、マニュアルを参照してください)。これらの番号は、どの給紙機構が存在するかに関係なく適用されます。存在しないものを選択すると、ユーザーズ・マニュアルに記載されているとおりのデフォルトが使用されます。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。 **-u 0** 手差し **-u 1** トレイ 1 **-u 2** トレイ 2 **-u 3** 多目的トレイ デフォルトでは、バナー・ページとトレーラー・ページはトレイ 1 から給紙されます。デフォルトを変更するには、仮想プリンター内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値に変更します。有効な値は、**-u** フラグと同じです。この変更を行うには、**chvirprt** コマンドを使用して編集します。
用紙サイズ	**qprt** コマンドのフラグ **-O** および **-Q** のどちらか、または両方を使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。 **-O** の値 3 は用紙サイズを示し、4 は封筒サイズを示します。値 1 と 2 は、後方互換性のためにスキップされます。封筒は、手差し、封筒フィーダー、または多目的トレイの場合のみ有効です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は 3 (Com 10) です。デフォルトを変更するには、それぞれの有効な給紙機構ごとに、**s0** から **s7** 属性をそれぞれ変更します。手差しと多目的トレイは用紙と封筒を両方ともサポートするので、用紙のデフォルトは **s0** と **s7** の *else* 部分 (%e1) で、封筒のデフォルトは *then* 部分 (%t3) です。 **用紙サイズ (-O 3)** 封筒サイズ (-O 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5 (JIS B5)** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7 カスタム (汎用)** その他の封筒 **注:** PCL キューの場合、選択したサイズが選択した給紙機構に存在しなければ、要求したサイズが検索シーケンスを使用して検索されます。サイズが見つかった場合は、その給紙機構が使用されます。 PostScript キューの場合は、選択したサイズが選択した給紙機構に存在しなければ、適切なサイズの用紙を給紙するようにプリンターからユーザーにプロンプトが出されます。このため、予期しない給紙機構が使用されたり、すぐには意味が分からないメッセージがオペレーター・パネルに出されたりすることがあります。適切な応答を判別するには、マニュアルを参照してください。

製品固有の情報

項目	説明
ピッチ	PCL 5 エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。 **フォント名** ピッチ **-s courier** **-p** (1 から 100) **-s courier-bold** **-p** (1 から 100) **-s courier-italic** **-p** (1 から 100) **-s courier-bolditalic** **-p** (1 から 100) **-s gothic -p** (1 から 100) **-s gothic-bold** **-p** (1 から 100) **-s gothic-italic** **-p** (1 から 100) **-s lineprinter** **-p** 17 **注:** その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da** フラグを使用します。lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。
丁合い	通常、必要な印刷部数の指定には **-N** コマンド・ライン・オプションが使用されます。この方式では、印刷ジョブ全体の指定された数のコピーが印刷システムに対して実行依頼されるか、キューに入れられます。Optra 410 は丁合いを内部でサポートするので、丁合いと各ページの印刷部数を内部でサポートするオプションが追加されています。この機能は、プリンターに取り付けられているメモリーの容量、およびジョブのサイズによって制限されます。**-W #** オプションは、各ページの必要な印刷部数を決定します (**#** は印刷部数)。**-S [!/+]** オプションは、丁合いが必要かどうかを制御します。デフォルトは **!** (必要ない) です。**-W** オプションと **-S** オプションを使用する主な利点は、プリンター・サブシステムの使用量を節約し、プリンターに # 部のコピーを送信せずにプリンター側で複数部印刷を処理できることです。**-S!** オプションと **-W #** を組み合わせて使用すると、各ページを # 部連続して印刷することもできます (必要な場合)。**-N** と **-W** を同時に使用できることに注意してください。この結果、**-N** 個の印刷ジョブが作成され、それぞれのジョブに各ページの **-W** 部のコピーが含まれます。
区切りページ	**-E** フラグは、区切りページを制御します。有効な値は **0**、**1**、**2**、および **3** で、これらはそれぞれ「**なし**」、「**コピー間**」、「**ジョブ間**」、および「**ページ間**」を表します。区切りページの給紙機構は、デフォルトではトレイ 1 で、**uS** 属性によって指定されます。**uS** の有効な値は、手差しフィーダーがサポートされないことを除いては、ヘッダー・ページとトレーラー・ページ (それぞれ **uH** と **uT**) の場合と同じです。デフォルトを変更するには、仮想プリンターの **uS** 属性を、有効な値の 1 つに変更する必要があります (前述の **chvirprt** コマンドを参照してください)。

Lexmark Optra Se レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
ページ回転	PCL 5e エミュレーションの場合は、**qprt** コマンドの **-z** フラグを使用したページ回転の選択がサポートされています。 **-z 0** 縦長 **-z 1** 横長
給紙機構	拡張 PCL (R) 5e エミュレーションと PostScript レベル 2 エミュレーションのどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。オプションの給紙機構はいくつかあります (取り付けられているものを判別するには、マニュアルを参照してください)。これらの番号は、どの給紙機構が存在するかに関係なく適用されます。存在しないものを選択すると、ユーザーズ・マニュアルに記載されているとおりのデフォルトが使用されます。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。 **-u 0** 手差し **-u 1** トレイ 1 **-u 2** トレイ 2 **-u 3** トレイ 3 **-u 4** トレイ 4 **-u 5** トレイ 5 **-u 6** 封筒フィーダー **-u 7** 多目的トレイ デフォルトでは、バナー・ページとトレーラー・ページはトレイ 1 から給紙されます。デフォルトを変更するには、仮想プリンター内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値に変更します。有効な値は、**-u** フラグと同じです。これは、**chvirprt** コマンドを使用して行います。
用紙サイズ	**qprt** コマンドの **-O** フラグと **-Q** フラグを使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。 **-O** の値 3 は用紙サイズを示し、4 は封筒サイズを示します。値 1 と 2 は、後方互換性のためにスキップされます。封筒は、手差し、封筒フィーダー、または多目的トレイの場合のみ有効です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は 3 (Com 10) です。デフォルトを変更するには、それぞれの有効な給紙機構ごとに、**s0** から **s7** 属性をそれぞれ変更します。手差しと多目的トレイは用紙と封筒を両方ともサポートするので、用紙のデフォルトは **s0** と **s7** の *else* 部分 (%e1) で、封筒のデフォルトは *then* 部分 (%t3) です。 **用紙サイズ (-O 3)** 封筒サイズ (-O 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5 (JIS B5)** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7 カスタム (汎用)** その他の封筒 **注:** PCL キューの場合、選択したサイズが選択した給紙機構に存在しなければ、要求したサイズが検索シーケンスを使用して検索されます。サイズが見つかった場合は、その給紙機構が使用されます。 PostScript キューの場合は、選択したサイズが選択した給紙機構に存在しなければ、適切なサイズの用紙を給紙するようにプリンターからユーザーにプロンプトが出されます。このため、予期しない給紙機構が使用されたり、すぐには意味が分からないメッセージがオペレーター・パネルに出されたりすることがあります。適切な応答を判別するには、マニュアルを参照してください。

製品固有の情報

項目 **説明**

用紙タイプ
Optra Se プリンターがサポートする用紙タイプは、普通紙 (デフォルト)、ボンド紙、透明、カード・ストック、ラベル、レターヘッド、事前印刷、色付き用紙、封筒 (封筒フィーダーのデフォルト)、およびカスタム・タイプ **x** (ここで **x** は 1 から 6) です。このコロン・ファイルは、これらの値の設定を行わず、その給紙機構に対するプリンターの設定をそのまま使用します。ユーザーは、指定された給紙機構に正しい用紙タイプが実際に装てんされていることを確認する必要があります。

ピッチ
PCL 5 エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。

フォント名
ピッチ

-s courier
-p (1 から 100)

-s courier-bold
-p (1 から 100)

-s courier-italic
-p (1 から 100)

-s courier-bolditalic
-p (1 から 100)

-s gothic **-p** (1 から 100)

-s gothic-bold
-p (1 から 100)

-s gothic-italic
-p (1 から 100)

-s lineprinter
-p 17

注: その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da** フラグを使用します。lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。

両面印刷モード
qprt のコマンド・ライン・オプション **-Y** がこれをサポートします。

0 片面印刷操作
1 両面印刷、長辺とじ
2 両面印刷、短辺とじ

丁合い
通常、必要な印刷部数の指定には **-N** コマンド・ライン・オプションが使用されます。この方式では、印刷ジョブ全体の指定された数のコピーが印刷システムに対して実行依頼されるか、キューに入れられます。Optra Se は丁合いを内部でサポートするので、丁合いと各ページの印刷部数を内部でサポートするオプションが追加されています。この機能は、プリンターに取り付けられているメモリーの容量、およびジョブのサイズによって制限されます。**-W#** オプションは、各ページの必要な印刷部数を決定します (**#** は印刷部数)。**-S [!/+]** オプションは、丁合いが必要かどうかを制御します。デフォルトは **!** (必要ない) です。**-W** オプションと **-S** オプションを使用する主な利点は、プリンター・サブシステムの使用量を節約し、プリンターに # 部のコピーを送信せずにプリンター側で複数部印刷を処理できることです。**-S!** オプションと **-W#** を組み合わせて使用すると、各ページを # 部連続して印刷することもできます (必要な場合)。**-N** と **-W** を同時に使用できることに注意してください。この結果、**-N** 個の印刷ジョブが作成され、それぞれのジョブに各ページの **-W** 部のコピーが含まれます。

区切りページ
-E フラグは、区切りページを制御します。有効な値は **0**、**1**、**2**、および **3** で、これらはそれぞれ「**なし**」、「**コピー間**」、「**ジョブ間**」、および「**ページ間**」を表します。区切りページの給紙機構は、デフォルトではトレイ 1 で、**uS** 属性によって指定されます。**uS** の有効な値は、手差しフィーダーがサポートされないことを除いては、ヘッダー・ページとトレーラー・ページ (それぞれ **uH** と **uT**) の場合と同じです。デフォルトを変更するには、仮想プリンターの **uS** 属性を、有効な値の 1 つに変更する必要があります (前述の **chvirprt** コマンドを参照してください)。

製品固有の情報

項目	説明
出力先	**=** (等号) は、出力先を指定するためのコマンド・ライン・オプションです。有効な値は、次のとおりです。 **0** 標準ビン **1** ビン 1 **2** ビン 2 **3** ビン 3 **50** アクティブ・ビン 出力先のデフォルト値は、標準ビン (**0**) です。アクティブ・ビンが選択されると、出力ビン容量センスの状態と、「**PAPER MENU (用紙メニュー)**」の下にある「**Configure Bins (ビンの構成)**」のオペレーター・パネル設定に基づいて、プリンターがビンを選択します。プリンターの応答について正確に調べるには、ご使用のプリンターのマニュアルを参照してください。

Lexmark Optra T レーザー・プリンター・ファミリー

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
ページ回転	PCL 5e エミュレーションの場合は、**qprt** コマンドの **-z** フラグを使用したページ回転の選択がサポートされています。 **-z 0** 縦長 **-z 1** 横長
給紙機構	拡張 PCL (R) 5e エミュレーションと PostScript レベル 2 エミュレーションのどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。オプションの給紙機構はいくつかあります (取り付けられているものを判別するには、マニュアルを参照してください)。これらの番号は、どの給紙機構が存在するかに関係なく適用されます。存在しないものを選択すると、ユーザーズ・マニュアルに記載されているとおりのデフォルトが使用されます。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。 **-u 0** 手差し **-u 1** トレイ 1 **-u 2** トレイ 2 **-u 3** トレイ 3 **-u 4** トレイ 4 **-u 5** トレイ 5 **-u 6** 封筒フィーダー **-u 7** 多目的トレイ デフォルトでは、バナー・ページとトレーラー・ページはトレイ 1 から給紙されます。デフォルトを変更するには、仮想プリンター内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値に変更します。有効な値は、**-u** フラグと同じです。これは、**chvirprt** コマンドを使用して行います。

製品固有の情報

項目	説明
用紙サイズ	**qprt** コマンドの **-O** フラグと **-Q** フラグを使用した用紙サイズの選択がサポートされています。**-O** フラグは、用紙と封筒の選択を制御します。 **-O** の値 3 は用紙サイズを示し、4 は封筒サイズを示します。値 1 と 2 は、後方互換性のためにスキップされます。封筒は、手差し、封筒フィーダー、または多目的トレイの場合のみ有効です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は 3 (Com 10) です。デフォルトを変更するには、それぞれの有効な給紙機構ごとに、**s0** から **s7** 属性をそれぞれ変更します。手差しと多目的トレイは用紙と封筒を両方ともサポートするので、用紙のデフォルトは **s0** と **s7** の *else* 部分 (%e1) で、封筒のデフォルトは *then* 部分 (%t3) です。 **用紙サイズ (-0 3)** 封筒サイズ (-0 4) **-Q 1 レター** 7 3/4 Monarch **-Q 2 リーガル** 9 (Com 9) **-Q 3 B5 (JIS B5)** 10 (Com 10) **-Q 4 A4** DL **-Q 5 エグゼクティブ** C5 **-Q 6 A5** B5 封筒 **-Q 7 カスタム (汎用)** その他の封筒 **注:** PCL キューの場合、選択したサイズが選択した給紙機構に存在しなければ、要求したサイズが検索シーケンスを使用して検索されます。サイズが見つかった場合は、その給紙機構が使用されます。 PostScript キューの場合は、選択したサイズが選択した給紙機構に存在しなければ、適切なサイズの用紙を給紙するようにプリンターからユーザーにプロンプトが出されます。このため、予期しない給紙機構が使用されたり、すぐには意味が分からないメッセージがオペレーター・パネルに出されたりすることがあります。適切な応答を判別するには、マニュアルを参照してください。
用紙タイプ	Optra T プリンターがサポートする用紙タイプは、普通紙 (デフォルト)、ボンド紙、透明、カード・ストック、ラベル、レターヘッド、事前印刷、色付き用紙、封筒 (封筒フィーダーのデフォルト)、およびカスタム・タイプ **x** (ここで x は 1 から 6) です。このコロン・ファイルは、これらの値の設定を行わず、その給紙機構に対するプリンターの設定をそのまま使用します。ユーザーは、指定された給紙機構に正しい用紙タイプが実際に装てんされていることを確認する必要があります。

製品固有の情報

項目	説明
ピッチ	PCL エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。 **フォント名** ピッチ **-s courier** **-p** (1 から 100) **-s courier-bold** **-p** (1 から 100) **-s courier-italic** **-p** (1 から 100) **-s courier-bolditalic** **-p** (1 から 100) **-s gothic -p** (1 から 100) **-s gothic-bold** **-p** (1 から 100) **-s gothic-italic** **-p** (1 から 100) **-s lineprinter** **-p** 17 **注:** その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da** フラグを使用します。lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。
両面印刷モード	**qprt** のコマンド・ライン・オプション **-Y** がこれをサポートします。 **0** 片面印刷操作 **1** 両面印刷、長辺とじ **2** 両面印刷、短辺とじ
丁合い	通常、必要な印刷部数の指定には **-N** コマンド・ライン・オプションが使用されます。この方式では、印刷ジョブ全体の指定された数のコピーが印刷システムに対して実行依頼されるか、キューに入れられます。Optra T は丁合いを内部でサポートするので、丁合いと各ページの印刷部数を内部でサポートするオプションが追加されています。この機能は、プリンターに取り付けられているメモリーの容量、およびジョブのサイズによって制限されます。**-W#** オプションは、各ページの必要な印刷部数を決定します (**#** は印刷部数)。**-S [!/+]** オプションは、丁合いが必要かどうかを制御します。デフォルトは **!** (必要ない) です。**-W** オプションと **-S** オプションを使用する主な利点は、プリンター・サブシステムの使用量を節約し、プリンターに # 部のコピーを送信せずにプリンター側で複数部印刷を処理できることです。**-S!** オプションと **-W#** を組み合わせて使用すると、各ページを # 部連続して印刷することもできます (必要な場合)。**-N** と **-W** を同時に使用できることに注意してください。この結果、**-N** 個の印刷ジョブが作成され、それぞれのジョブに各ページの **-W** 部のコピーが含まれます。
区切りページ	**-E** フラグは、区切りページを制御します。有効な値は **0**、**1**、**2**、および **3** で、これらはそれぞれ「**なし**」、「**コピー間**」、「**ジョブ間**」、および「**ページ間**」を表します。区切りページの給紙機構は、デフォルトではトレイ 1 で、**uS** 属性によって指定されます。**uS** の有効な値は、手差しフィーダーがサポートされないことを除いては、ヘッダー・ページとトレーラー・ページ (それぞれ **uH** と **uT**) の場合と同じです。デフォルトを変更するには、仮想プリンターの **uS** 属性を、有効な値の 1 つに変更する必要があります (前述の **chvirprt** コマンドを参照してください)。

製品固有の情報

項目	説明
出力先	= (等号) は、出力先を指定するためのコマンド・ライン・オプションです。有効な値は、次のとおりです。 **0** 標準ビン **1** ビン 1 **2** ビン 2 **3** ビン 3 **4** ビン 4 **5** ビン 5 **6** ビン 6 **7** ビン 7 **8** ビン 8 **9** ビン 9 **10** ビン 10 出力先のデフォルト値は、標準ビン (**0**) です。

Lexmark Optra W810 レーザー・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

製品固有の情報

項目	説明
ページ回転	PCL 5 エミュレーションの場合は、**qprt** コマンドの **-z** フラグを使用したページ回転の選択がサポートされています。 **-z 0** 縦長 **-z 1** 横長
給紙機構	拡張 PCL (R) 5 エミュレーションと PostScript レベル 2 エミュレーションのどちらの場合も、**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。オプションの給紙機構はいくつかあります (取り付けられているものを判別するには、マニュアルを参照してください)。これらの番号は、どの給紙機構が存在するかに関係なく適用されます。存在しないものを選択すると、ユーザーズ・マニュアルに記載されているとおりのデフォルトが使用されます。給紙機構番号は、PCL と PostScript のどちらの場合も同じです。 **-u 0** 手差し **-u 1** トレイ 1 **-u 2** トレイ 2 **-u 3** トレイ 3 **-u 4** トレイ 4 デフォルトでは、バナー・ページとトレーラー・ページは上段トレイから給紙されます。デフォルトを変更するには、コロン・ファイル内で **uH** 属性または **uT** 属性 (あるいはその両方) の値を目的の給紙機構の値に変更します。有効な値は、**-u** フラグと同じです。変更するには、**chvirprt** コマンドを使用して、仮想プリンターのコロン・ファイルを編集します。

製品固有の情報

項目	説明
用紙サイズ	**qprt** コマンドの **-Q** フラグを使用した用紙サイズの選択がサポートされています。最初の 5 つの用紙サイズも、後方互換性のために番号が付けられています。給紙機構に無効値が選択された場合は、無視されます。 **-O** のデフォルトは、3 (用紙) です。**-Q** のデフォルトは、用紙サイズの場合は 1 (レター)、封筒サイズの場合は Monarch です。 **用紙サイズ** **-Q 1** レター **-Q 2** リーガル **-Q 3** B5 用紙 **-Q 4** A4 **-Q 5** エグゼクティブ **-Q 6** A5 **-Q 7** B4 **-Q 8** A3 **-Q 9** レジャー (11x17) **-Q 10** 汎用 (11.69x17.69) デフォルトを変更するには、それぞれの有効な給紙機構ごとに、**s0 から s5** 属性をそれぞれ変更します。デフォルトの用紙サイズは、*else* 部分 (%e1) です。 **注:** 1. 手差しとトレイ 1 は、サイズ **-Q** 1、2、3、4、5、6、7、8、9、10 (レター、リーガル、B4、A4、エグゼクティブ、A5、B4、A3、11x17、汎用) をサポートします。 2. トレイ 2、3、および 4 は、サイズ -Q 1、2、4、7、8、9 (レター、リーガル、A4、B4、A3、11x17) をサポートします。 3. プリンター (およびこのコロン・ファイル) のデフォルトの用紙サイズは、米国の場合はレター、ヨーロッパの場合は A4 です。 4. 選択した給紙機構に無効な用紙サイズ値を使用すると、エラーが報告されます。 5. 選択したサイズが選択した給紙機構に存在しなければ (誤ったサイズ、または空)、要求したサイズが検索シーケンスを使用して検索されます。詳しくは、プリンターのマニュアルを参照してください。

製品固有の情報

項目 **説明**

ピッチ

PCL 5 エミュレーションの場合はピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名 (または書体) は **-s** フラグを使用して選択できます。1 から 100 字/インチ (cpi) の整数のピッチ値がサポートされています。縮刷機能フラグ **-K** はサポートされません。

フォント名
ピッチ

-s courier
-p (1 から 100)

-s courier-bold
-p (1 から 100)

-s courier-italic
-p (1 から 100)

-s courier-bolditalic
-p (1 から 100)

-s gothic **-p** (1 から 100)

-s gothic-bold
-p (1 から 100)

-s gothic-italic
-p (1 から 100)

-s lineprinter
-p 17

注: その他のフォント・スタイル用に ASCII のフォーマットを設定する場合は、基本オペレーティング・システムの **enscript** ユーティリティーを使用するか、または PostScript キューに対して **qprt** コマンドの **-da**、**-s**、および **-p** の各フラグを使用します。PostScript キューの場合は、**-p** はポイント・サイズを表し、フォントの有効なリストは `/usr/lib/ps/fontmap` にあります。有効なポイント・サイズは、1 から 1008 の任意の整数です。lineprinter フォント・スタイルに対してはピッチ 17 のみがサポートされています。

両面印刷モード

qprt コマンドの **-Y** フラグを使用した、オプションの両面印刷機能がサポートされています。デフォルトは 0 (片面印刷モード) です。

-Y 0 片面印刷
-Y 1 両面印刷、長辺とじ
-Y 2 両面印刷、短辺とじ

丁合いと印刷部数

Optra W810 プリンターは、印刷ジョブの複数部印刷の丁合いを内部でサポートします。この機能は、**qprt** コマンドの **-W** フラグと **-S** フラグによって制御されます。

-S! 丁合いオフ
-S+ 丁合いオン
-W# 印刷部数

注:

1. この機能は、**qprt** コマンドの **-N** フラグとは独立しています。**-N#** フラグを使用すると、印刷ジョブがプリンターに # 回送信されます。**-W#** フラグは印刷ジョブを 1 回送信し、このジョブが # 部印刷されます。
2. この機能は、プリンターに取り付けられているメモリーの容量、および印刷ジョブのサイズによって制限されます。

製品固有の情報

項目	説明
区切りページ	このプリンターは、内部で生成される区切りページをサポートします。この機能は、**qprt** コマンドの **-E** フラグによって制御されます。 **-E0** なし **-E1** コピー間 **-E2** ジョブ間 **-E3** ページ間 給紙機構のデフォルトはトレイ 1 です。デフォルトを変更するには、仮想プリンターの **uS** 属性を変更する必要があります。**uS** の有効な値は、次のとおりです。 **uS 1** トレイ 1 **uS 2** トレイ 2 **uS 3** トレイ 3 **uS 4** トレイ 4 **注:** この機能は、**qprt** コマンドの **-B** フラグとは独立しています。
フィニッシャー・ステープル	Optra W810 プリンターは、オプションのフィニッシャーが取り付けられている場合に、このオプションをサポートします。**y** の有効な値は、次のとおりです。 **-y 0** オフ **-y 1** オン
フィニッシャー・オフセット	Optra W810 プリンターは、オプションのフィニッシャーが取り付けられている場合に、このオプションをサポートします。**e** の有効な値は、次のとおりです。 **-e 0** オフ **-e 1** オン
穴パンチ	Optra W810 プリンターは、オプションのフィニッシャーが取り付けられている場合に、このオプションをサポートします。**o** の有効な値は、次のとおりです。 **-o 0** オフ **-o 1** オン
出力先	**=** (等号) は、出力先を指定するためのコマンド・ライン・オプションです。有効な値は、次のとおりです。 **0** 標準ビン **1** ビン 1 **2** ビン 2 **3** ビン 3 **4** ビン 4 **5** ビン 5 **6** ビン 6 **7** ビン 7 **8** ビン 8 **9** ビン 9 **10** ビン 10 出力先のデフォルト値は、標準ビン (**0**) です。

Lexmark Plus プリンター・モデル 2380-3、2381-3、2390-3、2391-3

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

製品固有の情報

項目	説明
給紙機構	**qprt** コマンドの **-u** フラグを使用した給紙機構の選択がサポートされています。 **-u 1** トラクター 1 **-u 2** トラクター 2 バナー・ページとトレーラー・ページは、印刷ジョブと同じ給紙機構を使用します。トラクターの切り替えは、プリンターが有人のときに行うことをお勧めします。
ピッチ、フォント、および品質	ピッチの選択がサポートされており、ピッチは **qprt** コマンドの **-p** フラグ、フォント名は **-s** フラグ、印刷品質は **-q** フラグを使用して選択できます。サポートされるデフォルト値には、次のものがあります。 **10** ピッチ **courier** フォント **品質** 1 (ドラフト) 有効なフォント値には、次のものがあります。 **フォント名** **-s** fast draft **-s** draft **-s** courier **-s** gothic **-s** prestige (239x のみ) **-s** presenter (239x のみ) **-s** orator (239x のみ) **-s** script (239x のみ) 有効な品質値には、次のものがあります。 **品質 (-q フラグ)** **0** fast draft **1** draft **2** レター品質 (238x のみ) **2** 高品質 (239x のみ) **3** 拡張高品質 (239x のみ) 有効なピッチ値は、10、12、17、20、および 24 です (239x の場合のみ)。 **注:** 1. draft または fast draft を選択すると、選択済みのフォントが指定変更されます。 2. **-e** フラグと強調印刷を使用すると、太字フォントがサポートされます。**-k** フラグとイタリック印刷を使用すると、イタリック・フォントがサポートされます。
ページ幅	**-w** フラグは、印刷可能ページの幅を文字数単位で制御します。 **Plus プリンター** デフォルト **2380 および 2390** 80 **2381 および 2391** 136

OKI MICROLINE 801PS/+F、801PSII/+F、800PSIILT

それぞれのプリンターおよびキュー・システムごとに、製品固有の情報を示します。

日本語の PostScript および ASCII データ・ストリームがサポートされています。日本語のテキスト・ファイルは印刷できません。OKI MICROLINE シリーズのプリンターはすべて、RS-232C ケーブルによって接続されます。

Printronix P9012 ライン・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

Serial Matrix コマンド・セットのみがサポートされています。P-series コマンド・セットはサポートされていません。

QMS ColorScript 100 モデル 20 プリンター

QMS ColorScript 100 モデル 20 プリンターは、カラーの PostScript ファイルと HPGL (Hewlett-Packard Graphics Language) ファイルを印刷できます。

HPGL エミュレーターは、プリンターに付属して出荷される DOS ディスケットに収録されています。PostScript データ・ストリームを使用して ASCII ファイルを印刷することもできます。

PostScript ファイルを印刷する場合は、印刷キューを作成する際に HPGL データ・ストリームを示す印刷キュー名を入力しないでください。HPGL ファイルを印刷するには、次の手順で行います。

1. 印刷キューを作成する際に、HPGL データ・ストリームを示す印刷キュー名を入力します。
2. 「HPGL Emulator」というラベルが付いた 3.5 インチ・ディスケットをディスケット・ドライブに挿入します。
3. root ユーザーとしてログインしていることを確認します。
4. 次のように入力して、HPGL エミュレーターのファイルを DOS ディスケットから適切なディレクトリーにコピーします。

   ```
   /usr/lib/lpd/pio/etc/pioqms100 -Q
   ```

HPGL 印刷ファイルが HPGL 印刷キューに対して実行依頼されると、システムは HPGL エミュレーターをプリンターにダウンロードし、必要に応じてエミュレーターを選択します。

PostScript ファイルを HPGL 印刷キューに実行依頼することもできます。システムがこれらのファイルを HPGL ファイルでなく PostScript として認識できるように、これらのファイル名の先頭は 2 文字の **%!** という文字列にする必要があります。

Texas Instruments OmniLaser 2115 ページ・プリンター

プリンターおよびキュー・システムの製品固有の情報を示します。

プリンター・データ・ストリーム (PostScript、HP LaserJet+、Diablo 630、TI 855、プロッター) の自動選択はサポートされていません。制御パネルを使用して、データ・ストリームを手動で選択する必要があります。

PostScript データ・ストリームを使用して ASCII ファイルを印刷することもできます。

TI 855 ソフトウェア・インターフェースの場合は、DP モードのみがサポートされます。WP モードはサポートされません。

プリンター制御装置の電源をオンにするたびに、次のように入力します。

```
splp -F! lpx
```

ここで、*lpx* は **lp0** などのプリンター・デバイス名です。このコマンドは、HPGL エミュレーターをプリンターに再度ダウンロードする必要があることをシステムに指示します。

システムをリブートする際に、プリンターをオフにしてからオンにして再初期化してください。

System V プリンターの構成

System V プリンターの構成は、AIX プリンターの構成とはいくつかの点で異なります。

高度な印刷サービス機能には、次のようなものがあります。

- 224 ページの『印刷フィルター』
- 237 ページの『PostScript プリンター』

System V 印刷サービス

System V 印刷サービスは、システム管理者 (またはプリンター管理者) がシステム上でプリンターを構成、モニター、および制御するために役立つ一連のユーティリティーです。

印刷サービスには、次の機能があります。

- ユーザーが印刷するファイルを受信する
- ファイルを正しく印刷できるようにフィルターに掛ける (必要な場合)
- 1 つ以上のプリンターの作業をスケジュールに入れる
- プリンターとインターフェースを取るプログラムを開始する
- ジョブの状況を追跡する
- プリンターの問題についてアラートを出す
- 用紙とフィルターのマウント状況を追跡する
- 問題が発生したときにエラー・メッセージを出す

ユーザーがプリンターにファイルを送ると、印刷サービスは要求 (*印刷ジョブ*) に固有の名前 (*要求 ID*) を割り当てます。

要求 ID は、ファイルを印刷するプリンターの名前と、ファイルを識別する固有の番号で構成されます。印刷ジョブの状況を調べたり、印刷ジョブを取り消したりするために、この要求 ID を使用します。印刷サービスは、要求ログにあるすべての印刷要求を追跡します。

印刷ジョブは、プリンターに送信される他の印刷ジョブとともにスプール (または整列) されます。それぞれの印刷ジョブが処理され、印刷される順番を整列して待ちます。この保留されている印刷ジョブの列を*印刷キュー* と呼びます。

それぞれのプリンターに固有のキューがあります。ジョブをキュー内で保持したり、キュー内のジョブを上位に移動したり、ジョブを別のキューに転送したりすることができます。

印刷要求処理

印刷要求は、すべての印刷ジョブを追跡するスプーリング・デーモン (バックグラウンド・プログラム) に送信されます。

次の図に示すように、それぞれの印刷要求は、すべての印刷ジョブを追跡するスプーリング・デーモン (バックグラウンド・プログラム) に送信されます。(この情報は要求ログに保存されます。) このデーモンは、印刷サービスの開始時に作成されます。スプーリング・デーモンは、プリンターと低速フィルターの状況を追跡する役割も果たします。プリンターがジョブの印刷を完了すると、デーモンは別のジョブがキューに入れられていればそのジョブの印刷を開始します。

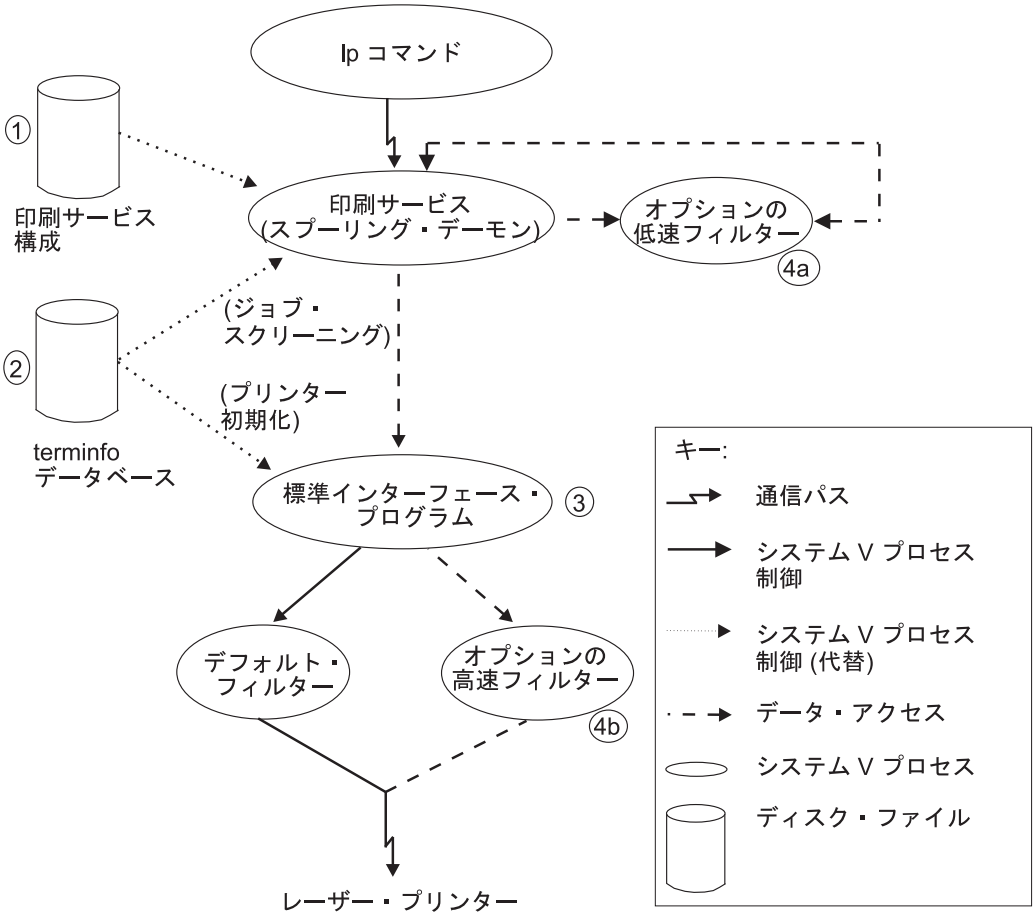

図 6. 印刷要求処理の概要

「印刷要求処理の概要」の図に示されているいくつかの項目を調整したり置き換えたりすることによって、印刷サービスをカスタマイズできます (次のリストの番号は、図の中にある番号付きの項目に対応しています)。

1. 印刷サービス構成: ほとんどのプリンターの場合、ディスクに保管されているプリンター構成のみを変更する必要があります。ローカル・プリンターの追加または変更については、**lpadmin** コマンドを参照してください。
2. **terminfo** データベース: **terminfo** データベースに登録されていないプリンターの場合は、そのプリンターの機能を記述する新規エントリーを追加できます。 220 ページの『terminfo データベースへのプリンター・エントリーの追加』を参照してください。印刷サービスは、並列して実行される 2 つの機能に **terminfo** データベースを使用します。これらは、受信済みの要求が目的のプリンターによって処理されるように印刷要求を選別する機能と、要求を印刷する準備が整うようにプリンターを設定する機能です。

例えば、ユーザーによって要求されたページ長を設定できるプリンターが **terminfo** データベースに示されていない場合、スプーリング・デーモンは要求を拒否します。一方、その設定が可能なデータベースが示されていれば、インターフェース・プログラムは同じ情報を使用してプリンターを初期化します。

3. 標準インターフェース・プログラム: 特に複雑なプリンターを使用している場合、または印刷サービスに備わっていない機能を使用する場合は、インターフェース・スクリプトを変更できます。このスクリプトは、プリンターの管理を行います。つまり、バナー・ページを印刷し、プリンターを初期化し、ユーザーのファイルのコピーをプリンターに送信するためのフィルターを呼び出します。
4. オプション・フィルター: システム上で使用されているアプリケーションとプリンターとの間でリンクを提供するために、低速フィルターと高速フィルターを追加できます。各タイプのフィルターは、ファイルを別の形式に変換でき (例えば、エスケープ・シーケンスのセットを別のセットにマップする)、ユーザーが要求した印刷モードを解釈することによって特殊なセットアップを行うことができます。低速フィルターは、プリンターの拘束を避けるために、スプーリング・デーモンによって別個に実行されます。高速フィルターを実行すると、その出力はプリンターに直接送られます。このため、これらのフィルターはプリンターを制御できます。

印刷要求ログ

ユーザーがプリンターにジョブを送信するたびに、印刷サービスはジョブ要求を記述するファイルを 2 つ作成し、/usr/spool/lp/temp ディレクトリーと /usr/spool/lp/requests ディレクトリーに 1 つずつ配置します。

ジョブに関する情報は 2 つのファイルに分割されるので、システムは機密情報を /usr/spool/lp/requests ディレクトリーに安全に保管できます。ジョブを実行依頼したユーザーは、/usr/spool/lp/temp 内の要求ファイルにアクセスできます。/usr/spool/lp/requests 内のファイルにアクセスできるユーザーは、プリンター管理者 (または *root* ユーザー) のみです。

要求ファイルは、ジョブがキュー内にある間のみ、これらのディレクトリーに残ります。ジョブが印刷を終了すると、2 つのファイルの情報が結合され、要求ログ /usr/spool/lp/logs/requests に追加されます。

要求ログは構造化されているので、一般的なシェル・コマンドを使用してデータを抽出できます。要求は印刷された順にリストされ、要求 ID から始まる行で区切られています。区切り行の下にある各行には、それぞれ単一の文字のマークが付けられています。このマークは要求ログ・コード で、行に含まれている情報の種類を示しています。それぞれの文字は、シングル・スペースによってデータと区切られています。エントリー例の下にある表は、これらのコードの説明です。

次に、印刷要求ログからのエントリー例を示します。

```
= ps-717, uid 1532, gid 18, size 7872, Tue May 10 14:43:10 1994
z ps
C 1
D ps
F /usr/spool/lp/temp/717-1
P 20
t simple
U hanna
s 0x0010
```

印刷要求ログ・エントリー

文字	行の内容
=	区切り行は、(コンマ区切りの) 要求 ID、要求を実行依頼したユーザーのユーザー ID (**uid**) とグループ ID (**gid**)、オリジナルの (フィルターに掛けられていない) ファイルの合計バイト数 (**size**)、および要求がキューに入れられた日時をリストします。
C	印刷部数。
D	プリンターまたはクラスの出力先、または `any` という単語。
F	`/usr/spool/lp/temp` ディレクトリー内のファイルの名前。この行は印刷されるそれぞれのファイルごとに繰り返され、ファイルは指定された順に印刷されます。
f	使用されたフォーム名 (該当する場合)。
H	使用された特殊処理のタイプ: • **resume (再開)** • **hold (保留)** • **immediate (即時)**
N	ファイルの印刷後に印刷サービスがユーザーにどの方法で通知したか (該当する場合): • **M** 電子メール・メッセージ • **W** ユーザーの端末装置に書き込まれるメッセージ
O	**lp** コマンドに指定された **-o** オプション。
P	印刷要求の優先順位 (該当する場合)。
p	印刷されたページのリスト。
r	ユーザーがファイルのロー処理を要求したことを示す、**lp** コマンドに指定された **-r** オプション。
S	使用された文字セット。
s	ジョブの結果 (個々のビットの組み合わせとして 16 進数形式で表される)。スプーラー内部で使用される重要なビットは、次のとおりです。 • **0x0004** 低速フィルターが正常に完了した。 • **0x0010** 印刷が正常に完了した。 • **0x0040** 要求が取り消された。 • **0x0100** フィルターまたは印刷の要求が失敗した。
T	バナー・ページのタイトル。
t	ファイルのコンテンツ・タイプ。
U	印刷要求を実行依頼したユーザーの名前。
x	低速フィルター。
Y	要求の印刷に使用されるフィルターに指定する特殊モードのリスト。
y	高速フィルター。
z	要求に使用されたプリンター。要求が *any (任意の)* プリンターまたはプリンター・クラスへのキューに入れられていた場合、またはプリンター管理者が要求を別のプリンターに転送した場合は、このプリンターは出力先 (**D** 行) と異なります。

印刷サービス・コマンド

すべてのユーザーが使用できる印刷サービス・コマンドの要約を示します。

コマンド	説明
cancel	印刷されるファイルの要求を取り消します。
lp	プリンターにファイルを送信します。
lpstat	印刷サービスの状況を報告します。

プリンターの誤動作が起こったときに、ユーザーが管理者を呼び出さなくてもプリンターをオフにできるように、管理者はプリンターを使用不可および使用可能にする権限をユーザーに与えることができます。 (ただし、印刷環境によっては、通常のユーザーがプリンターを使用不可にすることを許可すべきではない場合もあります。)

「管理用印刷サービス・コマンド」の表は、管理者のみが使用できる印刷サービス・コマンドのリストです。管理コマンドを使用するには、root ユーザーとしてログインする必要があります。

管理用印刷サービス・コマンドは、`/usr/lib` ディレクトリーにあります。これらのコマンドを頻繁に使用する場合は、**PATH** 変数に `/usr/lib` を含めてください。

管理用印刷サービス・コマンド

コマンド	説明
accept reject	指定された出力先へのキューにジョブを入れることを許可します。指定された出力先へのキューにジョブを入れることを禁止します。
cancel	ライン・プリンターへの要求を取り消します。
enable disable	指定されたプリンターを活動化します。
lpadmin	プリンター構成をセットアップまたは変更します。
lpc	(BSD) ライン・プリンターの制御を提供します。
lpfilter	フィルター定義をセットアップまたは変更します。
lpforms	事前印刷用紙をセットアップまたは変更します (用紙をマウントするには `/usr/sbin/lpadmin` を使用します)。
lpmove	出力要求を別の出力先に移動します。
lpsched lpshut	印刷サービスを開始します。印刷サービスを停止します。
lpsystem	リモート・システムを印刷サービスに登録します。
lpusers	印刷サービスのユーザーが要求できる、デフォルトの優先順位と優先順位の制限を設定または変更します。

デフォルトのプリンターのページ・サイズとスペーシング

ユーザーがファイルの印刷要求を実行依頼する際に、ページ・サイズ、文字ピッチ、および行ピッチ (スペーシング) は、通常はそのファイルが印刷される対象の用紙から決定されます。

ユーザーが用紙を要求しない場合、ユーザーは使用するページ・サイズと印刷スペーシングを指定できます。ユーザーが使用する用紙もページ・サイズと印刷スペーシングも指定しない場合は、デフォルトが使用されます。

注: 前記の情報は、PostScript プリンターには適用されません。

それぞれのプリンターごとにデフォルトを設定することによって、印刷要求の実行依頼を容易にすることができます。例えば、異なるデフォルト・ページ・サイズや印刷スペーシングを使用する、さまざまなプリンターを指定できます。1 つのプリンターを広幅 (132 列) 出力専用にし、別のプリンターを通常 (80 列、66

行) 出力専用に、さらに別のプリンターをモノスペース・フォント (12 字/インチ、8 行/インチ) の文字印刷専用にすることができます。ユーザーは該当するプリンターにファイルを送付するだけで、必要な出力のスタイルを得ることができます。

次のデフォルト設定を指定できます。

- ページ幅
- ページ長
- 文字ピッチ
- 行ピッチ

最初の 2 つは、それぞれ列数および行数として (またはインチ単位で) 指定します。文字ピッチと行ピッチは、それぞれ字/インチ (cpi) および行/インチ (lpi) で指定します。

さらに、文字ピッチは、10 cpi の場合は **pica**、12 cpi の場合は **elite**、プリンターが印刷できる最大 cpi (限度は 30 cpi まで) の場合は **compressed** と指定します。

デフォルト設定を指定するには、次のコマンドを使用します。

```
/usr/sbin/lpadmin -p printer_name -o width=scaled-number

/usr/sbin/lpadmin -p printer_name -o length=scaled-number

/usr/sbin/lpadmin -p printer_name -o cpi=scaled-number

/usr/sbin/lpadmin -p printer_name -o lpi=scaled-number
```

注: lpadmin コマンドは、プリンター・タイプを使用して、プリンターに対して可能な設定値を判別します。このため、これらのデフォルトを指定するには、その前にプリンター・タイプを設定する必要があります。

例えば、プリンター barney に対して、ページ幅は 11 インチ、ページ長は 14 インチ、文字ピッチは compressed、行ピッチは 3 行/インチと指定するには、次のように入力します。

```
/usr/sbin/lpadmin -p barney -o width=11i

/usr/sbin/lpadmin -p barney -o length=14i

/usr/sbin/lpadmin -p barney -o cpi=compressed

/usr/sbin/lpadmin -p barney -o lpi=3
```

デフォルトを指定しない場合は、ページ・サイズと印刷スペーシングはプリンターの初期化時に使用可能なものに設定されます。まずユーザー独自のデフォルトを指定せずにプリンター構成を定義し、次に **lpstat** コマンドを使用してプリンター構成を表示することによって、どの値がデフォルトか判別できます。デフォルト・ページ・サイズと印刷スペーシングを表示するには、次の手順で行います。

```
lpstat -p printer_name -l
```

次のような情報が表示されます。

```
   Default pitch: compressed CPI 3 LPI
   Default page size: Default page size: 11i wide 14i long
```

デフォルトを設定しない場合、**lpstat** コマンドはそのプリンターに関する **terminfo** データベース・エントリーからのデフォルトを報告します。

バナーの構成

バナーは、印刷ジョブによって印刷される印刷要求 (例えば、プリンター名、ユーザー、日付) を記述するページです。

バナー は、印刷ジョブによって印刷される印刷要求 (例えば、プリンター名、ユーザー、日付) を記述するページです。

バナーを印刷するかどうかユーザーが決定できるようにするには、次のコマンドを使用します。

```
/usr/sbin/lpadmin -o nobanner
```

/etc/lp/Systems ファイルの管理

BSD 接続を使用する任意のシステムに接続できます。

BSD 接続を使用する任意のシステムへの接続を可能にする、デフォルトのワイルドカード・エントリーが /etc/lp/Systems ファイルに提供されています。ワイルドカード・エントリーは次のとおりです。

```
*:x:-:bsd:-:n:10:-:-:Allow all BSD connections
```

この項目が存在すると、印刷サーバーは既知のシステムとして明示的に構成されていないシステムからの接続を受け入れます。

さらに、**lpsystem** コマンドを使用して、リモート・システム名を指定するエントリーをファイルに追加できます。

プリンターへのアクセスは、プリンターの users.allow ファイルと users.deny ファイルから制御できます。詳しくは、**lpadmin** コマンドを参照してください。

注: ワイルドカード・エントリーは、着信接続要求のみに使用され、発信要求には使用されません。

システム上にある各プリンターへのアクセスを個別に制御する必要がなく、プリンターにアクセス可能なリモート・システムを /etc/lp/Systems ファイルによって指定する場合は、ファイルからワイルドカード・エントリーを除去し、リモート・システムのエントリーを追加します。このためには、次のように行います。

- BSD システムのワイルドカード・エントリーを除去するには、次のコマンドを使用します。

  ```
  /usr/sbin/lpsystem -r "*"
  ```

- 特定のリモート・システムのエントリーを追加するには、次のコマンドを使用します。

  ```
  /usr/sbin/lpsystem system-name
  ```

プリンター・モデル・ファイル

プリンター・モデル・ファイル /usr/lib/scoadmin/printer/model.stz は、サポートされるプリンターとその属性のリストを保管します。

```
/usr/lib/scoadmin/printer/model.stz
```

フォーマットは次のとおりです。

```
key1:
attr1=val1
attr2=val2
key2:
```

...

このファイルには、サポートされるそれぞれのメークとモデルのテキスト記述、プリンター・インターフェース・スクリプトへのポインター、および **terminfo** エントリーが含まれています。また、必要なシリアル・ライン設定や、サポートされるコンテンツ・タイプなどの追加情報も含んでいる場合があります。

次の属性が使用可能です。

項目	説明
contents	引用符で囲まれ、コンマで区切られた、プリンターのサポート対象コンテンツ・タイプのリスト。通常は、**PS** (postscript の場合)、**pcl** (Hewlett-Packard の Printer Control Language の場合)、または **simple** (その他ほとんどのプリンターの場合) のみ。印刷フィルターを作成することによって、追加のコンテンツ・タイプをサポートできます。224 ページの『印刷フィルター』を参照してください。この属性はオプションです。
interface	プリンター・インターフェース・スクリプトは、ほとんどの場合は **standard** に設定する必要があります。『プリンター・インターフェース・スクリプト』を参照してください。この属性は必須です。
name	プリンターのメーク/モデルのテキスト記述。この属性は必須です。
terminfo	このプリンターに関連した **terminfo** エントリーの名前。28 ページの『terminfo データベース』を参照してください。この属性はオプションです。
stty	このプリンターに該当するシリアル・ラインまたはパラレル・ラインの設定値。この属性はオプションです。

また、次の例に示すように、それぞれのプリンターに固有キー名を関連付ける必要があります。

```
canon-jet-10ex:
name="Canon Bubble Jet 10ex"
terminfo=bj-10ex
interface=standard

hp-laserjet:
name="HP LaserJet (PCL)"
terminfo=hplaserjet
interface=standard
contents=pcl
stty="clocal -onlcr"
```

プリンター・インターフェース・スクリプト

プリンター・インターフェース・スクリプト は、印刷サービスがファイルを印刷するたびにプリンターの管理に使用するプログラムです。インターフェース・スクリプトは、プリンターを初期化し、プリンターの特定の機能を利用し、ファイルを印刷して、エラーを報告します。

注: 旧型の UNIX システムの印刷サービスと組み合わせて使用していたインターフェース・プログラムがある場合、そのプログラムは引き続き機能します。ただし、いくつかの **-o** オプションは*標準化* されており、すべてのインターフェース・プログラムに渡されることに注意してください。インターフェース・プログラムが使用する同様の名前のオプションに、これらのオプションが干渉する可能性があります。

プリンター・インターフェース・スクリプトはプリンター・モデルに関連しており、`/etc/lp/model` にあります。例えば、PostScript プリンター用のプリンター・インターフェース・スクリプトは、`/etc/lp/model/PS` という名前のものです。ユーザー独自のインターフェース・スクリプトを作成したり、既存のものをニーズに合わせてカスタマイズしたりすることもできます。217 ページの『プリンター・インターフェース・スクリプトの作成』を参照してください。

インターフェース・スクリプトは、次の処理を行います。

- プリンター・ポート (コンピューターとプリンターの間の接続) を初期化します。*standard* (/etc/lp/model/standard) インターフェース・スクリプトは、**stty** コマンドを使用してプリンター・ポートを初期化します。詳しくは、**stty** コマンドを参照してください。
- 物理プリンターを初期化し (前に印刷されたファイルが異常な状態のままになっている場合には、プリンターを正常な状態に復元する)、ユーザーから要求された文字ピッチ、行ピッチ、ページ・サイズ、および文字セットを設定します。*standard* インターフェース・スクリプトは、**lp.set** コマンドを使用してプリンターを初期化します。詳しくは、**lp.set** コマンドを参照してください。
- バナー・ページを印刷します (必要な場合)。
- 要求されたファイルを印刷します。*standard* インターフェース・スクリプトは、**lp.cat** コマンドを呼び出してファイルを印刷します。詳しくは、**lp.cat** コマンドを参照してください。
- エラーがあれば印刷サービスに報告します。*standard* インターフェース・スクリプトは、**lp.tell** コマンドを使用して、プリンター障害の記述を印刷サービスに送信します。印刷サービスは、その情報をアラートとして印刷管理者に転送します。詳しくは、**lp.tell** コマンドを参照してください。

印刷サービスは、プリンター・ポートを開きます。印刷サービスは、プリンター・ポートへの接続を標準出力としてインターフェース・スクリプトに提供し、プリンターをインターフェース・スクリプトの制御端末に設定します。ポートのハングアップが発生した場合は、**SIGHUP** シグナルがインターフェース・スクリプトに送信されます。

インターフェース・スクリプトの多くは特殊なオプションを備えており、ユーザーは **lp** コマンドの **-o** オプションを使用してこれらのオプションを指定できます。詳しくは、**lp** コマンドを参照してください。

印刷サービスは、次の例に示すように、インターフェース・スクリプトを実行してプリンターに印刷ジョブを送信します。

```
/etc/lp/interfaces/printer id user title copies options file1 file2 ...
```

インターフェース・スクリプトの引数は次のとおりです。

項目	説明
printer	インターフェース・スクリプトの名前 (プリンター名と同じ)。
id	**lp** コマンドから戻される要求 ID。
user	要求を行ったユーザーのログイン名。
title	ユーザーが指定したオプションのタイトル。
copies	ユーザーが要求した印刷部数。
options	ブランクで区切られたオプションのリスト。これらのオプションはユーザーによって指定されるか (**lp -o** を使用して)、印刷サービスによって指定されます (**lpadmin** コマンドを使用して管理者が指定したデフォルト値から)。*standard* インターフェースが認識するオプションのリストについては、**lp** コマンドを参照してください。
file	印刷されるファイルの絶対パス名。

インターフェース・スクリプトが呼び出されると、次の処理が行われます。

- /dev/null から標準入力を取得します。
- 標準出力がプリンター・ポートに送られます。
- 印刷要求を実行依頼したユーザーに対して表示されるファイルに、標準エラー出力が送られます。

印刷サービスは、次のシェル変数として、追加のプリンター構成情報をインターフェース・スクリプトに渡します。

項目	説明
TERM=*printer-type*	プリンター・タイプを指定します。この値は、拡張 **terminfo** データベースからプリンターの機能に関する情報を取得するためのキーとして使用されます。
FILTER=*pipeline*	要求の内容をプリンターに送信するために使用するフィルターを指定します。このフィルターはプリンターを制御できます。
CHARSET=*character-set*	印刷要求の内容を印刷する際に使用する文字セットを指定します。*standard* インターフェース・スクリプトは、**terminfo** データベースから文字セットを選択するために必要な制御シーケンスを抽出します。

プリンター・インターフェース・スクリプトの作成

terminfo データベースへのエントリーの追加によってサポートされないプリンターを使用している場合、または印刷要件が standard インターフェース・スクリプトや、/etc/lp/model ファイルに提供されているその他のインターフェース・スクリプトによってサポートされない場合は、ユーザー独自のプリンター・インターフェース・スクリプトを作成できます。

カスタマイズしたインターフェース・スクリプトを作成するには、次の手順で行います。

1. standard インターフェース・スクリプト (または /etc/lp/model 内にあるその他のスクリプトの 1 つ) を変更します。次に例を示します。

   ```
   cd /etc/lp/model

   cp standard okidatanew
   ```

2. カスタム・インターフェース・スクリプトが、正しい **stty** モード (ボー・レートや出力オプションなどの端末特性) を設定していることを確認します。次の行から始まるセクションを調べてください。

   ```
   ## Initialize the printer port
   ```

3. standard インターフェース・スクリプト内のコードを変更します。このスクリプトは、次のような行を使用して、印刷サービスまたはユーザーによって指定されるデフォルト・モードと調整済みモードを設定します。

   ```
   stty mode options 0<&1
   ```

   このコマンド・ラインは、**stty** コマンドの標準入力をプリンター・ポートから取得します。例えば、次の **stty** コマンド例は、ボー・レートを 1200bps に設定し、いくつかのオプション・モードを設定します。

   ```
   stty -parenb -parodd 1200 cs8 cread clocal ixon 0<&1
   ```

4. プリンター・ポートのハードウェア・フロー制御特性を設定します。standard インターフェース・スクリプトはハードウェア・フロー制御を設定しません。これはご使用のコンピューター・ハードウェアに応じて設定されます。standard インターフェース・スクリプトのコードに、この特性とその他のプリンター・ポート特性を設定する場所が示されています。次の行から始まるセクションを調べてください。

   ```
   # Here you may want to add other port initialization code.
   ```

5. プリンターによって列数が異なるため、インターフェース・スクリプトのヘッダーとトレーラーがご使用のプリンターに対応していることを確認してください。 standard インターフェース・スクリプトは、80 列のページに収まるバナーを印刷します (例外として、ユーザーのタイトルはこれより長くすることができます)。standard インターフェース・スクリプトのコード内で、次の行から始まるセクションを調べてください。

   ```
   ## Print the banner page
   ```

6. 一部のアプリケーションを特定のプリンターと組み合わせて実行する場合に、改ページをオフにする必要が生じることがあります。改ページをオフにする必要がある場合は、次の行で標準インターフェース・プログラム (/usr/lib/lp/model/standard) を変更できます。

```
if [ -n "${FF}" -a "no" = "${nofilebreak}" ]
```

改ページをオフにするには、**no** を **yes** に変更します。

7. ユーザー関連のエラー・メッセージをすべて標準出力または標準エラー出力に書き込むように、カスタム・インターフェース・スクリプトに指定します。印刷サービスは、標準出力エラーをページに印刷し、標準エラーをユーザーにメールで送信します。
8. 印刷が完了したときに、印刷ジョブの状況を示すコードを出して終了するように、インターフェース・スクリプトに指定します。終了コード表 (『印刷サービスの終了コード』) は、印刷サービスが終了コードを解釈する方法の説明です。

プリンターの障害について管理者にアラートを出す方法の 1 つは、コード 129 を出して終了することです。ただし、インターフェース・スクリプトが終了すると、印刷サービスは障害が解消した後で印刷ジョブを最初から再印刷します。ジョブ全体を再印刷せずに管理者にアラートを出すようにするには、インターフェース・スクリプトが印刷サービスに障害メッセージを送信して、障害が解消するまで待つよう指定します。障害が解消すると、インターフェース・スクリプトはジョブの印刷を再開します。ジョブの印刷が完了すると、インターフェース・スクリプトは障害が発生しなかった場合と同様にゼロを戻して終了します。もう 1 つの利点として、障害が解消したことをインターフェース・スクリプトが自動的に検出できるので、管理者はプリンターを再び使用可能に設定しなくて済みます。

障害メッセージを印刷サービスに送信するように指定するには、**lp.tell** コマンドを使用します。 standard プリンター・インターフェース・コードは、LPTELL シェル変数を指定して **lp.tell** コマンドを呼び出します。**lp.tell** プログラムは、標準入力を印刷サービスに送信します。印刷サービスは、メッセージをアラートとして管理者に転送します。標準入力が空の場合、**lp.tell** はアラートを開始しません。**lp.tell** (**LPTELL**) プログラムの使用法の例については、standard インターフェース・スクリプト内で次のコメントの直後にあるコードを検討してください。

```
# Here's where we set up the $LPTELL program to capture
        # fault messages.
        #
        # Here's where we print the file.
```

特殊終了コード 129 または **lp.tell** の場合、インターフェース・スクリプト自体がプリンターを使用不可に設定する必要はありません。インターフェース・スクリプトがプリンターを直接使用不可に設定することはできますが、このようにすると障害アラート・メカニズムが指定変更されます。アラートは、プリンターに障害が発生したことを印刷サービスが検出した場合のみ送信され、その主な検出手段は特殊終了コードと **lp.tell** プログラムです。

印刷サービスがファイルの印刷をいずれかの時点で中断する必要がある場合は、シグナル 15 を使用してインターフェース・スクリプトを強制終了します (詳しくは、**signal** コマンドと **kill** コマンドを参照)。

他のいずれかのシグナルを受信した時点でインターフェース・スクリプトが停止した場合、印刷サービスは、以降の印刷ジョブは影響を受けないと想定し、プリンターの使用を継続します。印刷サービスは、ジョブが正常に終了しなかったことを印刷ジョブの送信者に通知します。

シグナル **SIGHUP**、**SIGINT**、**SIGQUI**、および **SIGPIP** (トラップ番号 1、2、3、および 13) は、インターフェースの呼び出し時には無視されます。standard インターフェース・スクリプトはこの動作を変更して、適切な時点でこれらのシグナルを捕捉し、これらのシグナルがプリンターに問題があることを意味していると解釈して、障害を発行するようにします。

印刷サービスの終了コード

印刷サービスは、いくつかの終了コードを解釈します。

次の表で、印刷サービスが終了コードを解釈する方法を説明します。

コード	説明
0	印刷ジョブは正常に完了しました。
1 から **127**	印刷サービスは、ジョブの印刷中に問題を検出しました (例えば、過大な数の印刷不能文字があった、ジョブがプリンターの能力を超過したなど)。この問題は、以降の印刷ジョブに影響を及ぼしません。印刷サービスは、印刷ジョブの実行依頼者に、ジョブの印刷中にエラーが発生したことを通知する必要があります (**write** または **mail** のどちらかを使用して)。プリンターの障害が発生していた場合、その障害は解消されました。
128	印刷サービス内部で使用するために予約済みです。インターフェース・スクリプトがこのコードを出して終了してはなりません。
129	印刷サービスは、ジョブの印刷中にプリンター障害を検出しました。この問題は、以降の印刷ジョブに影響を及ぼします。プリンターの障害からリカバリーするために、印刷サービスが管理者による問題の修正を待つ必要がある場合、印刷サービスはプリンターを使用不可にする必要があります。印刷を継続することによって障害からリカバリーする場合は、印刷サービスはプリンターを使用不可に設定してはならず、数分後に印刷を再試行する必要があります。
> 129	印刷サービス内部で使用するために予約済みです。インターフェース・スクリプトがこの範囲のコードを出して終了してはなりません。

プリンター・インターフェース・プログラム

デフォルトでは、印刷サービスは標準インターフェース・スクリプト `/etc/lp/model/standard` を使用します。このインターフェース・スクリプトは、ほとんどの印刷要件の処理に使用します。

プリンターを追加した後でインターフェース・スクリプトを変更するには、**lpadmin** コマンドの **-i** オプションを使用して、インターフェース・プログラムを指定できます。詳しくは、**lpadmin** コマンドを参照してください。

次の例では、プリンター・ポート `/dev/tty01` 上の新規プリンター **laser** を追加します。この例では、ディレクトリー `/usr/doceng/laser_intface` にある、カスタマイズされたインターフェース・プログラムを使用します。このプログラムは **i10**、**i300**、および **impress** の 3 つのファイル・タイプを処理でき、ユーザー **doceng** および **docpub** のみが使用できます。(次のコマンド例は、読みやすさのために複数行に分割されています。)

```
lpadmin -p laser -v /dev/tty01 \

        -i /usr/doceng/laser_intface \

        -I "i10,i300,impress" \

        -u "allow:doceng,docpub"
```

terminfo データベース

印刷サービスは、標準インターフェース・スクリプトと **terminfo** データベースを使用して、それぞれのプリンターの初期設定、および選択されたページ・サイズ、文字ピッチ、行ピッチ、および文字セットのセットアップを行います。

このため、印刷サービスに新規プリンターを追加するには、通常は **terminfo** データベース (`/usr/lib/terminfo/terminfo.lp`) に正しいエントリーが存在していれば十分です。

terminfo データベースは、それぞれのプリンターをショート・ネームによって識別します。これは、**TERM** シェル変数の設定に使用される名前と同じ種類のものです。例えば、AT&T モデル 455 プリンターを示す **terminfo** データベース内での名前は **455** です。

ご使用のプリンターの **terminfo** タイプを指定するには、**lpadmin** コマンドの **-T** オプションを使用します。デフォルトで、**terminfo** データベースには多くの一般的なプリンターに対応するエントリーが含まれています。ご使用のプリンターに対応する **terminfo** タイプを選択してください。

terminfo にご使用のプリンターに対応するエントリーが含まれていない場合でも、印刷サービスに対してそのプリンターを使用できることがあります。ただし、ページ・サイズ、ピッチ、および文字セットの自動選択は使用できず、また印刷要求のたびにプリンターが正しいモードに設定されるように維持したり、プリンターに対してプリンター・フォームを使用したりする際に問題が生じることがあります。この場合は、ご使用のプリンターに対応するエントリーを *terminfo* に追加するか (『terminfo データベースへのプリンター・エントリーの追加』)、カスタマイズしたインターフェース・プログラムを作成して (217 ページの『プリンター・インターフェース・スクリプトの作成』)、プリンターに対して使用できます。

それぞれの端末装置またはプリンターごとに、多数の項目を **terminfo** データベースに定義できます。ただし、印刷サービスが使用するものはこれらのうち 50 に満たない数で、ほとんどのプリンターはさらに少数しか必要としません。次のコマンドを入力すれば、特定の **terminfo** エントリーに対して定義された項目を確認できます。

```
infocmp terminfo_name
```

terminfo データベースへのプリンター・エントリーの追加

プリンターの terminfo エントリーを作成できます。

ご使用のプリンターの *terminfo* エントリーを作成するには、次の手順で行います。

1. `/usr/lib/terminfo/terminfo.lp` ファイル内で、追加しているプリンターと同じコマンドを使用するエントリーを識別し、その情報を *filename* にコピーします。ここで *filename* は、プリンターに対して作成した *terminfo* エントリーを含むファイルです。
2. ご使用のプリンターのマニュアル、『プリンターの terminfo エントリーの定義』、および **terminfo** に記載されている情報を使用して、*filename* 内のエントリーを変更します。
3. 新規エントリーを作成した後、次のようにしてデータベースにコンパイルします。

   ```
   tic filename
   ```

terminfo エントリーを追加または削除した後、またはピッチ設定、ページの幅と長さ、または文字セットを制御する値を変更した後は、印刷サービスを停止し、再始動してください。

プリンターの terminfo エントリーの定義

次に、印刷サービスの terminfo エントリーとその定義を示します。

terminfo エントリー	**説明**
ブール:	
daisy	プリンターがオペレーターによる文字セットの変更を必要としている
数値:	
bufsz	印刷前にバッファーに入れられるバイト数
* cols	1 行の列数
* it	# 個のスペースごとに初期のタブを設定する
* lines	ページの行数
orc	水平解像度 (1 文字ごとのユニット数)
orhi	水平解像度 (インチごとのユニット数)
orl	垂直解像度 (1 行ごとのユニット数)
orvi	垂直解像度 (インチごとのユニット数)
cps	平均印刷速度 (字/秒)
文字列:	

terminfo エントリー	説明
* cr	復帰
cpi	字/インチ数を変更する
lpi	行/インチ数を変更する
chr	水平解像度を変更する
cvr	垂直解像度を変更する
csnm	文字セット名のリスト
mgc	すべてのマージン (上部、下部、サイド) のクリア
* hpa	絶対水平位置
* cud1	1 行下へ
* cuf1	キャリッジの右方への移動
swidm	横倍角印刷を使用可能にする
rwidm	横倍角印刷を使用不可にする
* ff	ページ替え
* is1	プリンター初期化ストリング
* is2	プリンター初期化ストリング
* is3	プリンター初期化ストリング
* if	初期設定ファイルの名前
* iprog	初期化プログラムのパス名
* cud	キャリッジを # 行下に移動する
* cuf	キャリッジを # 列右に移動する
* rep	文字を # 回繰り返す
* vpa	絶対垂直位置
scs	文字セットを選択する
smgb	下部マージンを現在行に設定する
smgbp	下部マージンを設定する
* smgl	左マージンを現在列に設定する
smglp	左マージンを設定する
* smgr	右マージンを現在列に設定する
smgrp	右マージンを設定する
smgt	上部マージンを現在行に設定する
smgtp	上部マージンを設定する
scsd	文字セットの定義を開始する
* ht	次の 8 スペースのタブ停止位置までタブで移動する

アスタリスク (*) のマークが付いた項目は、ご使用のシステムで使用可能です。その他の定義は追加できます。

プリンター・フォーム

印刷サービスには、フォームを作成および管理する機能が組み込まれています。

事前印刷のプリンター・フォーム は、プリンターに装てんするブランクの用紙です。アプリケーションは通常、ブランクの用紙に印刷されたときにその用紙の空欄を埋めるファイルを生成します。

フォームのフォーマットを指定するには、フォーム記述ファイルを作成します。

例えば、`/tmp/check.desc` という名前のファイルを作成し、次の情報のすべて、または任意のサブセットを取り込むとします。

```
Page length: 66
Page width: 80
Number of pages: 2
Line pitch: 10
Character pitch: 16
Character set choice: any
Ribbon color: blue
Comment:
       Check form
```

```
Alignment pattern:

      XXXX XXXXXXXXXXXX XXXXXXXXX
                                 xxxxxx
      xxxxxxxxxxxxxxxxxxxxxxxxx
```

印刷サービスは、印刷を開始する前に位置合わせパターンを使用してフォームを調整し、印刷の前に位置合わせを行うようにプロンプトを出します。

ご使用のプリンターに応じて、行数、インチ (i)、またはセンチメートル (c) 単位でページ長を指定します。ページ幅は、列数、インチ (i)、またはセンチメートル (c) 単位で指定します。前記の例では、ページ長は 66 行として指定されています。プリンターがインチを認識する場合は、ページ長を 11i として指定します。

印刷サービスへのフォームの追加

印刷サービスにフォームを追加できます。

フォームを作成した後、サービスにフォームを追加する必要があります。フォームをマウントする際にこの名前を使用するので、フォームを説明する名前を選択してください。次のコマンドは、/tmp/check.desc フォームを追加します。

```
lpforms -f check -F /tmp/check.desc
```

このコマンドは、フォームを /usr/spool/lp/admins/lp/forms ファイルに配置します。

フォームの除去

印刷サービスは、定義できるフォームの数に固定の限度を設けていません。

適用できなくなったフォームは除去することをお勧めします。除去しないと、ユーザーがフォームを選択する際に、使われなくなったフォームの長いリストを検討しなければならず面倒です。また、印刷サービスは一部のタスクを実行する前に、リストされているフォームをときどきすべて検索する必要があるので、使われなくなったフォームを除去しないでおくと、印刷サービスが不必要な処理を要求される可能性があります。

フォームを除去するには、次のコマンドを入力します。

```
/usr/sbin/lpforms -f form-name -x
```

フォームへのユーザー・アクセス

選択されたユーザーのみが特定のフォームを使用できるように制限できます。

例えば、小切手へのアクセスを給与計算部門または買掛管理部門の担当者のみに制限できます。

印刷サービスは、フォームへのアクセスを許可または拒否されるユーザーのリスト (管理者が設定する) を使用して、そのフォームを使用できるユーザーを制限します。ユーザーが特定のフォームの使用を許可されていない場合、印刷サービスはそのフォームを使用してファイルを印刷する要求をリジェクトします。

フォームへのユーザーのアクセスを許可または拒否するために使用される方式は、**cron** 機能と **at** 機能へのユーザーのアクセスを許可または拒否するために使用される方式と同様です。詳しくは、**at**、**cron**、および **crontab** の各コマンドを参照してください。

システムのユーザーがリモート・プリンターのフォームにアクセスできるようにする場合は、ローカル・システムの許可リストに含まれているすべてのユーザーを、リモート・システムの許可リストにも含める必要があります。

一方で、ローカル・ユーザーがリモート・プリンターのフォームを使用することを拒否する場合は、ローカルとリモートの両方の印刷サービスで拒否リストにそのユーザーを含める必要はありません。これらの拒否リストのどちらか一方に含まれていれば、ユーザーはリモート・フォームへのアクセスを拒否されます。ただし、ユーザーに分かりやすいように、リモート・システム上の拒否リストに含まれているローカル・ユーザーは、ローカル・システム上の対応する拒否リストにも含めるようにしてください。これにより、システム上のユーザーが許可のないフォームを要求したときに、フォームの使用を拒否されていることがそのユーザーに即時に通知されます。ユーザーが特定のリモート・フォームの使用を禁止されていることをローカル印刷サービスが*知らない* 場合、ユーザーがリモート・システムから「**permission denied**」メッセージを受け取るまでに遅延が生じます。

フォーム・アクセス・リストの定義

許可リストと拒否リストを使用して、フォームへのユーザー・アクセスを制御できます。

許可リストに名前を追加し、拒否リストから除去するには、次のコマンドを実行します。

```
lpforms -f form-name -u allow:user-list
```

```
lpforms -f form-name -u deny:user-list
```

拒否リストに名前を追加し、許可リストから除去するには、次のコマンドを実行します。

user-list は、コンマ区切りまたはスペース区切りのユーザー名のリストです。名前の区切りにスペースを使用する場合は、リスト全体 (**allow:** または **deny:** を含み、**-u** を含まない) を引用符で囲んでください。リストのそれぞれの項目には、システム名を含めることができます。

allow:all を指定すると、すべてのユーザーが許可されます。**deny:all** を指定すると、すべてのユーザーが拒否されます。

許可リストまたは拒否リストにユーザー名を追加しない場合、印刷サービスはすべてのユーザーがフォームを使用できると想定します。

フォームのマウント

事前印刷フォームを必要とするファイルの印刷を印刷サービスが開始する前に、ユーザーはプリンターにフォームをマウントする必要があります。

フォームにアラートが設定されている場合は、フォームのマウントを待機している印刷要求が一定の数だけキューに入ると、アラートが発行されます (アラートについては、250 ページの『フォームのマウントとフォント・カートリッジに関するアラート』を参照してください)。フォームをマウントするには、プリンターにフォームを装てんしてから、マウント済みであることを印刷サービスに通知する必要があります。まずプリンターを使用不可にすることをお勧めします。

プリンターにフォームをマウントするまで、そのフォームを必要としない印刷要求のみがプリンターに送られます。

フォームがマウントされたことを印刷サービスに通知するには、次の手順で行います。

1. プリンターを使用不可にします。
2. プリンターに新規フォームを装てんします。

3. フォームをマウントするには、次のコマンドを実行します。

   ```
   /usr/sbin/lpadmin -p printer-name -M -f form-name -a -o filebreak
   ```

4. プリンターを再度使用可能にします。

フォームに位置合わせパターンが登録されている場合は、フォームをマウントした後、位置合わせパターンが正しく見えるようにプリンターの調整が完了するまで、このパターンを繰り返し印刷するように要求できます。

-o filebreak オプションは、位置合わせパターン (存在する場合) の各コピーの後に用紙送りを追加するように印刷サービスに指示します。位置合わせパターンの各コピーを印刷するには、その前に Enter キーを押す必要があります。

用紙送りに使用される実際の制御シーケンスは、関連したプリンターによって異なり、**terminfo** データベースから取得されます。位置合わせパターンに用紙送りが既に組み込まれている場合は、**-o filebreak** オプションを省略します。

フォームの検討

印刷サービスに対してフォームを定義した後、確認したい情報のタイプに応じて、2 つのコマンドのどちらかを使用してフォームを検討できます。

lpforms コマンドは、フォームの属性を表示します。(**lpforms** コマンドによって生成される表示内容は、入力として使用できます。この情報は、後で参照するためにファイルに保管できます。) **lpstat** コマンドは、フォームの現在の状況を表示します。

注: フォームを印刷サービスから誤って削除した場合は、ファイルに取り込まれたフォーム定義を使用して、フォームを後から再定義できます。

フォームの状況を表示するには、次のコマンドを実行します。

```
lpstat -f form-name -l
```

出力の要約を受け取るには、**-l** を省略します。

次に例を示す詳細形式の出力は、**lpforms -l** の出力と同様です。

```
Page length: scaled-number
Page width: scaled-number
Number of pages: integer
Line pitch: scaled-number
Character pitch: scaled-number
Character set choice: character-set[,mandatory]
Ribbon color: ribbon-color
Comment:
comment
Alignment pattern: [content-type]
content
```

機密情報を含む可能性があるコンテンツを保護するために、**lpstat** コマンドを使用した場合は位置合わせパターンは表示されません。

印刷フィルター

単純なプリンター障害検出を行うために、印刷サービスにはデフォルトのフィルターが組み込まれています。このフィルターは、ファイルの変換や、特殊モードの処理は行いません。ただし、要件によってはこのフィルターで十分です。

フィルターは、次の目的に使用できるプログラムです。

- 特定のプリンター上で正しく印刷できるように、ユーザー・ファイルのデータ・フォーマットを別のものに変換する。
- ユーザーが **lp** コマンドの **-y** オプションによって要求する特殊な印刷モードを処理する (両面印刷、横長印刷、ドラフト印刷、高品質印刷など)。
- プリンターの障害を検出し、印刷サービスにその障害を通知して、印刷サービスからユーザーにアラートを出すことができるようにする。

フィルターによっては、これら 3 つのタスクをすべて実行できないものもあります。これらの役割はプリンターによって異なるため、印刷サービスはこれらの役割を別個に実装できるように設計されています。この分離により、ユーザーやプリンター製造メーカー (またはその他のソース) は印刷サービスを変更せずにフィルターを提供できます。

> **注:** フィルターを追加、変更、または除去すると、まだキューに入っている印刷要求が取り消される可能性があります。これは、フィルターの変更によって影響を受ける印刷要求を判別するために、まだキューに入っているすべての印刷要求を印刷サービスが評価するからです。フィルターが除去または変更されたために印刷可能でなくなった要求は、取り消されます (その要求を実行依頼したユーザーには通知が送信されます)。また、フィルターが変更されると、新規の印刷要求や変更された印刷要求への応答が遅れる可能性があります。これは、まだにキューに入っているそれぞれの印刷要求ごとに、多数の特性を評価する必要があるからです。フィルター処理を必要とする要求が多数ある場合には、この遅延が顕著になる可能性があります。
>
> この影響が生じる可能性があるので、フィルターの変更は印刷サービスがアイドル状態にあるときに行ってください。

ファイル変換

それぞれのプリンター (ローカルまたはリモート) ごとに、印刷可能なファイル・コンテンツ・タイプを指定できます。

ユーザーが印刷するファイルをプリンターに実行依頼し、そのコンテンツ・タイプを指定すると、印刷サービスはそのコンテンツ・タイプのファイルを処理できるプリンターを検索します。アプリケーションの多くはさまざまなプリンター用のファイルを生成できるので、たいていはこの機能で十分です。ただし、アプリケーションによっては、ご使用のプリンター上で印刷できないファイルを生成することがあります。

こうしたファイルをご使用のプリンターが処理できるタイプに変換するフィルターを定義し、作成することによって、印刷サービスのサポートするアプリケーションを増やすことができます。 (印刷サービスは、さまざまなタイプのファイルを PostScript に変換するためのフィルターをいくつか提供しています。) システムに追加するそれぞれのフィルターごとに、受け入れ可能な 1 つ以上の入力のタイプと、生成可能な出力のタイプ (通常はただ 1 つ) を指定する必要があります。

プリンターが処理できないファイル・コンテンツ・タイプをユーザーが指定すると (**lp -T** を実行して)、印刷サービスはファイルを受け入れ可能なタイプに変換できるフィルターの検索を試みます。印刷されるファイルがフィルターを通過した場合、印刷サービスはそのフィルターの出力タイプを、プリンター・タイプまたは別のフィルターの入力タイプと突き合わせます。印刷サービスは、このようにして出力タイプと入力タイプの突き合わせを続け、そのファイルを受け入れるプリンターに到達するまで、ファイルを一連のフィルターに通します。

例: HP DeskJet 500

この例では、ユーザー Chris がスプレッドシート・プログラムを実行し、スプレッドシートのコピーを含むファイルを生成しました。Chris は、印刷サービスを使用してこのファイルを印刷しようとしています。システムには HP DeskJet 500 プリンターのみがあります。そのスプレッドシート・アプリケーションは数種類のプリンター用の出力を生成する機能を備えており、Chris は HP DeskJet 500 によって処理できる出力を要求する必要があることを理解しています。 Chris がファイルの印刷を実行依頼すると、印刷サービスはいずれかのプリンターのキューにそのファイルを入れます。フィルターは必要ありません。

例: Tektronix 4014 の出力

この例では、ユーザー Marty が Tektronix 4014 端末装置に表示できるグラフィック・イメージを作成しました。Marty はこのイメージを印刷しようとしていますが、プリンターはすべて PostScript プリンターです。システムには、Tektronix タイプのファイルを PostScript に変換する **posttek** というフィルターが用意されています。プリンター・タイプを PostScript に設定することによって、印刷サービスは **posttek** フィルターを使用して Marty の出力を印刷前に変換できることを認識します。

特殊な印刷モード

フィルターは特殊な印刷モードを処理できます。

フィルター・テーブルに追加したそれぞれのフィルターを登録して、特殊モードなど印刷のさまざまな性質を処理できます。例を次に示します。

- 特殊モード
- プリンター・タイプ
- 文字ピッチ
- 行ピッチ
- ページ長
- ページ幅
- 印刷するページ
- 文字セット
- フォーム名
- 印刷部数

特殊モードと、特定ページの印刷を処理するには、フィルターが必要です。その他については、印刷サービスがデフォルトで処理を行います。ただし、その他の処理にもフィルターを使用すると効率が良くなる場合があります。また、フィルターが他の役割を適切に果たすために、フィルターに対してこれらの性質のいくつかを指定する必要が生じる可能性があります。例えばフィルターは、印刷されるページに収まるようにファイル内のページを分割する場合に、ページ・サイズと印刷スペーシングを認識している必要があります。別の例として、プリンターによっては、印刷サービスよりも効率的に複数部印刷を処理できるものがあります。このため、プリンターを制御できるフィルターが印刷部数に関する情報を使用すれば、印刷サービスによる複数部印刷のデフォルト処理をスキップできます。

プリンター障害の検出

ファイルの変換と特殊な印刷モードの処理はプリンターによって異なる役割ですが、プリンター障害の検出もこれとまったく同様です。

印刷サービスは一般的な障害の検出を行い、ほとんどのプリンターに対して正しく障害を検出できます。ただし、印刷サービスが検出できる障害の範囲は限られています。印刷サービスは、ハングアップ (プリンターがオンラインであることを示すシグナルである、キャリアの損失)、および印刷の過大な遅延 (データ・

フローをシャットオフする XOFF フロー制御文字を受信し、それに対応してフローを再度オンにする XON がない場合) を検査できます。ただし、印刷サービスは障害の原因を判別できないので、何を調べるべきかユーザーに通知できません。

適切に設計されたフィルターは、障害に幅広く対応できます。プリンターによっては、障害の理由を説明するメッセージをホストに送信できます。また、キャリアの損失やデータ・フローのシャットオフ以外のシグナルを使用して、障害を特定するものもあります。フィルターの便利な点は、より多くの障害が検出され、障害について通常よりも多くの情報が得られることです。

フィルターは、プリンター障害が解消するまで待ってから印刷を再開できます。このサービスにより、中断された印刷要求をすべて再印刷する必要がなくなるので、障害が発生したときの印刷が効率的になります。実フィルターは、プリンターによって使用されている制御シーケンスを認識し、この実フィルターのみがファイル内のページの切れ目を*理解* できます。このため、こうしたフィルターのみが、ファイル内で印刷を再開する場所を見つけることができます。

印刷サービスに備わっているインターフェースを使用して、フィルターはユーザーに障害情報を送信でき、可能な場合は印刷を再開できます。アラート・メカニズム (248 ページの『プリンター障害アラート』を参照) は、印刷サービスによって処理されます。このメカニズムにより、フィルターを管理するインターフェース・プログラムがすべてのエラー・メッセージをフィルターから取得し、アラート・メッセージに挿入してユーザーに送信できるようにします。障害が解消された後に印刷を自動的に再開するようにプリンター構成を設定した場合は、印刷が中断した個所が見つかるように、インターフェース・プログラムはフィルターをアクティブに保ちます。

フィルターに適したプログラム

troff、**nroff**、あるいは同様なワード・プロセッシング・プログラムをフィルターとして使用すると便利なように見えます。しかし、**troff** と **nroff** の両プログラムには、ソース・ファイル内で他のファイル (*インクルード・ファイル* と呼ばれる) の参照を可能にする機能があります。印刷サービスはインクルード・ファイルを認識しません。このため、ソース・ファイルが印刷のためにキューに入っているときに、ソース・ファイルによって参照されるファイルはキューに入りません。このため、**troff** プログラムや **nroff** プログラムはインクルード・ファイルにアクセスできず、失敗する可能性があります。その他のプログラムにも、フィルターとしての使用の制約になる同様な機能が備わっていることがあります。

フィルターとして使用するプログラムの評価について、次にいくつかのガイドラインを示します。

- 標準入力からデータを読み取り、標準出力にデータを書き込むことができるプログラムのみ、フィルターとして使用できます。
- そのプログラムによる処理を必要とする、ユーザーが印刷を実行依頼するファイルの種類を検討します。独立したプログラムが適しています (つまり、他のファイルを参照する必要がない)。

  ユーザーが印刷を実行依頼するファイル以外に、そのプログラムが必要とするファイルがあるかどうか調べます。該当する場合は、フィルターを使用するユーザーのディレクトリーにそのファイルが入っているか、フィルターを使用する権限があるすべてのユーザーに対してそのファイルが読み取り可能であることが必要です。この読み取りに関する前提条件が必要である理由は、印刷要求を実行依頼したユーザーのユーザー ID とグループ ID を使用してフィルターが実行されるからです。
- 印刷対象として実行依頼されるファイル内で参照ファイルの使用が許されている場合、またはユーザーが実行依頼するもの以外のファイルをプログラムが必要とする場合には、プログラムがこれらの追加ファイルにアクセスできず、失敗する可能性があります。そのプログラムをフィルターとしての考慮事項のもとで使用するのではなく、ファイルの印刷を実行依頼する前にそのプログラムを実行するようにユーザーに依頼してください。

常に絶対パス名によって指定される参照ファイルは、使用できる場合もあります が、これはフィルターをローカル印刷要求に使用する場合に限られます。それでも、リモート・マシンからご使用のマシンに対して実行依頼された印刷要求に対してフィルターを使用すると、参照ファイルがリモート・マシンのみに存在する場合に、フィルターは失敗する可能性があります。

印刷フィルターの定義

新規フィルターを追加する際には、その使用特性を定義する必要があります。

フィルターの使用特性を定義するには、次のフィルター特性の値を指定する引数を付けて **lpfilter** コマンドを発行します。

- フィルターの名前 (つまり、コマンド名)
- 受け入れる入力のタイプ
- 生成する出力のタイプ
- ジョブを送信する先のプリンターのタイプ
- ジョブを送信する先の具体的なプリンターの名前
- フィルターのタイプ (**高速** フィルターまたは**低速** フィルターのどちらか)
- オプション

詳細については、 234 ページの『印刷サービスへのフィルターの追加』を参照してください。

フィルター定義はファイルに保管するかコマンド・ラインに直接入力でき、フォーマットは次のとおりです。

```
Command: command-pathname [options]
Input types: input-type-list
Output types: output-type-list
Printer types: printer-type-list
Printers: printer-list
Filter type: fast or slow
Options: template-list
```

情報は任意の順序で指定できます。一部の情報は必須ではありません。次に示す項目に値を指定しない場合は、デフォルト値が割り当てられます。

lpfilter 引数	**デフォルト**
Command:	(デフォルトなし)
Input types:	any
Output types:	any
Printer types:	any
Printers:	any
Filter type:	slow
Options:	(デフォルトなし)

デフォルト値は柔軟なフィルターを定義するので、少なくとも入力タイプと出力タイプを指定してください。リストを入力する際には、リスト内の項目をブランクまたはコンマで区切ることができます (テンプレート・リスト でない限り)。テンプレート・リスト の項目はコンマで区切る必要があります。

これらの特性のそれぞれについて、次に説明します。

- **Command:** フィルター・プログラムの絶対パス。

  プログラムが常に必要とする固定のオプションがある場合は、ここに組み込みます。
- **Input types:** フィルターが処理できるファイル・コンテンツ・タイプのリスト。

印刷サービスは、フィルターが受け入れることのできる入力タイプの数に限度を設けていませんが、ほとんどのフィルターはただ 1 つのタイプを受け入れます。複数のファイル・タイプに十分な類似性があれば、フィルターがこれらのファイル・タイプを処理できます。最大 14 文字の英数字とダッシュ (下線は不可) からなる任意の名前を使用できます。印刷サービスはこれらの名前を使用してフィルターとファイル・タイプを突き合わせるので、一貫性のある命名規則を順守してください。例えば、複数のフィルターが同じ入力タイプを受け入れることができる場合は、それぞれのフィルターに入力タイプを指定する際に、その入力タイプに同じ名前を使用します。ファイルの印刷を実行依頼する際にそのファイルのタイプを指定する方法が分かるように、ユーザーにその名前を知らせてください。

- **Output types:** フィルターが出力として生成できるファイル・タイプのリスト。

  それぞれの入力タイプごとに、フィルターは単一の出力タイプを生成します。ただし、出力タイプはジョブごとに異なる場合があります。出力タイプの名前は、14 文字の英数字とダッシュに制限されています。

  これらの名前は、システム上に存在するプリンターのタイプと一致しているか、他のフィルターが処理する入力タイプと一致している必要があります。印刷サービスは、異なるフィルターによる複数のパスがファイルの変換に必要であると判断すると、複数のフィルターをシェル・パイプラインにグループ化します。ユーザーが印刷する可能性があるすべての種類のファイルを受け入れ (入力タイプとして)、これらのファイルをプリンターが処理できるタイプに直接変換するフィルターのセットを判別してください。

- **Printer types:** フィルターがファイルを変換できるプリンター・タイプのリスト。

  ほとんどのフィルターの場合、このリストは出力タイプのリストと同一です。

  例えば、初期設定時には単一のタイプを指定したプリンター (235 ページの『プリンター・タイプ』を参照) が、複数の異なるタイプのファイルを認識できる場合があります。基本的にこのプリンターには、さまざまなタイプをプリンターが処理できる 1 つのタイプに変換する内部フィルターがあります。したがって、フィルターは、プリンターが処理できるファイル・タイプ に一致する、複数の出力タイプのいずれかを生成できます。フィルターに、そのプリンター・タイプを処理することを示すラベルを付けてください。

  別の例として、同じタイプのファイルを受け入れるものとしてリストされている、2 つの異なるプリンター・モデルがあるとします。ただし、製造時のわずかな差異のために、一方のプリンターから得られる結果には違いがあります。これらのプリンターに、異なるプリンター・タイプのものであることを示すラベルを付けます。例えば A と B のラベルを付け、B が違いのあるものとします。タイプ B のプリンターによって生じる違いを考慮してファイルを調整するフィルターを作成します。このフィルターはこれらのプリンター・タイプのみに必要なので、タイプ B プリンターのみを処理対象としてリストします。

  ほとんどのプリンターとフィルターに対して、フィルター定義のこの部分はブランクのままにすることができます。

- **Printers:** プリンターによっては、フィルターに対しては正しいタイプのものであっても、その他の面でフィルターが生成する出力には適していない場合があります。

  例えば、1 つのプリンターを高速ターンアラウンド専用にしたいとします。プリンターがフィルター処理を行わずに処理できるファイルのみが、そのプリンターに送信されます。その他の同一タイプのプリンターは、印刷の前に高度なフィルター処理を必要とするファイルに使用できるようにします。この場合、フィルターには、後者のプリンターのグループのみを対象とすることを示すラベルを付けます。

ほとんどの場合、フィルターはその出力を受け入れるすべてのプリンターに対して機能するので、通常はフィルター定義のこの部分はスキップできます。

- **Filter type:** 印刷サービスは、*高速* フィルターと*低速* フィルターを認識します。

  高速フィルターは、ファイルの印刷の準備にかかるオーバーヘッドが小さく、実行時にプリンターにアクセスする必要があるために、高速と呼ばれます。プリンター障害を検出するためのフィルターは、高速フィルターにする必要があります。フィルター・オプションとして **PRINTER** キーワードを使用するフィルターは、高速フィルターとしてインストールする必要があります。

  低速フィルターは、ファイルの準備にかかるオーバーヘッドが大きく、プリンターへのアクセスを必要としないフィルターです。印刷サービスは低速フィルターをバックグラウンドで実行するので、プリンターを拘束することはありません。このため、低速フィルターを必要としないファイルは先の処理に進むことができます。他のファイルを同時に印刷できる場合は、低速フィルターがファイルを処理している間、プリンターがアイドル状態のままになることはありません。

  モードによって呼び出される (**-y** オプションを使用して) 低速フィルターは、印刷要求を発行したコンピューター上で実行する必要があります。印刷サービスは、モードの値をサーバー・マシンに渡すことができません。ただし、ファイル・コンテンツ・タイプ (**lp** コマンドの **-T** オプションの後に指定される) を、サーバー・マシンのコンテンツ・タイプと突き合わせることができます。このため、サーバー・マシン上で特殊モードを活動化するには、印刷サービスが入力タイプと出力タイプを突き合わせることができるように、コンテンツ・タイプを指定する必要があります。
- **Options:** オプションは、さまざまなタイプの情報をフィルター・コマンドのコマンド・ライン引数に変換する方法を指定します。

  この情報には、ユーザーからの仕様 (印刷要求による指定)、プリンター定義、要求の処理に使用されるフィルターが実装する仕様などがあります。

  情報のソースは 13 種類あり、これらはそれぞれキーワード によって表現されます。それぞれのオプションはテンプレート 内で定義されます。テンプレートは、次のフォーマットのステートメントです。

  ```
  keyword pattern=replacement
  ```

  このタイプのステートメントは、印刷サービスによって次のように解釈されます。「*keyword* によって参照される情報に、*pattern* と一致する値がある場合は、*replacement* 文字列を取り出し、含まれているアスタリスクをすべて指定された *pattern* に置き換えるか、含まれている正規表現を展開して、結果をコマンド・ラインに追加する。」

  フィルター定義に指定されるオプションは、これら 13 種類のキーワードを含まないことも、すべて含むことも、任意のサブセットを含むこともできます。また、完全なフィルター定義のために複数の定義が必要な場合は、単一のキーワードを複数回定義できます。 231 ページの『テンプレートを使用したオプションの定義』を参照してください。

フィルターの特性を定義するために十分な情報を収集したら、そのデータを引数として使用して、**lpfilter** コマンドを実行する準備が整います。数多くの引数が存在し、その一部は複数回入力する必要があるので (異なる値を指定して)、まずこの情報を別のファイルに記録し、必要に応じてそのファイルを編集してください。その後、そのファイルを **lpfilter** コマンドへの入力として使用できるので、それぞれの情報を個別に入力しなくて済みます。

テンプレートを使用したオプションの定義:

テンプレートは、フィルター定義内のステートメントで、フィルターの特性の 1 つの値に応じてフィルター・コマンドに渡すオプションを定義します。

フィルター定義には複数のテンプレートを組み込むことができます。複数のテンプレートは単一の行に入力してコンマで区切ることも、**Options:** 接頭部を先頭に付けて別々の行に入力することもできます。

テンプレートのフォーマットは次のとおりです。

keyword pattern=replacement

このタイプのステートメントは、印刷サービスによって次のように解釈されます。「*keyword* によって参照される情報に、*pattern* と一致する値がある場合は、*replacement* 文字列を取り出し、含まれているアスタリスクをすべて指定された *pattern* に置き換えるか、含まれている正規表現を展開して、結果をコマンド・ラインに追加する。」

一例として、印刷サービス・スケジューラーが、次の基準に基づいて印刷要求をフィルターに割り当てるようにしたいとします。

- フィルターによって生成される **OUTPUT** のタイプが **impress** ならば、**-I** オプションをフィルターに渡す。
- フィルターによって生成される **OUTPUT** のタイプが **postscript** ならば、**-P** オプションをフィルターに渡す。

これらの基準を指定するには、**lpfilter** コマンドのオプションとして次のテンプレートを指定します。

```
Options: OUTPUT impress=-I, OUTPUT postscript=-P
```

Options: 行が長すぎる場合は、次のようにして、それぞれのテンプレートを別々の行に入力します。

```
"Options: OUTPUT impress=-I"
"Options: OUTPUT postscript=-P"
```

どちらのテンプレート内でも、*keyword* は **OUTPUT** です。最初のテンプレートでは、*pattern* の値は **impress** で、*replacement* の値は **-I** です。2 番目のテンプレートでは、*pattern* の値は **postscript** で、*replacement* の値は **-P** です。

キーワードの定義と例:

キーワードを使用して、フィルター定義にオプションを定義できます。

フィルター定義のオプションを定義するために、次のキーワード を使用できます。

特性	キーワード	**可能な**パターン	**例**
コンテンツ・タイプ (入力)	INPUT	*コンテンツ・タイプ*	troff
コンテンツ・タイプ (出力)	OUTPUT	*コンテンツ・タイプ*	postscript
プリンター・タイプ	TERM	*プリンター・タイプ*	att495
プリンター名	PRINTER	*プリンター名*	lp1
文字ピッチ	CPI	*スケール付き 10 進*	10
行ピッチ	LPI	*スケール付き 10 進*	6
ページ長	LENGTH	*スケール付き 10 進*	66
ページ幅	WIDTH	*スケール付き 10 進*	80
印刷するページ	PAGES	*ページ・リスト*	1-5,13-20
文字セット	CHARSET	*文字セット*	finnish

特性	キーワード	可能なパターン	例
フォーム名	FORM	フォーム名	invoice2
印刷部数	COPIES	整数	3
特殊モード	MODES	モード	landscape

それぞれのタイプのテンプレートに指定する値 (つまり、各 *keyword* の *pattern* 引数と *replacement* 引数) を調べるには、次のことを考慮してください。

- INPUT と OUTPUT の両テンプレートの値は、フィルターによる変換を必要とするファイル・タイプと、フィルターによって生成される出力タイプからそれぞれ得られます。これらのタイプはそれぞれ、フィルターに登録されます。
- TERM テンプレートの値は、プリンター・タイプです。
- PRINTER テンプレートの値は、最終出力を印刷するプリンターの名前です。
- CPI、LPI、LENGTH、および WIDTH の各テンプレートの値は、ユーザー要求、使用されるフォーム、またはプリンターのデフォルト値から得られます。
- PAGES テンプレートの値は、印刷されるページのリストです。通常、このリストはページ範囲のコンマ区切りリストです。それぞれの範囲は、ダッシュで区切られた 1 組みの数、または単独の番号です (例えば、**1-6,8,10** はページ 1 から 6、8、および 10 を表す)。ただし、印刷要求の **-P** オプションに指定された値はすべて、変更されずに渡されます。
- CHARSET テンプレートの値は、使用される文字セットの名前です。
- FORM テンプレートの値は、**lp** コマンドの **-f** オプションによって要求されるフォームの名前です。
- COPIES テンプレートの値は、ファイルの印刷部数を決定します。フィルターがこのテンプレートを使用する場合、印刷サービスは印刷されるフィルター対象ファイルの印刷部数を 1 に減らします。これは、実際にはこの単一コピーからフィルターによって複数コピーが作成されるからです。
- MODES テンプレートの値は、**lp** コマンド (印刷要求の実行依頼に使用されるコマンド) の **-y** オプションから得られます。ユーザーは複数の **-y** オプションを指定できるので、MODES テンプレートには複数の値が存在する可能性があります。これらの値は、ユーザーによる指定順に左から右に適用されます。

テンプレートの *replacement* 部分は、テンプレートの値をフィルター・プログラムに与える方法を示しています。これは通常はリテラル・オプションですが、値の入る場所を示すプレースホルダー * (アスタリスク) が含まれる場合もあります。 *pattern* と *replacement* には、**ed** コマンドの正規表現構文を使用して、ユーザー入力オプションからフィルター・オプションへの複雑な変換を行うこともできます。 **ed** コマンドの正規表現構文は、**\(. . \)** と **\n** の構造を含めてすべてサポートされています。 . . **\n** は、*replacement* にコピーする *pattern* の部分を抽出するために使用できる構造で、**&** は *pattern* 全体を *replacement* にコピーするために使用できます。

注: コマンドまたは等号 (=) が *pattern* または *replacement* に含まれている場合は、エスケープ文字のバックスラッシュ (\) をその前に付けて特殊な意味を無効にしてください。一部の正規表現には、このようにエスケープ文字を付ける必要があるコンマが含まれていることに注意してください。これらの文字の前にあるバックスラッシュは、*pattern* または *replacement* が使用されるときに除去されます。

col フィルターの例:

col フィルターを使用してユーザーの印刷要求を変更する例を示します。

既に次の定義を指定した **col** という名前のフィルターを追加してあるものとします。

```
Input types:     N37, Nlp, simple
Output types:    simple
Command:         /usr/bin/col
Options:         TERM 450 = -b, MODES expand = -x
Options:         INPUT simple = -p -f
```

注: Options 以外のフィルター特性に対して、複数の定義 (つまり、複数の行) を指定した場合は、最後の定義のみが印刷サービスによって使用されます。

lpfilter コマンドへの入力としてこの定義を入力し、印刷サービスにこの定義を登録 した後、ユーザーの印刷要求は次のように処理されます。

- ユーザーが次のコマンドを入力した場合

  ```
  lp -y expand report.dec10
  ```

  フィルター・コマンドは次の引数を指定して実行されます。

  ```
  /usr/bin/col -x -p -f
  ```

- ユーザーが次のコマンドを入力した場合

  ```
  lp -T N37 -y expand report.dec10
  ```

  フィルター・コマンドは次の引数を指定して実行されます。

  ```
  /usr/bin/col -x
  ```

  限定条件: デフォルト・プリンターがタイプ 450 のものではないこと。

- ユーザーが次のコマンドを入力した場合

  ```
  lp -y expand -T 450 report.dec10
  ```

  フィルター・コマンドは次の引数を指定して実行されます。

  ```
  /usr/bin/col -b -x
  ```

***dpost* フィルターの例:**

フィルター・プログラム /usr/lib/lp/postscript/dpost は、1 つの入力タイプ troff を受け取り、出力タイプ **postscript** を生成し、タイプ PS (PostScript を表す) の任意のプリンターに対して機能します。

ユーザーが縦長モードと横長モードの用紙の向きを要求するときに、それぞれ省略形の **port** と **land** を指定するだけで済むようにしたいとします。これらのオプションは印刷サービスに組み込みのものではないので、ユーザーは **lp** コマンドの **-y** オプションを使用してこれらのオプションを指定する必要があります。

フィルター定義は次のようなものです。

```
Input types: troff
Output types: postscript
Printer types: PS
Filter type: slow
Command: /usr/lib/lp/postscript/dpost
Options: LENGTH * = -l*
Options: MODES portrait = -op, MODES land = -ol
```

PostScript プリンター (タイプ **PS**) 上でタイプ troff のファイルの印刷を実行依頼し、横長の向きと 60 行のページ長を要求するユーザーは、次のコマンドを入力します。

```
lp -T troff -o length=60 -y land -d any
```

このフィルターは、ファイルを変換するために印刷サービスによって次のように呼び出されます。

```
/usr/lib/lp/postscript/dpost -l60 -ol -pl
```

オプション・テンプレートの例:

この例は、フォーム `-y group=number` の **MODES** オプションをフィルター・オプション **-n***number* に変換するためのテンプレートとして使用してください。

231 ページの『テンプレートを使用したオプションの定義』トピックの例に、次のオプション・テンプレートを追加します。詳細については、 231 ページの『テンプレートを使用したオプションの定義』を参照してください。

```
Options: MODES group\=\([1-9]\) = -n\1
```

これにより、ユーザーがコマンド `lp -y group=4` を指定した場合、**dpost** コマンドにはオプション **-n4** が組み込まれます。

その他の例については、次のコマンドを実行してください。

```
/usr/sbin/lpfilter -f filter -l
```

ここで、*filter* は出荷時インストール済みの PostScript フィルターの名前です。(PostScript フィルターのリストについては、 237 ページの『PostScript プリンター』を参照してください。)

印刷サービスへのフィルターの追加

フィルターを定義した後、システムにフィルターを追加できます。

フィルターを定義した後、システムにフィルターを追加するには、次のいずれかのコマンドを使用します。

```
/usr/sbin/lpfilter -f filter-name -F filename
/usr/sbin/lpfilter -f filter-name -
```

最初のコマンドは、ファイルからフィルター定義を取得し、2 番目のコマンドは、標準入力からフィルター定義を取得します。*filter-name* はユーザーが選択した任意の文字列で、最大 14 文字の英数字と下線です。

フィルターを変更する必要がある場合は、どちらか同じコマンドを再入力します。変更する必要がある項目の情報のみを提供する必要があります。新しい情報を指定しない項目はそのまま同じです。

フィルターの除去

印刷サービスは、定義できるフィルターの数に固定の限度を設けていません。

印刷サービスが余分な処理を行わずに済むように、適用されなくなったフィルターは除去してください。印刷サービスは、特定の状況で使用できるものを見つけるために、すべてのフィルターを検査する必要があります。

フィルターを除去するには、次のコマンドを実行します。

```
/usr/sbin/lpfilter -f filter-name -x
```

フィルターの検査

フィルター定義を印刷サービスに追加した後、**lpfilter** コマンドを実行してフィルターを検査できます。

lpfilter コマンドの出力は、入力として適したフォーマットで表示されるフィルター定義です。この出力をファイルに保管すれば、フィルターを印刷サービスから誤って削除した場合に、このファイルを使用してフィルターを後から再定義できます。

フィルターの定義を画面に表示するには、次のコマンドを入力します。

```
/usr/sbin/lpfilter -f filter-name -l
```

将来参照するためにフィルターの定義をファイルに取り込むには、次のコマンドを入力します。

```
/usr/sbin/lpfilter -f filter-name -l > filename
```

フィルターのデフォルトの復元

ソフトウェアの工場出荷時に、デフォルトのフィルターのセットが付属しています。フィルターを追加、変更、または削除する際に、これらのオリジナル・フィルターの一部を上書きまたは除去することがあります。

フィルターのデフォルト・セットの一部またはすべてを変更した後、オリジナルの形式に復元するには、次のコマンドを入力します。

```
/usr/sbin/lpfilter -f filter-name -i
```

filter-name は復元するフィルターの名前に置き換えるか、すべてのデフォルト・フィルターを復元する場合は **all** という語に置き換えます。

プリンター・タイプ

printer-type 属性は、**lpadmin** コマンドの **-T** *printer-type* オプションを使用して定義します。

プリンター・タイプは、プリンターの総称名です。通常、これは製造メーカーの名前から得られます。例えば、ACME Computer 356 ドット・マトリックス・プリンターのタイプは **356** です。印刷ソフトウェアはタイプに基づいて **terminfo** データベースからプリンターに関する情報を抽出するので、それぞれのプリンターにタイプ を割り当てることは重要です。この情報にはプリンター機能のリストが含まれており、このリストによってユーザーが印刷サービスに提供する構成情報が検査されます。(構成しているプリンターのタイプにある既知の機能と突き合わせてユーザー提供の情報を検査することによって、印刷サービスはユーザーが不適切な情報を提供した場合にそれを検出できます。)**terminfo** データベースは、ファイルを印刷する前に特定のプリンターを初期化するために必要な制御データも指定します。

プリンター・タイプを指定する必要はありませんが、指定することをお勧めします。印刷サービスを介して使用可能なプリンターをタイプ別に分類することによって、ユーザーがシステムをより便利に使用できるようになります。

プリンター・タイプのリストを指定する場合は、名前をコンマで区切ってください。プリンター・タイプを定義しない場合は、デフォルトの **unknown** が使用されます。

プリンターが複数の種類のプリンターをエミュレートできる場合は、プリンターに複数のタイプを割り当てることができます。例えば、プリンターが IBM プロプリンター XL、Epson FX86e、および HP LaserJet II をエミュレートできる場合は、terminfo データベースにこれらのタイプをそれぞれ **593ibm**、**593eps**、**593hp** として指定します。複数のプリンター・タイプを指定すると、印刷サービスはそれぞれの印刷要求ごとに、これらのいずれかを状況に応じて使用します。

次の例では、**lpadmin** コマンドを使用して、タイプ **593ibm** をプリンター **laser** に関連付ける方法を示します。

```
/usr/sbin/lpadmin -p laser -T 593ibm
```

注: 複数のプリンター・タイプを指定する場合は、コンテンツ・タイプとして **simple** を指定する必要があります。

コンテンツ・タイプ

ほとんどのプリンターは、プリンター・タイプと同タイプのファイル、および **simple** タイプのファイル (ASCII ファイル) を印刷できます。

content-type 属性は、**lpadmin** コマンドの **-I** *content-type-list* オプションを使用して定義します。ほとんどのプリンターは、プリンター・タイプと同タイプ (プリンター・タイプが定義されている場合)、およびすべてのプリンターのデフォルト・コンテンツ・タイプであるタイプ **simple** (ASCII ファイルを意味する) の 2 種類のファイルを印刷できます。

コンテンツ・タイプ **simple** のファイルは、印刷可能 ASCII 文字と次に示す制御文字の 2 種類の文字のみを含んでいることが想定されます。

項目	説明
バックスペース	行の先頭以外で、キャリッジを 1 スペース分戻します。
タブ	キャリッジを次のタブ停止位置まで移動します。デフォルトでは、ほとんどのプリンター上で 8 列ずつのスペースを空けて停止位置が設定されています。
改行	キャリッジを次の行の先頭に移動します (一部のプリンターには特殊なポート設定が必要になる場合があります。251 ページの『プリンター・ポート特性』を参照してください)。
用紙送り	キャリッジを次のページの先頭に移動します。
復帰	キャリッジを同じ行の先頭に移動します (一部のプリンターでは失敗する場合があります)。

キャリッジ という言葉は最新のレーザー・プリンターには使われなくなっていますが、これらのプリンターはキャリッジが行うものと似たアクションを行います。プリンターが、**simple** を含む数種類のファイルを処理できる場合は、コンテンツ・タイプ・リストに **simple** を明示的に含める必要があります。プリンターがタイプ **simple** のファイルを受け入れないようにする場合は、**lpadmin** のコマンド・ラインにブランクの *content-type-list* (**-I ""**) を指定します。ただし、一部のプリンターは、複数の異なるタイプのファイルを受け入れ、正しく印刷できます。この種のプリンターを追加する際には、その新規プリンターが受け入れるコンテンツ・タイプの名前をリストに追加して指定します。(デフォルトでは、リストに含まれているタイプは **simple** の 1 つのみです。) リモート・プリンターを追加する場合は、そのプリンターが存在するシステムの管理者によって設定された、そのプリンターのコンテンツ・タイプをリストします。

content-type-list は、コンマ区切りまたはスペース区切りの名前のリストです。名前の区切りにスペースを使用する場合は、リスト全体 (**-I** を除く) を引用符で囲んでください。

よく使われるコンテンツ・タイプ:

コンテンツ・タイプはプリンター・タイプ名と似ていますが、プリンターを使用するユーザーに分かりやすい名前を選択できます。

コンテンツ・タイプ名は 14 文字を超えてはならず、文字、数字、および下線のみを含むことができます。次の表では、受け入れられるコンテンツ・タイプをいくつか示し、説明します。

注: 名前 **simple** と **any** は、特定の意味を持つものとして印刷サービスに認識されます。これらの名前は常に正しく使用してください。 **terminfo** という名前も予約済みで、すべての タイプのプリンターを参照するものとして使用されます。

よく使われるコンテンツ・タイプ

タイプ	説明
cif	BSD **cifpbt** の出力
daisy	Diablo 630 (*daisy-wheel*) プリンター用のファイルの印刷
dmd	端末装置からのビットマップ表示内容の印刷
fortran	ASA 紙送り制御フォーマット
otroff	BSD または System V より前の **troff** (古い **troff**) によって生成される CAT 組版命令
pcl	HP LaserJet ネイティブ出力フォーマット
plot	Tektronix ディスプレイおよびデバイス用のプロット命令
postscript	PostScript 言語
raster	Varian ラスター装置用のラスター・ビットマップ・フォーマット
simple	ASCII ファイル
tek4014	Tektronix 4014 デバイス用にフォーマット設定されたファイルの印刷
tex	DVI フォーマットのファイル
troff	**troff** からのデバイス独立の出力

lp コマンドの **-d any** オプションによってプリンターを指定して、ファイルの印刷が印刷サービスに対して実行依頼された場合、印刷サービスはそのジョブを処理できるプリンターを検索します。印刷サービスは、コンテンツ・タイプ名またはプリンター・タイプ名を使用して、適切なプリンターを識別できます。このため、ファイルの印刷を実行依頼する際には、どちらかの名前を指定できます (または名前を指定しない)。複数の異なるタイプのプリンターが同じコンテンツ・タイプを印刷できる場合は、これらのプリンターを追加する際に同じコンテンツ・タイプ名を使用します。このようにすれば、プリンターを使用するユーザーが、印刷の出力先に関係なく同じ名前を使用して印刷したいファイルのタイプを指定できるので便利です。

ほとんどの製造メーカーが、単純な ASCII ファイルを受け入れるプリンターを製造しています。これらのプリンターのタイプは異なっていても (したがって、異なる初期化制御シーケンスを使用していても)、同じタイプのファイルを処理できることがあります。こうしたファイル・タイプを **simple** と呼んでいます。一部の製造メーカーは、ANSI X3.64 によって定義されたエスケープ・シーケンスを受け入れるプリンターを製造しています。ただし、これらのプリンターがすべての ANSI 機能をサポートしているとは限らず、異なる機能のセットをサポートしている場合があります。これらのプリンターは、異なるコンテンツ・タイプ名を割り当てることによって区別できます。

デフォルトのコンテンツ・タイプ:

それぞれのプリンターごとにコンテンツ・タイプをリストすることが望ましい場合がありますが、常に必要ではありません。

プリンターのコンテンツ・タイプをリストしない場合は、プリンター・タイプがそのプリンターの処理できるコンテンツ・タイプの名前として使用されます。プリンター・タイプを指定しない場合、印刷サービスはプリンターがコンテンツ・タイプ **simple** のファイルのみを印刷できるものと想定します。正しいプリンターを明示的に指定するようにユーザーに要求していて、印刷の実行依頼前にファイルがプリンターのために正しく準備されていれば、このデフォルトで十分です。

PostScript プリンター

PostScript は汎用プログラミング言語の 1 つで、この言語を使用してページ上でのテキストとグラフィックス両方の外観を指定できます。

PostScript プリンターには、PostScript 言語ファイルを処理するためのインタープリターを実行するコンピューターが装備されています。 PostScript プリンターがファイルを受け取ると、そのファイルをインタープリターによって実行してから印刷します。製造メーカーが特別な機能を提供していない限り、PostScript プリンターに対して実行依頼されるファイルは PostScript 言語で作成されている必要があります。

テキストとグラフィックスを管理し、組み合わせる優れた機能を備えているほかに、印刷出力をサポートする主要アプリケーションのほとんどが PostScript をサポートしています。グラフィックス・オペレーターによって幾何学的図形を容易に作成でき、これらの図形は任意の方向で配置および拡大縮小できます。テキスト機能を使用して複数の異なるフォントを指定し、任意の位置、サイズ、または向きを指定してページに配置できます。テキストはグラフィックスとして扱われるので、テキストとグラフィックスを容易に組み合わせることができます。さらに、この言語は解像度とデバイスから独立しているので、低解像度デバイスでドラフト・コピーを校正した後、別の高解像度デバイスで最終版を印刷できます。

ワード・プロセッサーや出版ソフトウェアなど、PostScript をサポートするアプリケーションは、ユーザーによる介入を必要とせずに PostScript 言語の文書を作成します。このため、PostScript の機能を利用するためにこの言語の詳細な知識は必要ありません。ただし、一部のアプリケーションや特殊な端末装置が生成する標準ファイルは、この言語で記述されていないため、PostScript プリンターでは印刷できません。印刷サービスは、これらのファイルの多くを PostScript に変換するオプションのフィルターを提供しているので、ユーザーは PostScript を利用しながら **troff** などの標準アプリケーションも引き続き使用できます。

Retail Type 1 フォントをインストールして、デスクトップ上で実行されるアプリケーションに使用できます。アプリケーションがこれらのフォントを使用する PostScript 出力を生成する場合には、これらのフォントを PostScript プリンターにダウンロードできます。**lp** コマンドは、**download** フィルターを使用してこの処理を自動的に行います。

詳しくは、「*Commands Reference, Volume 1*」の **cancel** コマンドを参照してください。

詳しくは、「*Commands Reference, Volume 2*」の **dslpaccept**、**dslpaccess**、**dslpadmin**、**dslpdisable**、**dslpenable**、**dslpreject**、および **dslpsearch** の各コマンドを参照してください。

詳しくは、「*Commands Reference, Volume 3*」の **lp**、**lpstat**、**mkprtldap**、および **mksecldap** の各コマンドを参照してください。

PostScript プリンターの使用

PostScript プリンターとフィルターがインストールされている場合、印刷サービスは PostScript ファイルを他のファイルと同様に管理します。

`psfile` が PostScript 文書を含むファイルで、**psprinter** が PostScript プリンターとして印刷サービスに定義されている場合に、次のコマンドを実行するとします。

```
lp -d psprinter -T PS psfile
```

このコマンドは、印刷要求をスケジュールに入れ、PostScript プリンターへの要求の送信を管理します。

PostScript 以外の印刷要求

PostScript は言語であり、PostScript プリンターはその言語で作成された印刷要求を予期するので、一部のアプリケーションが生成する標準印刷要求を PostScript プリンターが解釈できない場合があります。

次に、一部の PostScript プリンターが解釈できない印刷要求の例を示します。

PostScript 以外のコンテンツ・タイプ

コンテンツ・タイプ	印刷要求のタイプ
simple	ASCII (シンプル) テキスト・ファイルの印刷
troff	**troff** コマンドからの出力の印刷
daisy	Diablo 630 (*daisy-wheel*) プリンター用のファイルの印刷
dmd	端末装置からのビットマップ表示内容の印刷
tek4014	Tektronix 4014 デバイス用にフォーマット設定されたファイルの印刷
plot	プロット・フォーマットのファイルの印刷

これらのフォーマットの印刷要求を PostScript 言語に変換するフィルターが、印刷サービスに付属しています。例えば、ASCII テキストを含むファイルを PostScript コードに変換する場合、フィルターはそのテキストを取得してそれを基にプログラムを作成し、印刷パラメーター、例えばフォントやページ上のテキストのレイアウトなどを指定します。

PostScript フィルターのインストール後、ユーザーが **-T** オプションを使用して印刷要求のコンテンツ・タイプを指定すると、これらのフィルターが印刷サービスによって自動的に呼び出されます。例えば、ユーザーが次のコマンドを入力したとします。

```
lp -d psprinter -T simple report2
```

出力先プリンター (*psprinter*) が PostScript プリンターとしてシステムに定義されていれば、ASCII ファイル `report2` (*ASCII* またはシンプル・フォーマットのファイル) は PostScript に自動的に変換されます。

フィルターが提供する追加の PostScript 機能

フィルターによって追加の PostScript 機能が提供される場合があります。

224 ページの『印刷フィルター』で説明したフィルターも、印刷の柔軟性を高めるために PostScript 機能を利用します。モード・オプション (**lp** コマンドの **-y** オプションによって呼び出される) を使用して、これらの機能のほとんどにアクセスできます。これらのフィルターを使用すれば、印刷ジョブに対して通常と異なるオプションをいくつか使用できます。次のリストでこれらのオプションについて説明し、それぞれ **lp** コマンド・ラインに組み込むオプションを示します。

項目	説明
-y reverse	ページの印刷順序を反転します。
-y landscape	物理ページの方向を縦長から横長に変更します。
-y x=*number*,**y=***number*	原点を移動して、物理ページ上での論理ページのデフォルト位置を変更します。
-y group=*number*	単一の物理ページにある複数の論理ページをグループ化します。
-y magnify=*number*	文書内の各ページの論理サイズを変更します。
-o length=*number*	文書の各ページ内の行数を選択します。
-P *num_list*	印刷される文書のサブセットをページ番号によって選択します。ここで、*num_list* はコンマ区切りのページ番号またはページ範囲です (例えば、**1,4,6-8,14-** はページ 1、4、6、7、8、および 14 から最後までを印刷します)。
-n *number*	文書を複数部印刷します。

注: PostScript 出力を作成するアプリケーションに対してこれらのフィルターを使用する場合は、アプリケーションのフォーマットが PostScript ファイルの構造化コメントのフォーマットに準拠していることを確認してください。特に、各 PostScript ページの先頭に、次のコメントによってマークを付ける必要があります。

```
%%Page: label ordinal
```

ここで、*ordinal* は文書の一連のページ内でページの位置を指定する正整数、*label* は任意のページ・ラベルです。

例えば、コンテンツ・タイプ **simple** (このファイルのコンテンツが ASCII フォーマットであることを示す) のファイル `report2` があるとします。このファイルのうち 6 つのページ (ページ 4 から 9) を印刷します。それぞれの物理ページには 2 つの論理ページが含まれています。システムにあるプリンターの 1 つ (**psprinter**) が PostScript プリンターなので、次のコマンドを入力してこの印刷を行うことができます。

```
lp -d psprinter -T simple -P 4-9 -y group=2 report2
```

これらの論理ページをグループ化するフィルターは、スペースの使用効率が最大限になるように物理ページ上でこれらのページの配置を試みます。このため、`group=2` を指定すると、物理ページが横長の方向になるように、ページは横並びに印刷されます。横長モードでは、物理ページではなく論理ページの方向が制御されるので、`group=2` オプションと組み合わせた場合は、2 つの論理ページが上下に重なって配置されます。

PostScript プリンターのサポート

PostScript プリンターのサポートは、他のプリンターのサポートと同様です。

lpadmin コマンドを使用して、PostScript プリンターをシステムに対して定義する必要があり、プリンターの管理のために適切なソフトウェアをインストールする必要があります。フォントのサポートと、*低速* フィルター処理を行う環境の設定のために、PostScript プリンターに対して追加の作業が必要になることがあります。

PostScript プリンターのインストールと保守:

PostScript プリンターは、他のプリンターと同様に、**lpadmin** コマンドを使用してインストールされます。

PostScript プリンターは PS インターフェース・プログラムを使用する必要があります。このプログラムは、**lpadmin** コマンド・ラインに **-m PS** を指定することによって要求されます。

注: PostScript プリンターのプリンター・タイプとコンテンツ・タイプは、PostScript フィルター内で使用されているプリンター・タイプと整合している必要があります。このため、プリンター・タイプ **PS**、**PS-b**、**PS-r**、または **PS-br**、およびコンテンツ・タイプ **PS** を指定して、PostScript プリンターをインストールしてください。

PS プリンター・タイプには 2 つの役割があります。まず、印刷サービスはこれらのタイプによって、プリンターと通信するための適切な高速フィルターを活動化します。**PS** と **PS-r** は、シリアル・ポート経由で接続されたプリンターとの通信に使用され、**PS-b** と **PS-br** は、パラレル・ポート経由で接続されたプリンターとの通信に使用されます。2 つ目に、**PS** インターフェースは PS プリンター用の PostScript バナー・ページを作成します。プリンター・タイプが **PS-r** または **PS-br** ならば、バナー・ページは最後に印刷され、文書のページは逆順に印刷されます。プリンター・タイプは、**lpadmin** コマンドの **-T** オプションによって指定されます。

PostScript ページ順序テーブル:

PostScript プリンターを構成する際に、特定のコンテンツ・タイプを選択することによって、入力のフィルター処理を選択できます。

プリンター	接続タイプ	ページ順序
PS	シリアル	通常
PS-b	パラレル	通常
PS-r	シリアル	逆
PS-br	パラレル	逆

-b 仕様 (**PS-b** または **PS-br** を選択した場合に使用される) はバッチ を表し、通常はパラレル接続に使用されますが、PostScript プリンターの状況メッセージが必要ない場合は、シリアル接続に使用することもできます。**PS** と **PS-r** のプリンター・タイプは、パラレル接続には使用できません。

PostScript プリンターを構成する際に **lpadmin** コマンドの **-I** オプションを指定すると、低速フィルター処理を使用せずにプリンターによって処理されるコンテンツ・タイプを指示できます。サーバー・システム上にあるプリンターの場合は、**PS** が適切なコンテンツ・タイプです。ただし、クライアント・システム上のプリンターの場合は、ネットワークとシステム・リソースの管理が問題になる可能性があるので、どこで低速フィルター処理を行うか検討してください。

PS 以外の有効なコンテンツ・タイプを指定すると、サーバー・システム上で入力の低速フィルター処理を強制的に行うことができます。逆に、コンテンツ・タイプ **PS** を指定すると、高速フィルター処理と印刷のために印刷要求がサーバー・システムに転送される前に、入力のフィルター処理がローカル側で行われます。

サーバー・システム上でプリンターを構成するには、次のコマンドを入力します。

```
/usr/sbin/lpadmin -p ps1 -T PS-b -I PS -m PS
```

ローカル・フィルター処理を行わないクライアント・システム上でプリンターを構成するには、次のコマンドを入力します。

```
/usr/sbin/lpadmin -p ps1 -T PS-b -I simple,daisy,dmd,tek4014,plot
```

ローカル・フィルター処理を行うクライアント・システム上でプリンターを構成するには、次のコマンドを入力します。

```
/usr/sbin/lpadmin -p ps1 -T PS-b -I PS
```

インストール手順の中で、プリンターにフォントをインストールしたり、コンピューターにダウンロード可能フォントをインストールしたりすることができます。詳細については、 242 ページの『PostScript フォントのインストールと保守』を参照してください。

PostScript フィルターのインストールと保守:

付属の PostScript フィルターは、ほとんどの状況に対応できます。ただし、環境によっては、フィルターの記述を変更して、異なる方法でフィルターをインストールすると役立つことがあります。

ここでは、これらのフィルターの場所と機能について説明します。PostScript フィルターは、ディレクトリー `/usr/lib/lp/postscript` に格納されています。

> **注:** フィルターには高速フィルターと低速フィルターの 2 種類があります。これらのタイプの定義については、**lpfilter**、および 228 ページの『印刷フィルターの定義』を参照してください。

システムと PostScript プリンター間で通信を行うための前提条件は、システムに **postio** フィルターまたは **lp.cat** フィルターが存在することです。これらのプログラムのみが、PostScript プリンターと直接通信する必須の PostScript フィルターです。他のタイプの文書を PostScript に変換し、PostScript プリンターで印刷できるようにするフィルターについては、 242 ページの『PostScript フィルター』を参照してください。

PostScript フィルター:

ファイル・コンテンツ・タイプ、および関連したフィルターのリストを示します。

ファイル・コンテンツ・タイプ	フィルター
simple	**postprint**
troff	**dpost**
daisy	**postdaisy**
dmd	**postdmd**
tek4014	**posttek**
plot	**postplot**

特殊な機能を実行するフィルターについては、『特殊用途 PostScript フィルター』を参照してください。

特殊用途 PostScript フィルター:

特殊用途フィルターとその機能のリストを示します。

機能	フィルター
プリンターとの通信	**postio、lp.cat**
フォントのダウンロード	**download**
ページの反転または選択	**postreverse**
マトリックス・グレースケール	**postmd**

PostScript フォントのインストールと保守:

PostScript の利点の 1 つは、フォントを管理する能力です。フォントはタイプ 1 フォーマットのアウトライン形式で、プリンターに格納されるか、プリンターと通信するコンピューターに格納されます。

文書を印刷する際に、PostScript インタープリターはそれぞれの文字をそのアウトライン記述から必要に応じて (適切なサイズで) 生成します。文書に必要なフォントが、使用されるプリンターに格納されていない場合は、文書を印刷する前にプリンターにそのフォントを転送する必要があります。この伝送処理は、*フォントのダウンロード* と呼ばれます。

フォントの格納とアクセスは、いくつかの方法で行われます。

- フォントはプリンターに永続的に格納できます。これらの*プリンター常駐* フォントは、製造メーカーによってプリンターの ROM にインストール済みの場合があります。プリンターにディスクが備わっている場合は、ユーザー (つまり、印刷サービス管理者) がフォントをそのディスクにインストールできます。ほとんどの PostScript プリンターには 35 種類の標準フォントが出荷時に付属していますが、廉価版モデルの場合は 13 種類のみです。
- **exitserver** オペレーターを使用する特殊な PostScript プログラミング手法により、フォントをプリンターに送信することによって*永続的にダウンロード* できます。この方法でダウンロードされたフォントは、プリンターの電源がオフになるまでプリンターのメモリーに残ります。このフォントにメモリーが割り当てられるので、PostScript 印刷要求に使用可能なメモリーが減少します。**exitserver** プログラムを使用するにはプリンター・システムのパスワードが必要で、このプログラムはプリンター管理者用に予約済みの場合があります。そのプリンターがサービスを提供する印刷要求の大部分がそのフォントを継続的に使用する場合には、この方式が便利です。
- フォントをユーザーによる印刷要求の前に付加し、ユーザー印刷要求の一部として送信できます。文書が印刷されると、フォントに割り当てられていたスペースは他の印刷要求のために解放されます。フォントは、ユーザーのディレクトリーに格納されます。この方式は、使用量が限られているフォントの場合にお勧めします。

- 多数のユーザーが共用するシステムにフォントを格納できます。これらのフォントは、*ホスト常駐* と呼ばれます。このシステムは、プリンターのサーバーである場合も、ネットワークによってプリンターに接続されたシステムである場合もあります。それぞれのユーザーが、印刷される文書内でフォントを要求できます。使用可能フォントが多数ある場合、またはすべての印刷要求がこれらのフォントを継続的に使用しない場合には、この方式が便利です。サーバーに接続されたプリンター上のみでフォントが使用される場合は、これらのフォントをサーバーに格納してください。1 つのシステム上で、ネットワーク上の複数のプリンターにジョブを送信できるユーザーがフォントを使用する場合は、そのユーザーのシステムにフォントを格納してください。

印刷サービスは、リストの最後に示した方式を使用してフォントを管理するために、特殊なダウンロード・フィルターを提供しています。

印刷サービスは、多数の PostScript プリンターに常駐している 35 種類の標準 PostScript フォント用の **troff** 幅テーブルを使用できます (**dpost** プログラムによって使用される)。

プリンター常駐フォントのリストの取得:

ほとんどの PostScript プリンターには、プリンター ROM 常駐のフォントが出荷時に装備されています。一部のプリンターには、追加フォントを格納するディスクが備わっています。

接続された PostScript プリンターの ROM またはディスクにあるタイプ 1 フォントのリストは、プリンター製造メーカーの資料から入手できます。シリアル・ポート経由で接続された PostScript プリンターの場合は、**postio** コマンド、および PostScript プログラム **romfonts.ps** を使用して、これらのフォントのリストを生成することもできます。

シリアル・ポートに接続された PostScript プリンター用のプリンター常駐フォントのリストを取得するには、次の手順で行います。

1. PostScript プリンターが接続されている先のデバイスを取得します。

   ```
   lpstat -v
   ```

   PostScript プリンター **prlocal** がシリアル・ポート経由で接続されているシステムの場合、このコマンドは次のような出力を戻します。

   ```
   device for prlocal: /dev/tty01
   ```

   この出力は、プリンターがデバイス `/dev/tty01` に接続されていることを示しています。
2. *root* ユーザー として、次のコマンドを実行します。

   ```
   cd /usr/lib/lp/postscript
   ```
3. *root* ユーザー として、次のコマンドを実行します。

   ```
   postio -L /tmp/postio.o -l /dev/tty01 -t romfonts.ps
   ```

   この例の **prlocal** プリンターの場合、これによって次のような出力がファイル `/tmp/postio.o` 内に生成されます。

   ```
   printer startup
       %%[ status: waiting; source: serial 25 ]%%
       %%[ status: endofjob ]%%
       %%[ status: idle ]%%
       sending file romfonts.ps
       waiting for end of job
       %%[ status: busy; source: serial 25 ]%%
       /AGaramond-Bold
       /AGaramond-BoldItalic
       /AGaramond-Italic
   ```

```
        /AGaramond-Regular
        /AvantGarde-Book
        /AvantGarde-BookOblique
        /AvantGarde-Demi
        /AvantGarde-DemiOblique
       . . . more PostScript font
names . . .
        /ZapfChancery-MediumItalic
        /ZapfDingbats
        %%[ status: endofjob ]%%
    job complete
```

この例では、**prlocal** プリンターのプリンター常駐フォントがリストされています。

プリンターのフォント・リストへのプリンター常駐フォントの追加:

プリンターのフォント・リストにプリンター常駐フォントを追加できます。

プリンターのインストール時に、プリンター常駐フォントのリストをそのプリンターのフォント・リストに追加する必要があります。プリンターのメモリー内にあるフォント名のみを含むようにこのフォント・リスト・ファイルを編集し (例: AGaramond-Bold から ZapfDingbats まで)、ファイル `/etc/lp/printers/prlocal/residentfonts` に保管すれば、ホスト・コンピューターからこれらのフォントがダウンロードされないようになります。

プリンターのフォント・リストにプリンター常駐フォントを追加するには、次の手順で行います。

1. フォント・リストが保管されているプリンター管理ディレクトリーまでナビゲートします。特定のプリンターに対して、このフォント・リストはファイル `/etc/lp/printers/`*printer-name*`/residentfonts` に保管されています。 **-p** オプションを指定すると、**download** はこのファイルを検査して、プリンター内でどのタイプ 1 フォントが ROM 常駐およびディスク常駐 (一部の PostScript プリンターにはフォント・ディスクが直接接続されています) であるかを調べ、これらのフォントをダウンロードしないようにします。
2. **lpadmin** コマンドを使用してシステム上で PostScript プリンターが最初にセットアップされるとき、このファイルは自動的には作成されません。このファイルはユーザー自身が作成する必要が生じることがあります。(フォント・リスト・ファイルは、手動で編集する必要があります。つまり、**vi** などのテキスト・エディターを使用します。)

   フォントをプリンターに永続的にダウンロードした場合は、このファイルにフォント名を追加します。(これにより、フォントが既にプリンター上に存在する場合に、時間のかかるフォントのダウンロードが行われないようになります。)

   注: プリンターがリモート・システムに接続されている場合は、そのシステムに常駐する、プリンターにダウンロード可能なフォントをこのリストに含める必要があります。これにより、ネットワーク経由でフォントの不必要な送信が行われなくなります。

ホスト常駐フォントのインストールと保守:

一部のフォントはホストに常駐し、個々の印刷要求の必要に応じてプリンターに伝送されます。

システム上のユーザー全員が PostScript フォントを使用できるようにすることは、管理者の責務です。このためには、前述のガイドラインを使用して、これらのフォントをインストールする方法と場所を理解しておく必要があります。フォントは名前によって要求され、ファイルに格納されているので、印刷サービスはフォントの名前と、これらのフォントを格納するファイルの名前の間にある対応を示すマップ・ファイルを維持しています。フォントをホストにインストールする際には、これらの情報を両方とも更新する必要があります。

ホスト常駐の PostScript フォントをインストールするには、次の手順で行います。

1. フォント・ファイルを適切なディレクトリーにコピーします。PostScript プリンターで使用可能なフォントは、/usr/share/lib/hostfontdir ディレクトリーまたはその他のディレクトリーにあります。
2. フォントの名前と、そのフォントが格納されているファイルの名前を、マップ・テーブルに追加します。また管理者は、hostfontdir ディレクトリー内で、製造会社 (フォントを作成した会社) がそれぞれのフォントに割り当てた名前と、そのフォントが格納されているファイルの名前の間にある対応を示すマップ・テーブルを作成し、維持する必要があります。スラッシュ (/) から始まるファイル名は、そのまま使用されます。それ以外のパス名は、ホスト・フォント・ディレクトリーを基準とした相対パスです。マップ・テーブル内のコメントは、**%** (PostScript の場合と同様) によって導入され、行の最後まで続きます。例えば、*Palatino Bold* というフォントをマップするには、次の行をマップ・テーブルに追加します。

   ```
   Palatino-Bold /usr/share/lib/hostfontdir
   ```

   (マップ・テーブル自体は、/usr/share/lib/hostfontdir/map ファイル内にあります。)

   このエントリーがシステムのマップ・テーブルに追加された後、ユーザーは印刷ジョブに Palatino Bold フォントを使用できるようになります。このフォントの要求を含むファイルの印刷を実行依頼するとき、印刷サービスはファイルをプリンターに送信する前に、ファイル /usr/share/lib/hostfontdir のコピーをそのファイルの前に付加します (residentfonts ファイル内でこのフォントが定義されていない限り)。
3. **troff** を使用する場合は、標準の **troff** フォント・ディレクトリー内で、このフォント用の新しい幅テーブルを作成する必要があります。

ホスト常駐フォントのダウンロード:

プリンターにロードされていないフォントの要求が PostScript 文書に含まれている場合は、**download** フィルターがこの要求を管理します。

download フィルターは、高速フィルターとして呼び出され、フォントがプリンターと同じシステムに常駐している場合は、フォントを自動的にダウンロードします。**download** フィルターは、リモート・プリンターにフォントを送信することもできます。このためには、**-y** オプションを使用して、**download** フィルターを低速フィルターとして呼び出す新規フィルター・テーブル・エントリーを作成できます。また、入力タイプを変更して、このフィルターの選択を強制することもできます。

download フィルターは次の処理を行います。

- PostScript 文書を検索して、要求されているフォントを判別します。これらの要求は、ヘッダー・コメント内で次の PostScript 構造化コメントによって文書化されています。

  ```
  %%DocumentFonts: font1 font2 . . .
  ```

- そのプリンターの常駐フォントのリストを検索して (/etc/lp/printers/printer-name/residentfonts 内)、要求されたフォントをダウンロードする必要があるかどうか調べます。
- フォントがプリンターに常駐していない場合は、ホスト常駐フォント・ディレクトリーを検索して、要求されたフォントが使用可能かどうか調べます。ダウンロード候補となるフォントは、*download* に読み取り可能なファイルを指示するマップ・テーブルにリストされたフォントのみです。タイプ 1 フォントは、%%DocumentFonts: コメントや PostScript ファイルに複数回現れる場合であっても、単一の文書に対して 1 回 (最大) ダウンロードされます。フォントのダウンロードは、PostScript ジョブの実行期間中にのみ行われます。ただし、**exitserver** オペレーターを使用した特殊な PostScript プログラミング手法を使用して、プリンターの RAM にフォントの永続ダウンロードを行うことができます。

リストされていないフォント、またはアクセス不能ファイルの要求は無視されます。マップ・テーブルを読み取ることができない場合は、すべての要求が無視されます。

- フォントが使用可能ならば、フィルターはそのフォントのファイルを取得し、印刷されるファイルの前に付加します。
- フィルターは、フォント定義ファイルと PostScript ソース・ファイル (印刷されるファイル) を PostScript プリンターに送信します。

フォント・カートリッジと文字セット

それぞれのプリンターで使用可能なフォント・カートリッジまたは文字セットを指定できます。

プリンターによって、さまざまなフォント・スタイルを印刷する方法に違いがあります。フォント・カートリッジを使用するものや、事前プログラム済みの選択可能な文字セットを使用するものなどがあります。印刷サービスは、こうした違いからユーザーが受ける影響を最小限にすることができます。

使用可能なフォント・カートリッジまたは文字セットをリストする際には、これらに名前を割り当てます。これらの名前は、管理者の便利のため、およびシステムのユーザーの便利のために使用されます。異なるプリンターに類似したフォント・カートリッジや文字セットが存在する場合があるので、すべてのプリンターに共通のフォント名を使用してください。これにより、ユーザーはどのプリンターが使用されているか、フォント・カートリッジまたは選択可能文字セットのどちらが使用されているかを知らなくても、ファイルの印刷を実行依頼し、特定のフォント・スタイルを要求できます。

プリンターが装てん可能フォント・カートリッジを使用している場合は、そのカートリッジの名前のみをリストすれば済みます。プリンターが選択可能文字セットを使用している場合は、そのセットの名前をリストし、さらに **terminfo** データベース内でセットを一意的に識別する名前または番号に、それぞれのセットをマップする必要があります。

文字セットの指定:

選択可能文字セットを使用できるプリンターの場合は、文字セットの名前を判別し、それぞれのセットを **terminfo** データベース内の名前または番号にマップします。

- **terminfo** データベースにリストされている文字セットの名前を判別するには、次のように入力します。

  ```
  tput -T printer-type csnm 0
  ```

  printer-type は、対象のプリンター・タイプの名前です。このコマンドは、0 番目の文字セット (プリンターの初期化後にデフォルトで得られる文字セット) の名前を表示します。

  その他の文字セットの名前を表示するには、前記のコマンドを繰り返し、0 を 1、2、3 などに置き換えてください。一般に、**terminfo** の名前はプリンターのユーザー資料に使用されている名前と厳密に一致している必要があります。ただし、すべての製造メーカーが同じ名前を使用しているとは限らないので、**terminfo** の名前はプリンター・タイプ間で異なる場合があります。
- 文字セット名のリストを指定し、これらの文字セットを **terminfo** の名前または番号にマップするには、次のように入力します。

  ```
  /usr/sbin/lpadmin -p printer_name -S characterset_list
  ```

  characterset_list は、コンマ区切りまたはスペース区切りの名前のリストです。名前の区切りにスペースを使用する場合は、リスト全体 (-S を除く) を引用符で囲んでください。リストの各項目は、文字セット名のマッピング (別名) で、次のどちらかの形式です。

  ```
  csN=characterset_name
  characterset_name1=characterset_name2
  ```

変数 *N* は、**terminfo** データベース内の文字セットの番号を識別する、0 から 63 の番号です。*characterset_name1* は、文字セットを **terminfo** データベース内の名前によって識別します。どちらの例でも、等号 (=) の右側にある名前は、文字セットの別名として選択した名前です。

注: terminfo の名前が適切ならば、文字セットの別名のリストを指定する必要はありません。文字セットは、**terminfo** の名前、番号、または別名によって参照できます。

例えば、プリンターには標準文字セット (セット #0) のほかに 2 つの選択可能文字セット (セット #1 と #2) が存在するとします。プリンター・タイプは **5310** です。選択可能文字セットの名前を判別するには、次のコマンドを入力します。

```
tput -T 5310 csnm 1english
tput -T 5310 csnm 2finnish
```

`english` および `finnish` という単語は、選択可能文字セットの名前で、コマンドの出力です。文字セット 2 を参照するためには `finnish` という名前で十分ですが、標準セット (セット 0) とセット 1 にはより適切な名前が必要です。次のコマンドを入力して、同義語を定義します。

```
/usr/sbin/lpadmin -p printer_name -S "cs0=american, english=british"
```

これにより、次の 3 つのコマンドから得られる結果は同じになります。(**lp** コマンドは印刷ジョブをプリンターに送付し、この例では **cs1** 文字セットを処理できるいずれかのプリンターに印刷ジョブを送付します。)

```
lp -S cs1 -d any . . .

lp -S english -d any . . .


lp -S british -d any . . .
```

プリンターで使用できる文字セットをリストしない場合、サービスは、選択可能文字セットを使用するプリンターが、そのプリンターの既知の **cs***N* 名または **terminfo** 名をいずれも受け入れるものと想定します。

- 文字セットのマッピングを除去するには、次のように入力します。

  ```
  /usr/sbin/lpadmin -p printer_name -S none
  ```

-

プリンターで使用するフォント・カートリッジの指定:

新規プリンターで使用できるフォント・カートリッジを指定するまで、印刷サービスはどのフォント・カートリッジもそのプリンターに取り付け可能であると見なさず、フォント・カートリッジを必要とする印刷要求をすべてリジェクトします。

- プリンターで使用するフォント・カートリッジのリストを指定するには、次のように入力します。

  ```
  /usr/sbin/lpadmin -p printer_name -S font_cartridge_list
  ```

  font_cartridge_list は、コンマ区切りまたはスペース区切りのフォント・カートリッジ名のリストです。名前の区切りにスペースを使用する場合は、リスト全体 (`-S` を除く) を引用符で囲んでください。これらのフォント・カートリッジのみが、プリンターに取り付け可能であると見なされます。
- フォント・カートリッジ・リストをプリンターから除去するには、次のように入力します。

  ```
  /usr/sbin/lpadmin -p printer_name -S none
  ```

プリンターに取り付け可能なフォント・カートリッジのリストを指定した後、これらのカートリッジを取り付けることができます。『プリンターのフォント・カートリッジの交換』を参照してください。

プリンターのフォント・カートリッジの交換:

フォント・カートリッジを必要とするファイルを印刷サービスが印刷する前に、プリンターにフォント・カートリッジを取り付け、マウントする必要があります。

フォント・カートリッジのアラートをセットアップした場合は、取り付けとマウントを行うフォント・カートリッジに対して十分な数の印刷ジョブがキューに入れられると、印刷サービスはアラートを出します。詳細については、250 ページの『フォームのマウントとフォント・カートリッジに関するアラート』を参照してください。

フォント・カートリッジを交換するには、まず現行フォント・カートリッジをプリンターから取り外す必要があります。次に、プリンターに新規フォント・カートリッジを取り付け、新規フォント・カートリッジが使用可能であることを、カートリッジのマウントによって印刷サービスに通知する必要があります。現在印刷中のプリンター上でこの作業を行うことは難しく、印刷サービスはプリンターのフォント・カートリッジを必要としないファイルの印刷を継続するので、まずプリンターを使用不可に設定してください。

フォント・カートリッジの取り付けまたは交換を行うには、次の手順で行います。

1. プリンターを使用不可にします。
2. プリンターから現行フォント・カートリッジを取り外します (該当する場合)。
3. プリンターに新規フォント・カートリッジを取り付けます。
4. 次のように入力して、新規フォント・カートリッジをマウントします。

   ```
   /usr/sbin/lpadmin -p printer_name -M -S font_cartridge_name
   ```

   フォント・カートリッジを必要とする印刷要求がすべて、*printer_name* で印刷されます。
5. プリンターを再度使用可能にします。

フォント・カートリッジをアンマウントするには、次のように入力します。

```
/usr/sbin/lpadmin -p printer_name -M -S none
```

注: 現行フォント・カートリッジをプリンターから物理的に取り外した後、新規フォント・カートリッジを取り付けてマウントする前に、現行フォント・カートリッジをアンマウントする必要はありません。

プリンター障害アラート

印刷サービスは、プリンター障害を検出してアラートを出すための方式を備えています。

障害は、用紙切れ、リボン切れ、トナー切れなどの単純な問題から、地域の電源障害やプリンター障害などの重大な問題までさまざまです。障害の指標も広い範囲にわたり、キャリア (プリンターがオンラインであることを示すシグナル) の損失から、XOFF またはメッセージの送信まで多岐にわたります。

印刷サービス自体が認識するプリンター障害の指標は、ハングアップ (キャリアの損失) と印刷の過大な遅延 (対応する XON がない XOFF フロー制御文字) の 2 つのクラスのみです。これら以外の障害については、プリンター・サービスは障害の原因を判別できないので、アラートを出すことができません。ただし、その他のプリンター障害を検出して印刷サービスに通知するフィルターを追加すれば、印刷サービスからアラートを受けることができます。詳しくは、224 ページの『印刷フィルター』を参照してください。

プリンター障害アラートの発行:

プリンター障害の発生時にアラートを発行するように印刷サービスを調整するには、次のコマンドのいずれかを入力します。

- /usr/sbin/lpadmin -p *printer-name* -A mail -W *minutes*
- /usr/sbin/lpadmin -p *printer-name* -A write -W *minutes*
- /usr/sbin/lpadmin -p *printer-name* -A '*command*' -W *minutes*

最初の 2 つのコマンドは、アラートが発生するたびに、それぞれメール・メッセージを送信するか、メッセージを端末装置に直接書き込むように印刷サービスに指示します。 3 番目のコマンドは、アラートが発生するたびに *command* コマンドを実行するように印刷サービスに指示します。*command* を実行するために、3 番目のコマンドを入力した時点で現在有効になっているシェル環境が保管され、復元されます。この環境には、環境変数、ユーザー ID とグループ ID、および現行ディレクトリーが含まれます。 *minutes* 引数は、アラートを繰り返す間隔の分数です。

プリンター障害アラートを使用不可にする:

必要に応じて、プリンター障害アラートを使用不可にすることができます。

障害発生時に印刷サービスがアラートを発行しないようにする場合は、次のように入力します。

```
/usr/sbin/lpadmin -p printer-name -A none
```

プリンター障害の発生時に別のユーザーにメールを送信するかメッセージを書き込むようにしたい場合は、**lpadmin** コマンドに **-A'** *mail login-ID*' オプションまたは **-A 'write** *login-ID*' オプションを指定して使用します。 *login-ID* を指定しない場合、メールまたはメッセージは現行ログイン名に送信されます。**su** コマンドを使用してログインを変更した場合、このログインは自身のものではない可能性があります。

反復プリンター障害アラートを使用不可にする:

障害が発生した後、アラートを反復して受け取り始めた場合に、アラートの送信を停止するように印刷サービスに指示できます (現行の障害のみを対象として)。

アラートの送信を停止するように印刷サービスに指示するには (現行の障害のみを対象として)、次のコマンドを実行します。

```
/usr/sbin/lpadmin -p printer-name -A quiet
```

注: アラート・タイプ **quiet** は、アクティブ・アラートの終了のみに使用し、新規プリンターのアラート・タイプとして **quiet** を指定しないでください。

ここで説明するコマンドのいずれかの *printer-name* が **all** ならば、アラート条件はすべてのプリンターに適用されます。

アラート方式を定義しない場合は、プリンター障害が発生するたびにメールを 1 回受け取ります。**-W** オプションを指定せずに方式を定義した場合は、障害が発生するたびにアラートが 1 回出されます。

プリンター障害リカバリー・メカニズム

プリンター障害が修正され、プリンターが再び印刷可能な状態になったときに、プリンター障害リカバリー・メカニズムによって印刷サービスのリカバリーが可能です。

注: システムのユーザーのためにリモート・プリンターをアクセス可能にしている場合には、この情報は適用されません。

プリンター障害が修正され、プリンターが再び印刷可能な状態になったときに、印刷サービスは次のようにしてリカバリーします。

- 印刷が停止したページの先頭から印刷を続行します。
- 障害が発生したときにアクティブだった印刷要求の先頭から印刷を再開します。
- プリンターを再び使用可能にするようにユーザーが印刷サービスに指示するまで待機します。

注: 印刷が停止した個所でページの先頭から印刷を続行するには、印刷を正しく再開する前に、プリンター障害が解消するまで待つことができるフィルターを使用する必要があります。ページ境界を追跡し、ファイルの印刷が停止した個所を認識できるように、このフィルターはプリンターが使用している制御シーケンスを詳細に解釈できることが必要です。印刷サービスに付属して提供されるフィルターには、このような機能はありません。適切なフィルターが使用されていなければ、リカバリーを意図したとおりに進行できないことがアラートによって通知されます。

プリンターの障害リカバリー・メカニズムの指定:

lpadmin コマンドを使用して、プリンター障害リカバリー・メカニズムを指定できます。

障害が解消した後に印刷サービスをリカバリーする方法を指定するには、次のいずれかのコマンドを入力します。

- `/usr/sbin/lpadmin -p printer-name -F continue`
- `/usr/sbin/lpadmin -p printer-name -F beginning`
- `/usr/sbin/lpadmin -p printer-name -F wait`

これらのコマンドは、ページの先頭から続行するか、最初から再始動するか、またはユーザーが **enable** コマンドを入力してプリンターを再び使用可能にするまで待つかを印刷サービスに指示します。

プリンター障害の発生後に印刷サービスを再開する方法を指定しない場合、印刷サービスは印刷が停止したページの先頭から続行するか、それに失敗した場合は印刷要求の最初から続行することを試みます。

continue を指定してリカバリーする場合に、インターフェース・プログラムが継続して実行されていないためにプリンター障害が解消したことを検出できない場合は、成功するまで数分ごとに印刷が試行されます。 **enable** コマンドを発行すれば、印刷サービスの即時再試行を強制できます。

フォームのマウントとフォント・カートリッジに関するアラート

フォームをマウントする必要がある場合、またはフォント・カートリッジまたはフォームを待機している要求がしきい値を超えた場合に、アラートを出すように印刷システムを設定できます。

交換可能なフォント・カートリッジを受け入れるプリンターを使用していて、それぞれのプリンターで使用できるフォント・カートリッジをリストしてある場合は、特定のフォント・カートリッジを使用する印刷要求をユーザーが実行依頼できます。ただし、ユーザーが使用を要求したときにフォント・カートリッジがマウントされていない場合は、フォント・カートリッジをマウントするまでジョブはキュー内で待機します。 247 ページの『プリンターで使用するフォント・カートリッジの指定』を参照してください。フォームを指定してファイルを印刷する際に、そのフォーム (またはフォント・カートリッジ) がマウントされていない場合は、該当するフォームをマウントするまで、ジョブはキュー内で待機します。 223 ページの『フォームのマウント』を参照してください。

フォームとフォント・カートリッジのマウントを促すアラートのセットアップ:

フォームとフォント・カートリッジのマウントを促すアラートをセットアップできます。

フォームをマウントする必要があることを示すアラートを設定するには、次のように入力します。

```
lpforms -f form_name -A alert_method -Q number -W minutes
```

フォント・カートリッジをマウントする必要があることを示すアラートを設定するには、次のように入力します。

```
lpadmin -S font_cartridge_name -A alert_method -Q number -W minutes
```

変更のセットアップ

項目	説明
alert_method	使用するアラート方式 (**mail**、**write**、または任意のコマンド)
number	アラートを再開するまでの待ち要求の数
minutes	アラートの間隔の分数

フォームまたはフォント・カートリッジのアラート方式を定義しない場合は、そのアラートが出されません。方式を定義し、ただしアラート間隔の分数を定義 (**-W** オプションを使用) しない場合は、アラートが毎回 1 つずつ出されます。

- プリンター・キューに **check** フォームの要求が複数入っていて、そのフォームがまだマウントされていない場合に、電子メール・アラートを 5 分ごとに送信するように印刷サービスに指示するには、次のように入力します。

  ```
  lpforms -f check -A mail -Q 2 -W 5
  ```

- プリンター・キューに **dingbat** フォント・カートリッジの要求が 3 つ以上入っていて、**dingbat** フォント・カートリッジがまだマウントされていない場合に、端末装置にアラートを 2 分ごとに書き込むように印刷サービスに指示するには、次のように入力します。

  ```
  lpadmin -S dingbat -A write -Q 3 -W 2
  ```

- いずれかのフォームまたはフォント・カートリッジの要求がキューに入っている場合にアラートを出すように設定するには、次のいずれかを入力します。

  ```
  lpforms -f any -A mail -W 5
  ```

  ```
  lpadmin -S any -A mail -W 5
  ```

- フォームまたはフォント・カートリッジのマウントを指示するアラート・メッセージの受信を停止するには、次のいずれかを入力します。

  ```
  lpforms -f form_name -A quiet
  ```

  ```
  lpadmin -S font_cartridge_name -A quiet
  ```

- フォームまたはフォント・カートリッジをマウントする必要があるときにアラートを出さないようにするには、次のいずれかを入力します。

  ```
  lpforms -f form_name -A none
  ```

  ```
  lpadmin -S font_cartridge_name -A none
  ```

フォームとフォント・カートリッジのマウントを促すアラート・メッセージの停止:

プリンター・ポート特性

プリンター・ポート特性は、**lpadmin** コマンドの **-o** "stty='*stty-option-list*'" オプションによって定義される属性です。

コンピューターに直接接続されるプリンター、およびネットワーク経由で接続されるプリンターのために、インターフェース・プログラムによってプリンター・ポート特性を設定する必要があります。これらの特性は、プリンターとの低レベル通信を定義します。特性には、ボー・レート、XON/XOFF フロー制御の使用、1 バイトあたりのビット数 7、8、またはその他、パリティーのタイプ、出力の後処理などがあります。standard インターフェース・プログラムは、**stty** コマンドを使用してプリンター・ポートを初期化し、ボー・レートとその他いくつかのデフォルト特性を最小限設定します。

デフォルトのプリンター・ポート特性:

標準インターフェース・プログラムによって適用されるデフォルト特性を、次の表にリストします。

デフォルト	説明
9600	9600 ボー・レート
cs8	8 ビット・バイト
-cstopb	1 バイトあたり 1 ストップ・ビット
-parenb	パリティー生成なし
ixon	XON/XOFF フロー制御を使用可能にする
-ixany	XON による出力の再開のみを許可する
opost	次に示すようにデータ・ストリームの後処理を行う
-olcuc	小文字を大文字にマップしない
onlcr	改行を復帰/改行にマップする
-ocrnl	復帰を改行にマップしない
-onocr	列 0 でも復帰を出力する
nl0	改行の後に遅延なし
cr0	復帰の後に遅延なし
tab0	タブの後に遅延なし
bs0	バックスペースの後に遅延なし
vt0	垂直タブの後に遅延なし
ff0	用紙送りの後に遅延なし

デフォルト特性がご使用のプリンターの要件を十分満たしている場合もありますが、プリンターにはさまざまなものがあるので、異なる特性を設定する必要が生じることもあります。すべての特性のリストについては、**stty** コマンドを参照してください。

stty プログラムによって処理されるものと異なるプリンター・ポート特性を必要とするプリンターを使用している場合は、インターフェース・プログラムをカスタマイズする必要があります。詳しくは、 215 ページの『プリンター・インターフェース・スクリプト』を参照してください。

新規プリンターを追加する際には、追加のポート特性のリストを指定できます。指定するリストはデフォルト・リストの後に適用されるので、変更したくない項目をリストに含める必要はありません。追加のリストは次のように指定します。

```
/usr/sbin/lpadmin -p printer-name -o "stty='stty-option-list'"
```

stty-option-list に複数の項目を指定する場合、二重引用符と単一引用符が両方とも必要であることに注意してください。

グラフィカル・データの印刷にプリンターを使用するとします。この場合、改行文字は復帰を追加せずに単独で出力する必要があります。次のコマンドを入力します。

```
/usr/sbin/lpadmin -p printer-name -o "stty=-onlcr"
```

リストにある項目はただ 1 つなので、単一引用符が省略されている点に注意してください。

別の例として、プリンターに送信されるデータに奇数パリティーが必要であるとします。次のコマンドを入力します。

```
/usr/sbin/lpadmin -p printer-name -o "stty='parenb parodd cs7'"
```

複数の名前を持つプリンターのセットアップ

印刷サービスは、複数の機能を実行する単一のプリンターを、複数の名前によってセットアップできます。

例えば、ご使用のプリンターが縦長と横長の両方のモードをサポートしている場合は、それぞれの機能ごとに異なる名前をセットアップし、それぞれのプリンター名にジョブを送信できます。これら複数のプリンターは、*仮想プリンター* と呼ばれます。

印刷スプーラー・システムは、プリンターが接続されているデバイスによってではなく、名前のみによってプリンターを区別します。同じデバイスに異なる名前を指定し、複数の印刷ジョブが同時に出現しないようにするには、*実* プリンターと追加の*仮想* プリンターを両方ともセットアップする必要があります。実プリンターは実際の印刷を実行し、仮想プリンターは印刷ジョブを実プリンターに渡します。

- Hewlett-Packard LaserJet の機能を使用する 2 つの仮想プリンター **port** と **land** をセットアップするには、次の手順を使用します。
  1. 実プリンターをセットアップします。*Name* を **real** と設定し、*Model* を **HPLaserJet** に設定します。
  2. 仮想プリンター *port* と *land* をセットアップします。*Model* を **network** に設定します。**real** の接続先と同じ *Device* を指定します。
  3. ファイル `/usr/spool/lp/remote` を作成し、次の行を追加します。

     ```
     port: lp -dreal -oportrait
     land: lp -dreal -olandscape
     ```

     これにより、プリンター **land** への印刷時には、印刷システムは **-olandscape** オプションを使用して (横長モードで印刷するため) 印刷ジョブをプリンター **real** に送信し、プリンター **port** への印刷時には、印刷システムは **-oportrait** オプションを使用して (縦長モードで印刷するため) 印刷ジョブをプリンター **real** に送信します。

     注: -dreal の後にリストされるオプションは、プリンター・モデルによって異なります。ご使用のプリンターに対応する `/usr/spool/lp/admins/lp/interface` 内のインターフェース・スクリプトを調べて、プリンターまたはクラスに依存する **-o** オプションを判別してください。
- 横長モードでファイルを印刷するには、次のように入力します。

  ```
  lp -dland filename
  ```

  この手順により、**lp** の **-dland** オプションが、プリンターに必要なオプションに変換されます (この例では、**-dreal -ol**)。

  これを実現するもう 1 つの方法は、このタイプの印刷を実行する単純なシェル・スクリプトを作成することです。次に例を示します。

  ```
  :
  # Land - shell script to print in landscape mode
  #
  # syntax: land <file> <file> ...
  #
  #
  lp -dreal -ol $@
  ```

どちらの方式を選択するかは、ご使用のアプリケーションが印刷システムにアクセスする方法によって異なります。多くのアプリケーションはプリンターの名前の指定のみを受け入れるので、この場合は仮想プリンターが唯一の解決策です。その他のアプリケーションは、印刷ジョブを実行依頼するコマンドをより詳細に制御できます。この場合は、前述の例に示したシェル・スクリプトを使用できます。

AIX 上でのディレクトリー対応 (LDAP) システム V による印刷

Lightweight Directory Access Protocol (LDAP) は、ユーザー情報やその他のネットワーク関連エンティティーのリポジトリーにアクセスするために使用される、分散階層ディレクトリー・サービス・アクセス・プロトコルです。IBM Directory は、LDAP ディレクトリー・サーバーです。

AIX System V 印刷サブシステムは IBM Directory を使用するので、印刷情報の中央保管が可能です。この機能を使用して、プリンター、印刷キュー、およびシステム情報をクライアント/サーバー環境で共通して維持できます。**mkprtldap** コマンドは、System V 印刷情報を含むサーバー、および印刷情報の取得に IBM Directory (LDAP) を使用する 1 つ以上のクライアントとして、IBM Directory を構成します。

System V 印刷サブシステムはディレクトリー対応であるため、LDAP ディレクトリーに保管された情報を使用して System V 印刷サブシステムを管理できます。System V 印刷サブシステムは、情報をディレクトリーに保管するオプションを備えたさまざまな AIX サブシステムの 1 つです。ディレクトリーに保管された情報は、AIX システムを管理するためにサブシステムによって使用されます。 LDAP ディレクトリーを使用するその他のサブシステムには、セキュリティー、NIS (Network Information Service) などがあります。

AIX 上でのディレクトリー対応 (LDAP) System V による印刷には、次のものが必要です。

- AIX オペレーティング・システム
- IBM Directory Server および Client v4.1 以降

注: IBM Directory は、AIX 基本オペレーティング・システムのメディアに収録されています。

印刷サブシステムのセットアップの計画

IBM Directory (LDAP) を使用するために印刷サブシステムをセットアップするには、2 つのステップが必要です。

まず、System V 印刷情報を保管するために IBM Directory (LDAP) サーバーを構成します。このサーバーは、System V 印刷情報の中央リポジトリーとして機能します。次に、IBM Directory Server を使用して System V 印刷情報を取得するように、ホスト・システム (クライアント) を構成します。

注: AIX 上でのディレクトリー対応の System V による印刷をセットアップするために使用する **mkprtldap** コマンドは、root ユーザーのみが実行できます。 **mkprtldap** コマンドは、System V 印刷情報に関して IBM Directory を使用するために、IBM Directory のサーバー・システムとクライアント・システムを構成する処理のみを行います。プリンター、印刷キュー、およびシステムの追加、削除、および管理を行うには、ディレクトリー対応の System V 印刷コマンド (**dslpaccept**, **dslpaccess**、**dslpadmin**、**dslpdisable**、**dslpenable**、**dslpreject**、および **dslpsearch**) を実行します。ディレクトリー対応の System V 印刷コマンドは、`bos.svprint` ファイル・セットに含まれています。このファイル・セットは、クライアントとサーバーの両方にインストールする必要があります。ディレクトリー対応の System V 印刷コマンドを実行する前に、**mkprtldap** コマンドを使用して、クライアント側の構成を完了しておく必要があります。

System V 印刷情報を保管するための IBM Directory (LDAP) の構成:

System V 印刷情報を保管するために IBM Directory (LDAP) を構成できます。

System V 印刷情報の中央リポジトリーとして機能する AIX システムに IBM Directory Server ソフトウェアをインストールし、構成するには、次の手順で行います。

注: システムに IBM Directory Server がインストール済みの場合は、ステップ 2 に直接進んでください。

1. AIX 基本オペレーティング・システムのメディア・ソフトウェアから、IBM Directory Server ソフトウェアをインストールします。IBM Directory は IBM DB2® データベースを必要とします。IBM DB2 データベースは、システムに既にインストール済みでない限り、IBM Directory Server のインストール時にデフォルトでインストールされます。

   注: IBM Directory のインストールとトラブルシューティングの詳しい指示については、IBM Directory 製品に付属の資料を参照してください。
2. System V 印刷情報を保管するために IBM Directory を構成するには、サーバー・フラグ・オプションを指定して **mkprtldap** コマンドを実行します。構文は次のとおりです。

   ```
   mkprtldap -s -a AdminDN -p Adminpasswd -w ACLBindPasswd [-f] [-d node DN]
   ```

サーバー・フラグ・オプションについては、 258 ページの『サーバー・フラグ・オプション』で詳しく説明しています。

mkprtldap コマンドは、ディレクトリー・サーバーが他の用途のために (例えば、ホワイト・ページ情報用に) セットアップされていても機能します。この場合、**mkprtldap** コマンドは AIX 情報ツリーと印刷サブツリーの情報を既存のデータベースに追加します。この印刷ツリーは、アクセス制御リスト (ACL) を使用して、他のツリーとは独立して保護されます。この場合、LDAP サーバーは通常どおり動作します。**mkprtldap** コマンドを使用して同じデータベースを共用するように System V 印刷情報を構成する前に、既存のデータベースをバックアップしてください。

-s フラグを使用した構成

構成作業中に、**-s** フラグを **mkprtldap** コマンドに使用すると、次のことが行われます。

1. システム上の IBM Directory の DB2 構成が検査されます。DB2 が IBM Directory 用に構成されていない場合、**mkprtldap** コマンドはデフォルト・インスタンス名として **ldapdb2** を使用する DB2 インスタンスを作成し、デフォルト・データベース名として **ldapdb2** を使用する DB2 データベースを作成します (存在しない場合)。既存のデータベースが検出された場合、**mkprtldap** コマンドは AIX System V 印刷情報を既存のデータベースに追加します。
2. ディレクトリーが既に構成済みならば、IBM Directory の管理者識別名 (DN) とパスワードを要求します。ディレクトリーの管理者 DN とパスワードが設定されていない場合は、**mkprtldap** コマンドがこれらをコマンドに与えられた値に設定します。
3. リブート後にサーバーが始動するように、IBM Directory Server プロセス (**slapd**) を `/etc/inittab` ファイルに追加します。
4. AIX 情報ツリー DN (cn=aixdata コンテナー・オブジェクト) をディレクトリー上に作成します (存在しない場合)。印刷サブツリーが、AIX 情報サブツリーの下に作成されます。既存の AIX 情報サブツリーがディレクトリーに存在する場合は、印刷サブツリーがその下に作成されます。 System V 印刷情報はすべて、印刷サブツリーの下に格納されます。ディレクトリー対応の System V 印刷コマンドを実行して、作成された印刷サブツリーの下にプリンターと印刷キューを追加する必要があります。
5. デフォルト・サフィックス cn=aixdata を `/etc/sldap32.conf` ファイルに追加します (サフィックスが存在しない場合)。AIX 情報ツリー・コンテナー・オブジェクト cn=aixdata を作成します (ディレクトリー内に見つからない場合)。cn=aixdata は最上位のコンテナー・オブジェクトで、その下に印刷サブツリー (ou=print) が作成されます。
6. 印刷サブツリーは、コマンドに渡される *ACLBindPasswd* パラメーターの値を基に ACL によって保護されます。System V 印刷情報の取得にディレクトリーを使用するようにクライアントを構成する際には、同じ値を使用する必要があります。

7. **-d** フラグを使用する場合に、ディレクトリー上の有効な既存ノードをコマンドに渡すと、指定したノードの下に AIX 情報サブツリーが作成されます。その後、印刷サブツリーが、AIX 情報サブツリーの下に作成されます。
8. 前述のステップがすべて完了した後、IBM Directory Server を開始します。

注: IBM Directory が既に構成済みならば、**mkprtldap** コマンドを実行するために管理者 DN とパスワードが必要です。LDAP 構成は **/etc/slapd32.conf** ファイルに保管されます。

注: IBM Directory (LDAP) サーバー構成が正常に行われない場合に、サーバー側の構成を元に戻すオプションはありません。構成中に発生するエラーについては、IBM Directory の資料を参照してください。データベース情報が **mkprtldap** コマンドによって作成された場合は、手動で除去する必要があります。**mkprtldap** コマンドによって既存のデータベースにデータが追加された場合は、セットアップの失敗からリカバリーする方法をユーザーが決める必要があります。データまたはデータベースを除去する方法について詳しくは、IBM DB2 の資料を参照してください。

System V 印刷情報サブツリー:

System V 印刷情報は印刷サブツリーの下に保管されます。さらに、このサブツリーは、ディレクトリー上のデフォルトの AIX 情報ツリー (**cn=aixdata**) に保管されます。

AIX 情報ツリーは最上位のコンテナー・オブジェクトで、その下に各種のディレクトリー対応 AIX サブシステムの情報を保管できます。印刷情報はディレクトリー内のデフォルト・ロケーションに保管することをお勧めします。ただし、**mkprtldap** コマンドは、印刷情報をディレクトリーの既存ノードの下に保管するオプションを提供しています。

次の図に、ディレクトリー情報ツリー (DIT) の形式でディレクトリーに保管される AIX System V 印刷情報を示します。

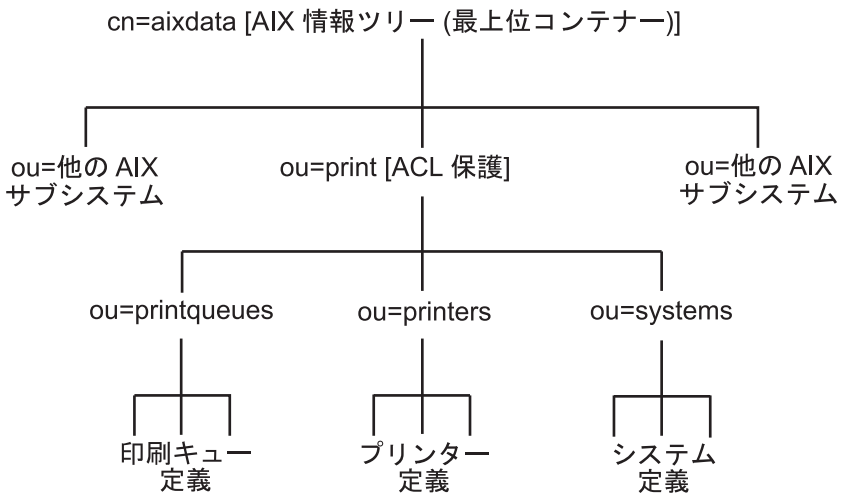

図 7. AIX System V 印刷情報の編成:

このツリー・チャートは、LDAP ディレクトリー内の System V 印刷情報の編成を示しています。AIX 情報ツリーには、オブジェクト・クラス・タイプ **container** の最上位オブジェクト **cn=aixdata** が含まれています。情報の保管に使用されるサフィックスも **cn=aixdata** です。サブシステム固有の情報は、この最上位コンテナー・オブジェクトの下に保管されます。最上位コンテナー・オブジェクトの下には、サブシステム固有の最上位オブジェクト (例えば、System V 印刷情報の場合は **ou=print**、ユーザー/グループ情報の場合は **cn=aixsecdb**) があります。System V 印刷情報は、**ou=print** オブジェクトの下に保管されます。 **ou=print** オブジェクトの下には、プリンター、印刷キュー、およびシステム情報用の 3 つのオブジェクトがあります。プリンター定義は、**ou=printer** オブジェクトの下に保管されます。印刷キュー定義は **ou=print** キュー・オブジェクトの下に保管され、システム情報は **ou=system** オブジェクトの下にあります。印刷ツリー全体は、**ou=print** オブジェクトの下で ACL によって保護されます。ツリーの ACL 保護に使用される ACL BIND パスワードは、**mkprtldap** コマンドの **-w** オプションを使用して、サーバー構成時に指定されます。

System V 印刷情報を保管するために IBM Directory を正常に構成した後、セットアップの次のステップでは、LDAP ディレクトリー・サーバーを使用するようにクライアントを構成します。

この AIX 印刷サブツリーは ACL によって保護されているので、DMT (ディレクトリー管理ツール) を使用して、AIX 情報ツリーの下のディレクトリー情報ツリー (DIT) 内にある System V 印刷情報を表示できない場合があります。クライアントが AIX 印刷サブツリーにアクセスするためには、印刷バインド DN (デフォルトは **ou=print,cn=aixdata**) と ACL BIND パスワード、または管理者 DN を使用して、バインドを行う必要があります。

System V 印刷情報の取得に IBM Directory Server を使用するクライアントの構成:

System V 印刷情報の取得に IBM Directory Server (LDAP) を使用するように、クライアントを構成できます。

System V 印刷情報の取得に IBM Directory (LDAP) を使用するようにクライアントを構成するには、次の手順で行います。

1. クライアントとしてセットアップするシステムに、IBM Directory Client ソフトウェアをインストールします。
2. クライアント・オプションを指定した **mkprtldap** コマンドを実行して、クライアントを構成します。構文は次のとおりです。

   ```
   mkprtldap -c -h DirectoryServerHostname -w ACLBindPasswd [ -d PrintBindDN ] [-U]
   ```

クライアントの構成中に、**mkprtldap** コマンドは次の処理を行います。

- IBM Directory (LDAP) サーバー・ホスト名を /etc/ldapsvc/server.print ファイルに保管します。
- AIX 印刷バインド DN を /etc/ldapsvc/server.print ファイルに保管します。
- AIX 印刷バインド DN の ACL BIND パスワードを /etc/ldapsvc/system.print ファイルに保管します。 ACL BIND パスワードの値は、ディレクトリー・サーバーの構成時に指定されたものと同じであることが必要です。
- **-U** フラグが指定された場合は、前のクライアント構成を元に戻します。このオプションは、/etc/ldapsvc/system.print ファイルと /etc/ldapsvc/server.print ファイルを、前に保管済みの /etc/ldapsvc/server.print.save ファイルと /etc/ldapsvc/system.print.save ファイルのコピーに置き換えます。

クライアント側の構成ファイル /etc/ldapsvc/server.print と /etc/ldapsvc/system.print が作成され、IBM Directory Server に関する情報と ACL 情報が保管されます。つまり、IBM Directory Server 名、ACL 情報 (*printbindDN* および *printbindPassword*)、LDAP ポート、および印刷のディレクトリー・コンテキスト (System V 印刷サブシステムが必要とする) が保管されます。デフォルトの *printbindDN* は ou=print,cn=aixdata で、**-d** フラグが指定されていない場合に使用されます。**-d** フラグに DN が指定されている場合、印刷バインド DN は ou=print,cn=aixdata, <指定された DN> です。次に、/etc/ldapsvc/server.print ファイルと /etc/ldapsvc/system.print ファイルの例を示します。

/etc/ldapsvc/server.print ファイルの例:

```
PRINTSERVER=server.ibm.com
LDAPPORT=389
PRINTBINDDN=ou=print,cn=aixdata
```

ここで

PRINTSERVER は、IBM Directory Server があるシステムのホスト名、*LDAPPORT* はバインド先のポート番号、*PRINTBINDN* は ACL バインド DN で、印刷のディレクトリー・コンテキストとしても使用されます。

/etc/ldapsvc/system.print ファイルの例:

```
PRINTBINDPASSWD=ldap
```

ここで

PRINTBINDPASSWD は印刷サブツリーの ACL BIND パスワードです。

クライアントのセットアップ時に **-w** フラグによって指定された ACL BIND パスワードは、サーバー構成時に指定された ACL BIND パスワード値と一致している必要があります。印刷ディレクトリー・コンテキストは、printbindDN と同じです。 /etc/ldapsvc/server.print ファイルのファイル・アクセス権は 644 に設定され、ファイルは root によって所有されます。/etc/ldapsvc/system.print ファイルのファイル・アクセス権は 640 に設定されます。ファイルは root によって所有され、lp グループのメンバーはこのファイルの読み取りアクセスを許可されます。

サーバー・フラグ・オプション

ここでは、フラグについて説明し、ディレクトリー対応の System V による印刷の構成例を示します。

サーバー側のオプション

フラグ	パラメーター	説明
-a	*AdminDN*	IBM Directory (LDAP) 管理者 DN を指定します。
-d	*node DN*	[詳細オプション] - このオプションを使用するには、AIX 情報ツリーと印刷サブツリーを作成する先のディレクトリー上にある有効な既存ノード DN が必要です。
-f		1 つ以上の AIX 情報ツリーがディレクトリーに存在する場合に、印刷サブツリー (および必要な場合は AIX 情報サブツリー) を強制的に作成するには、**mkprtldap** コマンドに強制フラグを指定する必要があります。
-p	*adminpasswd*	ディレクトリー (LDAP) 管理者のパスワードを指定します。
-s		このコマンドを実行して、System V による印刷のためにディレクトリーを構成することを指示します。
-w	*ACLBindPasswd*	ディレクトリー上で印刷サブツリーの ACL 保護を行うためのパスワードを指定します。

クライアント側のオプション

フラグ	パラメーター	説明
-c		このコマンドを実行して、System V 印刷情報の取得にディレクトリーを使用するようにクライアントを構成することを指示します。
-d	*PrintBindDN*	印刷バインド DN を指定します。デフォルトの印刷バインド DN は ou=print,cn=aixdata です。クライアントの構成時に使用する印刷バインド DN は、**mkprtldap** コマンドのサーバー・セットアップの終了時に表示されます。
-h	*DirectoryServerHostname*	System V 印刷情報を保管するためにセットアップされた IBM Directory Server のホスト名。
-U		前に行ったクライアントの構成を元に戻します。

使用法

フラグ	パラメーター	説明
?		**mkprtldap** コマンドの使用法に関する情報を表示します。

System V による印刷の例:

よくあるシナリオの例を示します。

1. 管理者 DN cn=root と root パスワードを使用して、System V による印刷のために IBM Directory の新規インストールを構成するには、次のように入力します。

   ```
   mkprtldap -s -a cn=root -p root -w aclBindpassword
   ```

   ここで、*aclBindpassword* は、ACL による印刷サブツリーの保護のために使用されるパスワードです。ACL BIND パスワードは、ディレクトリー上での System V 印刷の構成時に指定されます。この構成により、ディレクトリー管理者の DN と、**cn=root** および root のパスワードが設定されます。このコマンドを実行すると、サフィックスと、最上位コンテナー・オブジェクト **cn=aixdata** がセットアップされます。印刷サブツリー (**ou=print**) は、この AIX 情報ツリー (**cn=aixdata** オブジェクト) の下に作成されます。
2. 構成済みの IBM Directory Server があるマシン上で System V 印刷を構成するには、管理者 DN とパスワードが必要です。例えば、既存の管理者の DN とパスワードが **cn=admin** と **passwd** ならば、次のように入力します。

   ```
   mkprtldap -s -a cn=admin -p passwd -w pass123wd
   ```
3. IBM Directory Server の既存ノード (例: **o=ibm,c=us**) の下に System V 印刷を構成するために、**mkprtldap** コマンドには **-d** フラグ・オプションが用意されています。これは詳細オプションで、ディ

レクトリーの既存ノードの下に印刷情報を保管する必要がある場合のみ推奨されます。推奨されるオプションは、**-d** オプションを指定せずに、ディレクトリーのデフォルト・ロケーションに印刷サブツリーを保管することです。

ディレクトリー上で System V 印刷を構成するには、管理者 DN とパスワードが必要です。例えば、既存の管理者の DN とパスワードが **cn=admin** と **passwd** であるとします。次のように入力します。

```
mkprtldap -a cn=admin -p passwd -w acl123passwd -d o=ibm,c=us
```

このコマンドを実行すると、AIX 情報ツリー (**cn=aixdata**) が **o=ibm,c=us** オブジェクトの下に作成されます。この新規オブジェクト (**cn=aixdata, o=ibm, c=us**) の下に、印刷サブツリーが作成されます。

4. 他のサブシステム固有の情報に関する既存の AIX 情報ツリーがディレクトリーが含まれている場合に、System V 印刷情報を別個の AIX 情報ツリーの下に構成します。セキュリティー情報または NIS 情報を保管する既存の AIX 情報ツリーが、ディレクトリーに含まれている場合があります。ディレクトリー上で、別の AIX 情報ツリーの下にある分離したロケーションに、印刷情報を保管する必要が生じることがあります。デフォルトでは、ディレクトリーに既に情報ツリーが存在している場合に、**mkprtldap** コマンドは AIX 情報ツリーを作成しません。印刷情報を保管するための AIX 情報ツリーの作成を **mkprtldap** コマンドに強制するには、 **-f** フラグを使用します。

   セキュリティー情報と NIS サブシステム情報が、**cn=aixdata,o=ibm,c=us** の AIX 情報ツリーの下に保管されている例を考えます。印刷情報用に、既存のものと異なる新規の AIX 情報ツリーを作成するには、**-f** フラグを付けてこのコマンドを実行し、デフォルト・ロケーションまたは別のノードを指定します。

   ディレクトリー上で System V 印刷を構成するには、管理者 DN とパスワードが必要です。例えば、既存の管理者の DN とパスワードが **cn=admin** と **passwd** ならば、次のように入力します。

   ```
   mkprtldap -a cn=admin -p passwd -w passwd123 -f
   ```

   このコマンドを実行すると、AIX 情報ツリー (**cn=aixdata**) がサフィックス (**cn=aixdata**) を指定して作成され、この新規 AIX 情報ツリー (ou=print, cn=aixdata) の下に印刷情報が保管されます。この例では、ディレクトリーに 2 つの AIX 情報ツリー **cn=aixdata,o=ibm,c=us** と **cn=aixdata** が存在します。印刷情報は、**cn=aixdata** オブジェクトの下にあります (サフィックス **cn=aixdata**)。 **mkprtldap** コマンドが印刷情報をディレクトリーに追加する先としては、デフォルト・ロケーションを使用することをお勧めします。

5. ホスト **server.ibm.com** 上で System V による印刷のためにセットアップされた IBM Directory を使用するようにクライアントを構成するには、次のように入力します。

   ```
   mkprtldap -c -h server.ibm.com -w passwd
   ```

   ACL BIND パスワード (passwd) が、ディレクトリー・サーバーのセットアップ時に指定されたものと同じであることを確認してください。**-d** オプションを使用して印刷バインド DN 値を指定せずにコマンドを実行すると、このコマンドはデフォルトの印刷バインド DN **ou=print,cn=aixdata** を使用します。印刷バインド DN は、サーバーの構成時に **mkprtldap** コマンドの実行終了結果として表示されるものと一致している必要があります。

6. クライアント側の構成ファイルの情報を変更するには、新しい情報を指定して **mkprtldap** コマンドを実行します。次のように入力します。

   ```
   mkprtldap -c -h server.ibm.co.uk -w aclpasswd -d ou=print,cn=aixdata,c=uk
   ```

   既に構成済みのクライアントに対してこのコマンドを実行すると、/etc/ldapsvc/server.print ファイルと /etc/ldapsvc/system.print ファイルの情報が変更され、新しい構成情報が格納されま

す。/etc/ldapsvc/server.print ファイルと /etc/ldapsvc/system.print ファイルの元の内容は、/etc/ldapsvc/server.print.save ファイルと /etc/ldapsvc/system.print.save ファイルに保管されます。

サーバーおよびクライアントの構成時にアクセスされるファイル:

mkprtldap コマンドは、サーバーおよびクライアントの構成時にいくつかのファイルにアクセスし、これらのファイルを変更します。

次の表に、サーバーおよびクライアントの構成時に **mkprtldap** コマンドによってアクセスされ、変更されるファイルと、その説明を示します。また、ファイルの内容も示します。

モード	ファイル	説明
rw	/etc/slapd32.conf	サーバー・セットアップ - IBM Directory (LDAP) 構成情報を格納します。
rw	/etc/ldapsvc/server.print	クライアント構成 - System V 印刷情報 (マシン名、ディレクトリー上の印刷サブツリーの場所、および LDAP ポート) を保管するために構成されたディレクトリー・サーバーに関する情報を格納します。
rw	/etc/ldapsvc/system.print	クライアント構成 - ディレクトリー上の印刷サブツリーの ACL BIND パスワードを格納します。

基本オペレーティング・システム・スプーラーのトラブルシューティング

スプーラー内を移動するスプーラー・ジョブを追跡することによって、基本オペレーティング・システム・スプーラーのトラブルシューティングを行うことができます。基本オペレーティング・システム・スプーラーに対して実行依頼されたジョブは、予測可能な方法でスプーラー・コンポーネント間を移動します。この移動は、スプーラー・キューの構成、特にスプーラー・キューのバックエンドに完全に依存しています。

注: スプーラーの本格的なトラブルシューティングを行うには、root 権限が必要です。 root 権限を持たないユーザーが行う操作は、次のものに制限されます。

- ジョブをスプーラーに実行依頼する
- /dev ディレクトリー内のデバイス・ドライバーの入り口点にデータを直接送信する
- スプーラー・キューの状況を照会する
- ユーザーが所有するスプーラー・ジョブの状況を変更する (取り消しを含む)

注: このトラブルシューティング情報は、シェル・プロンプトへのアクセスが可能であることを前提としています。基本オペレーティング・システム・スプーラー自体のフロントエンドがいくつか市販されています。この環境でのトラブルシューティングも十分可能ですが、スプーラーにジョブを実際に実行依頼するために使用されたコマンドやメソッドに問題がある場合には、スプーラーにジョブを実行依頼するために使用されたコマンドやメソッドを正確に判別する手段を備えたアプリケーションが必要です。

ローカル・プリンターのトラブルシューティング

ローカル・プリンターに問題がある場合は、次の項目を確認してください。

- **qdaemon** が実行されていることを確認します。**qdaemon** から fork されたプロセスが実行されていないことを確認します。
- システム日付が正しいことを確認します。qconfig ファイルが変更されると、**qdaemon** は qconfig.bin ファイルを自動的に再作成します。qconfig ファイルの日付が qconfig.bin ファイルの日付より古い場合、qconfig ファイルは変更された直後であっても要約されません。
- qconfig.bin ファイルと qconfig ファイルの日付が正しく、qconfig ファイルの変更内容が正しい場合は、/etc/qconfig ファイルが /usr/lpd/qconfig ファイルにリンクされなくなっています。

- /tmp ディレクトリーが満杯でないことを確認してください。「**No Virtual Printers Defined**」などのメッセージが出された場合は、/tmp ディレクトリーが満杯になっている可能性があります。
- root 以外のユーザーが印刷できない場合は、/tmp ディレクトリーのアクセス権を確認します。また、使用されている印刷コマンドのアクセス権を確認します (**enq** を含む)。
- /var/spool/lpd/qdir ファイル内に使われなくなったキュー名がないかどうか確認します。/etc/qconfig ファイルを新規にインストールした場合に、キューが新規の /etc/qconfig ファイルから除去され、使用されなくなったキュー名を使用して印刷要求が行われると、問題が起こります。**qdaemon** は、エラー・メッセージをログに記録します。メッセージが古いキューを指しているかどうか調べてください。該当する場合は、使用されなくなったキュー項目を /var/spool/lpd/qdir ファイルから除去するまで問題が起こります。
- 印刷コマンドによって要求されるオペレーター・アテンション・メッセージを受信していない場合は、ソケットが接続されていて、**ping** コマンドを使用してホスト名を ping できることを確認してください。
- 印刷コマンドからのオペレーター・アテンション・メッセージは、TCP/IP サブシステムの **writesrv** コマンドによって送付されます。メッセージを受信していない場合は、次のコマンドを実行して、**writesrv** コマンドが実行されているかどうか検査してください。

  ```
  lssrc -s writesrv
  ```

  writesrv コマンドが実行されていない場合は、次のコマンドを使用して開始します。

  ```
  startsrc -s writesrv
  ```

  最後に、次に示すいずれかのコマンドの出力に **writesrv** がリストされているかどうか確認します。

  ```
  netstat -a | pg
  ```

  または

  ```
  netstat -a | grep writesrv
  ```

注: AIX オペレーティング・システムは、USB (Universal Serial Bus) を経由でシステムに接続するプリンターをサポートしていません。

操作不能のプリンターのトラブルシューティング

ローカル接続プリンターがまったく機能しない場合は、以下のトラブルシューティングの手順を使用してください。

操作不可のプリンターがある場合は、次の項目を確認してください。

- 電源ケーブルのみがプリンターに接続されている状態で、プリンターのテスト・パターンを実行します。
- プリンター用の正しいケーブルを使用していることを確認します。
- ケーブルがしっかりと接続されていることを確認します。
- プリンター用のデバイスが作成済みであることを確認します (「Devices (デバイス)」、SMIT、またはコマンド・ラインで)。
- リブートの直後、またはリブート後にまだプリンターにデータを送信していないときに、次のコマンドを試行します。

  ```
  echo Does the printer work? > /dev/lpn
  ```

ここで、*lpn* はテストしているプリンター・デバイスの名前です。そのメッセージがプリンターで印刷された場合は、プリンターの仮想プリンター定義をセットアップします。ステートメントがハングするかエラー・メッセージを戻す場合、オペレーティング・システムやキューイング・システムに問題はありません。次のうち 1 つ以上に問題があります。

  - ケーブル。
  - ボー・レート、ハンドシェーク、ポート番号などの設定。プリンターとコンピューターは同じ設定値を使用している必要があります。
  - コンピューターのポートが正しくない。
  - プリンターの故障。
- 8 ポート、16 ポート、または 64 ポートのアダプター上、またはモデム上でシリアル・プリンターを動作させる際に問題が生じる場合は、コンピューターの S1 または S2 上でプリンターを直接動作させることを試みてください。プリンターが S1 または S2 上で動作することを確認した後、プリンターを目的のポートに移動します。S1 と S2 が使用できない場合は、プリンターを別のポートに移動することを試みます。

リモート・プリンターのトラブルシューティング

この情報は、リモート・プリンターのトラブルシューティングに使用できます。

リモート印刷サーバーとして動作するホストについては、次の項目を確認してください。

- すべてのクライアント・マシン (外部ホスト) が `/etc/hosts.lpd` ファイルにリストされていることを確認します。
- TCP/IP サブシステムが実行されていることを確認します。
- `/usr/spool/lpd` ディレクトリーが存在することを確認します。
- **lpd** デーモンが実行されていない場合は、`/etc/locks/lpd` ディレクトリーが存在しないことを確認します。
- **lpd** デーモンと **qdaemon** が両方とも実行されていることを確認します。
- 261 ページの『ローカル・プリンターのトラブルシューティング』を確認します。

リモート印刷サーバーに対して印刷を行うホストについては、次の項目を確認してください。

- リモート印刷サーバーのキュー名とサーバー名が、`/etc/qconfig` ファイル内で正しく指定されていることを確認します。
- TCP/IP サブシステムが実行されていることを確認します。
- **qdaemon** デーモンが実行されていることを確認します。

アダプターに関する考慮事項

16 ポートの RS-232 アダプターは送信可 (CTS) をサポートしないので、ジョブの印刷中にプリンターの電源がオフになった場合、このアダプターに接続したプリンターはジョブの印刷を完了しません。

16 ポートの RS-232 アダプターは送信可 (CTS) をサポートしません。ジョブの印刷中にプリンターの電源がオフになった場合、このアダプターに接続したプリンターはジョブの印刷を完了しません。手動でジョブを再開するか、ジョブを削除する必要があります。

リソースに関する考慮事項

印刷を行うとプロセスが生成されます。ほとんどの場合、ジョブの印刷には 1 つから 5 つのプロセスが必要です。

他のアクティビティーと同様に、システム上のプロセス数がいっぱいになる可能性があります。これは、非常に活発に使用されているシステム上で単一の印刷ジョブを実行依頼して発生することもあれば、他のアクティビティーがあまり行われていないシステム上で多数のジョブを実行依頼して発生することもあります。

プロセス数が多すぎると、システムが誤った動作をする可能性があります。システムの動作に異常がある場合は、リソースを検査して、プロセス数が多すぎないかどうか調べてください。

var ファイルシステムがいっぱいになったときに発生する印刷問題の解決

/var ファイルシステムがいっぱいになると、印刷の問題が発生します。

/var ファイルシステムがいっぱいになると、印刷の問題が発生します。この問題は通常、印刷キューに送信された印刷ジョブが何らかの理由でバックアップを開始し、ファイルシステム内のスプーリング・ディレクトリーが大きくなりすぎた場合に起こります。通常影響を受けるスプーリング・ディレクトリーは、/var/spool/lpd と /var/spool/qdaemon です。

キュー・デーモンが機能を停止した、プリンターが停止した、プリンターの電源がオフになった、またはプリンターに送信された大きな印刷ジョブがリソースをすべて占有した場合に、印刷キューはバックアップを行う可能性があります。スプーリング・ディレクトリー以外にも、ファイルシステム内の他のディレクトリーが大きくなりすぎた場合に、/var ファイルシステムがいっぱいになることもあります。

/var ファイルシステムがいっぱいになった場合は、次のいずれかの作業を実行してください。

キュー・デーモンの再活動化

キュー・デーモン (または **qdaemon**) プロセスは、印刷ジョブ要求と、これらの要求を処理できるプリンターを追跡します。

qdaemon は、未処理の要求のキューを維持し、デバイスが使用可能になると、適切な時点で適切なデバイスに要求を送ります。**qdaemon** が機能を停止した場合は、印刷の問題が発生するので、次の手順を使用して **qdaemon** を再始動する必要が生じます。

注: 一部のコマンドには、root ユーザーまたはシステム・グループの権限が必要です。

1. 次の **ps** コマンドを入力して、**qdaemon** が機能を停止したかどうか判別します。

   ```
   ps -ef | grep qdaemon
   ```

   /var/sbin/qdaemon、**qdaemon**、または **/etc/qdaemon** という名前の実行中のプロセスが見つからない場合は、**qdaemon** が実行されていません。
2. 次の **startsrc** コマンドを入力して、**qdaemon** を再始動します。

   ```
   startsrc -s qdaemon
   ```

   システム・リソース・コントローラー (SRC) を使用していない場合は、**qdaemon** コマンドを使用してキュー・デーモンを再始動することもできます。
3. 印刷キュー内のすべてのジョブを **qdaemon** が印刷するまで待ちます。
4. 次のように入力して、**lpd** デーモンを稼働させます。

   ```
   startsrc -s lpd
   ```

   lpd デーモンは、ネットワーク上でリモート印刷サーバーを提供します。

印刷キュー・バックログのクリア

印刷キュー・バックログをクリアする必要がある場合は、**qdaemon** を停止し、/var ファイルシステムがいっぱいになっているかどうか確認する必要があります。

root としてログインする必要があります。

/var ファイルシステムがいっぱいになった場合は、次の手順を使用してキュー・ディレクトリーをクリアし、**qdaemon** を再始動します。

1. 可能な場合は、すべての現行印刷ジョブが印刷を完了するまで待つか、印刷ジョブを取り消します。印刷ジョブを取り消すには、**lpstat** コマンドを発行して印刷ジョブ番号を取得してから、**enq** コマンドを使用して印刷ジョブを取り消します。

   ```
   enq -x JobNumber
   ```

   lpstat コマンドは、ライン・プリンターの現在の状況に関する情報を表示します。**enq** コマンドは、ファイルをキューに入れます。
2. 次のコマンドを発行して **qdaemon** を停止します。

   ```
   stopsrc -s qdaemon
   ```
3. 次のコマンドを発行して、**qdaemon** が他のプロセスを fork していないことを確認します。

   ```
   ps -ef | grep qdaemon
   ps -ef | grep pio
   ```

   ps コマンドは、プロセスの現在の状況を表示します。**grep** コマンドは、パターンと一致する個所があるかどうかファイルを検索します。

   前に示した **grep** コマンドからそれぞれ 1 行ずつが戻された場合は、ステップ 4 を飛ばしてステップ 5 に進みます。複数の行が戻された場合は、ステップ 5 に進みます。
4. **ps -ef** コマンドから他の qdaemon または pio が戻された場合は、それぞれのプロセス ID を指定して次のコマンドを発行することにより、これらのプロセスを強制終了します。

   ```
   kill -9 pid
   ```

   次の例は、**ps -ef** から戻された qdaemon を示しています。プロセス ID は 3357 です。

   ```
   root 3357 2288 0 13:32:21 - 0:04 dtterm
   ```

   このプロセス ID を強制終了するには、コマンド・ラインで kill -9 3357 を入力します。
5. このステップは、現行印刷ジョブが削除されないようにする必要がある場合のみ実行します。それ以外の場合は、ステップ 7 に進みます。

   印刷ジョブが次のいずれかのディレクトリー内でキューに入れられている場合は、そのコピーを作成し、/tmp に配置します。キューイング・システムが実行を再開したときに、このジョブを印刷できます。

   ```
   /var/spool/qdaemon
   /var/spool/lpd
   ```

   注: これらのディレクトリー内では、ファイルには分かりづらいシステム名が付きます。
6. /var ファイルシステムがいっぱいになると、**qdaemon** またはスプーラーに問題が生じることがあります。大きな印刷ジョブが失敗したり、ゼロの長さをもつ 00root ファイルが qdir ディレクトリーに作成されたりする場合があります。この場合にシステムをリブートしても、ファイルが消去されなかったり、**qdaemon** が再始動しなかったりする可能性があります。

df コマンドを入力し、/var の **%used** 列を調べて、ファイルシステムがいっぱいになっていないかどうか確認します。必要に応じて、ファイルシステムのスペースを空けます。

df コマンドは、ファイルシステムの合計スペースと使用可能なスペースに関する情報を表示します。

7. 次のように入力して、ディレクトリーを移動します。

   ```
   cd /var/spool/lpd/qdir
   ```

8. **pwd** コマンドを発行して、現行ディレクトリーが正しいことを確認します。次に、**rm** コマンドを使用して、このディレクトリー内のファイルをすべて除去します。

   ```
   rm *
   ```

   pwd コマンドは、現行ディレクトリーの絶対パス名 (ルート・ディレクトリーからの) を標準出力に書き込みます。**rm** コマンドは、指定したファイルのエントリーをディレクトリーから除去します。

9. 次のように入力して、再びディレクトリーを移動します。

   ```
   cd /var/spool/lpd/stat
   ```

10. **pwd** コマンドを発行して、現行ディレクトリーが正しいことを確認します。次に、このディレクトリー内のファイルをすべて除去します。

    ```
    rm *
    ```

11. 次のように入力して、再びディレクトリーを移動します。

    ```
    cd /var/spool/qdaemon
    ```

12. **pwd** コマンドを発行して、現行ディレクトリーが正しいことを確認します。次に、このディレクトリー内のファイルをすべて除去します。

    ```
    rm *
    ```

13. リモート・キューまたは **lpd** に問題がある場合は、このステップを行います。次のように入力して、ディレクトリーを移動します。

    ```
    cd /var/spool/lpd
    ```

    pwd を発行して、現行ディレクトリーが正しいことを確認します。次に、**rm** コマンドを使用して、このディレクトリー内のファイルをすべて除去します。

    ```
    rm *
    ```

    注: rm コマンドは、サブディレクトリーを除去しません。

14. **qdaemon** を開始します。

    ```
    startsrc -s qdaemon
    ```

    キューイング・システムは正常に始動します。一部のキューがまだ停止している場合は、次のように入力して起動します。

    ```
    enable QueueName
    ```

プリンター・リソースの再割り当て

この手順を使用して、1 つの印刷ジョブがすべてのプリンター・リソースを使用しないようにすることができます。

注: 一部のコマンドには、root ユーザーまたはシステム・グループの権限が必要です。

1. 次に示す 2 つの方法のどちらかによって、印刷ジョブがすべてのリソースを使用しているかどうか判別します。
   - 次の **lpq** コマンドを使用します。

```
lpq
```

lpq コマンドは、フラグを付けずに入力されると、デフォルト・キューの状況を報告します。

- 次の **enq** コマンドを使用します。

  ```
  enq -q
  ```

  enq コマンドは、ファイルを共用リソース (通常はプリンター) へのキューに入れます (つまり、ファイルを特定のリソースへのキューに入れます)。 **-q** フラグは、デフォルト・キューの状況を表示します。

2. 次のいずれかのコマンドを使用して、印刷キューからジョブを除去します (所有するもの以外のジョブを取り消すには、root ユーザー権限が必要です)。

   - 次の **enq** コマンドを使用します。

     ```
     enq -x 21
     ```

     この例では、**enq** コマンドは **-x** フラグを使用してジョブ番号 **21** を取り消します。

   - 次の **lprm** コマンドを使用します。

     ```
     lprm -P lp0 42
     ```

     この例では、**lprm** コマンドは **lp0** プリンター・キュー (**-P** フラグによって指定された) からジョブ番号 **42** を除去します。また、コマンド・ラインでユーザー名を指定して、特定のユーザーを対象にジョブを除去することもできます。

   - 次の **qadm** コマンドを使用します。

     ```
     qadm -X lp0
     ```

     この例では、**qadm** コマンドは **-x** フラグを使用して、**lp0** プリンター上のジョブをすべて取り消します。

   - 次に示す **qcan** コマンドの SMIT 高速パスを使用します。

     ```
     smit qcan
     ```

     この例では、「**By Print Queue (印刷キュー別)**」オプションを選択して、特定のユーザーのジョブをすべて取り消すか、特定のプリンター上のジョブをすべて取り消すことができます。

3. まず次の **split** コマンドを使用して印刷ジョブを小さな部分に分割し、次にファイルを一連のジョブとして送信するように、印刷ジョブの送信者に指示します。

   ```
   split -50 bigfile
   ```

   split コマンドは指定されたファイルを読み取り、セグメントに分けて一連の出力ファイルに書き込みます。前の例では、`bigfile` は `bigfileaa`、`bigfileab`、`bigfileac` などの名前が付いた 50 行のセグメントに分割されます。

不要なディレクトリー・ファイルの削除

スプーリング・ディレクトリーにある不要なファイルを削除できます。

これらの一部のコマンドには、root ユーザーまたはシステム・グループの権限が必要です。

1. 次の **du** コマンドを入力して、不要なファイルがスプーリング・ディレクトリーに保管されているかどうか判別します。

   ```
   du -rs /var/spool
   ```

du コマンドは、ディスクの使用量を要約します。**-s** フラグは、/var/spool ディレクトリーとそのディレクトリーに含まれるファイルの合計ディスク使用量のみを表示するように、**du** コマンドに指示します。**-r** フラグは、ファイルまたはディレクトリーを読み取ることができない場合にエラー・メッセージを表示するように、**du** コマンドに指示します。

2. 次のいずれかを行って、いっぱいになったディレクトリーにあるファイルを削除または移動します。
   - 不要なファイルを削除します。例えば、次のようにします。

     ```
     rm extrafile
     ```
   - 数時間前のファイルを安全な一時ディレクトリーに移動します。例えば、次のようにします。

     ```
     mv extrafile /u/spoolhold
     ```

     注: 所有するもの以外のファイルを除去または移動するには、root ユーザー権限が必要です。
3. 次のようにして、ユーザーがスプーリング・ディレクトリー内にファイルを保管しないようにします。
   - **chmod** コマンドを使用して、スプーリング・ディレクトリーにアクセス権を設定します。一般ユーザーを除外するようにこのディレクトリーを変更します。例えば、次のようにします。

     ```
     chmod go-rw /var/spool/lp0
     ```
   - ディレクトリーを空にする **cron** ジョブを作成します (root ユーザー権限が必要です)。crontab ファイルを編集します。例えば、crontab ファイルに次の行を追加します。

     ```
     find /spool -mtime +7 -a -exec rm -f
     ```

     この行により、**/var/spool** ディレクトリー内のファイルが最終変更から 1 週間後に除去されます。
   - ユーザー・グループ全体のポリシーを設定します。

     ディスクの所有量が特定のしきい値を超えるユーザーをすべて識別し、これらのユーザーにファイルのクリーンアップを要求する E メールを送信するように、スクリプトを作成します。
   - ユーザーが使用頻度の低いファイルを保存できるように、例えば共通領域にテープ・ドライブを配置するなど、代替のファイル保管手段を提供します。
4. 最後の手段として、次のどちらかの方式を使用して、スプール・ディレクトリーに追加のスペースをマウントします。
   - **mount** コマンドを使用します。このコマンドは、ファイルシステムを指定した位置で使用可能にします。例えば、次のようにします。

     ```
     mount /var/spool morespool
     ```
   - **smit mount** コマンドを使用し、「**Mount a File System (ファイルシステムのマウント)**」オプションを選択して、ファイルシステム名と属性を指定します。

端末接続プリンターのトラブルシューティング

ASCII 端末に接続されたプリンターが出力を生成しない場合には、次の項目を確認してください。

- 端末装置の **AUX** ポートが、ご使用のプリンターと同じ設定値を使用して構成されていることを確認します。このためには、AUX ポートの値の設定について、ご使用の端末装置の資料を参照してください。プリンターのシリアル・インターフェースの構成については、ご使用のプリンターの資料を参照してください。関係のある値には、ボー・レート、パリティー、データ・ビット、ストップ・ビット、XON/XOFF などがあります。
- ご使用の端末装置が別のタイプの端末装置をエミュレートしている場合は、**PIOTERM** 環境変数を設定する必要が生じることがあります。

  ```
  export PIOTERM=TerminalTypeEmulated
  ```
- プリンター用の正しいケーブルを使用していることを確認します。

- 端末装置の補助ポートにケーブルがしっかりと接続されていることを確認します。
- 次のように入力して、印刷キューが「READY (作動可能)」であることを確認します。

  ```
  lpstat
  ```

端末接続プリンター・キューの状況が「READY (作動可能)」と表示されない場合は、次のコマンドを入力してキュー上のジョブをすべて取り消し、キューを再始動します。

```
qadm -Xqname
qadm -Uqname
```

ここで、*qname* は端末接続プリンター・キューの名前です。印刷ジョブを再実行依頼する必要があります。

- **pioout** コマンドに正しいアクセス権があることを確認します。

  ```
  /usr/lib/lpd/pio/etc/pioout -r-sr-xr-x
  ```

  アクセス権を再設定するには、次のコマンドを入力します。

  ```
  chmod 4555 /usr/lib/lpd/pio/etc/pioout
  ```
- 261 ページの『ローカル・プリンターのトラブルシューティング』を確認します。
- プリンター制御コードが端末装置の制御コードと競合する場合があります。ここまでのチェックリスト項目に従っても出力が生成されない場合は、仮想プリンターを ASCII プリンターとして再構成します。25 ページの『仮想プリンターと印刷キューの構成』を参照してください。

キーボード入力のエコーがプリンター出力と混合している場合は、次のことを確認してください。

- 端末接続プリンターに固有の仮想プリンター属性を調整します。このためには、次の SMIT 高速パス・コマンドを使用します。

  ```
  smit chvirprt
  ```
- 印刷要求を再実行依頼し、要求の印刷中はタイプ入力を行わないようにします。
- ASCII 端末がロックしている場合は、端末の電源をオフにしてからオンにします。

7 ビット・インターフェースに接続された 8 ビット・プリンターに関する考慮事項

7 ビット・インターフェースに接続して 8 ビット・プリンターを使用する際には、いくつかの考慮事項があります。

一部のプリンターは、ホストへのインターフェースが 8 ビット (1 バイトあたり 8 ビット) であることを前提としています。8 ビット・プリンターを 7 ビット・インターフェースに接続すると、印刷は可能でも印刷出力が受け入れられない場合があります。ご使用のプリンターが 8 ビット・インターフェースを前提としているかどうか調べるには、プリンターのマニュアルを参照してください。

次の状況では、誤った印刷出力が生成される可能性があります。

- プリンター・コマンド・シーケンスに 8 ビット値が含まれていることがあります。

  8 ビット・プリンターを 7 ビット・インターフェースに接続する必要がある場合は、印刷出力の誤りを防止するために、次の手順で行ってください。

  1. SMIT 高速パス **smit lsvirprt** を入力します。
  2. 印刷キューを選択し、次のように入力します。

     ```
     j=!j=!
     ```

3. Enter キーを押して終了します。

   これにより、印刷ファイル初期化ストリング (8 ビットのコマンド・シーケンスを含んでいる可能性がある) がプリンターに送信されないようになります。

   注: プリンターの初期化もこの手順によってバイパスされます。このため、直前の印刷ファイルが残したピッチ、行送りなどの属性によっては、出力が正しく印刷されない場合があります。

- プリンターの文字コード・ポイントが 8 ビット値で、それぞれの図形文字が 8 ビットの整数値によって表現されている場合は、これが原因で誤った文字が印刷されることがあります。この問題を避けるには、印刷ファイル内の文字をすべてポータブル ASCII 文字セット内の文字にする必要があります。
- 7 ビット・インターフェースの使用時には、印刷されるグラフィック・ファイルに影響が及びます。これは、データ・ポイントの一部が失われるからです。

キュー・デーモンのトラブルシューティング

qdaemon コマンドに問題が発生した場合は、このトラブルシューティングの手順を使用してください。

通常の環境では、**qdaemon** コマンドはシステムの始動時に開始されて、システムがシャットダウンするまで実行され、ユーザーがこのコマンドについて意識する必要はありません (詳しくは、**qdaemon** コマンドを参照してください)。ただし、場合によっては、**qdaemon** コマンドが実行を停止したり、機能を実行できなくなったりすることがあります。次に、このような状況で行う必要がある作業について説明します。

次のいずれかの状態になった場合は、**qdaemon** コマンドの保守が必要であることを示しています。

- **enq** コマンドを要求すると、次のメッセージが戻される。

  ```
  cannot awaken qdaemon (request accepted anyway)
  ```

- **qdaemon** コマンド自体の内部で重大な不整合が検出され、エラー・メッセージが表示される。
- **ps -ef** コマンド (すべてのプロセスの完全リストを表示するプロセス状況コマンド) が、**/usr/sbin/qdaemon** または **qdaemon** という名前のプロセスを表示しない。

qdaemon コマンドを開始するには、次のコマンドを発行します。

```
startsrc -s qdaemon
```

一般に、root 特権を持つユーザーのみがこのコマンドを使用できます。新規の **qdaemon** コマンドが、初期化プロセスを実行します。

qdaemon コマンドが実行を継続しない場合は、**qdaemon** コマンドと **enq** コマンドの両方に適切なアクセス権があることを確認してください。root 権限を持つユーザーは、**qdaemon** コマンドと **enq** コマンドを両方とも所有します。 **qdaemon** コマンドと **enq** コマンドは、これらのコマンドを所有するユーザーが実行した場合と同じように実行されることが必要です。許可ビット **s** は、ファイルの所有者がプロセスの実効所有者 (ユーザー ID) になるように設定します。これら 2 つのコマンドに適切なアクセス権は、次のとおりです。

項目	説明
qdaemon -r-sr-s—-	これらのアクセス権を確認するには、aclget /usr/sbin/qdaemon を入力します。 アクセス権をリセットするには、tcbck -y /usr/sbin/qdaemon を入力します。これらのアクセス権をリセットするには、root ユーザー権限が必要です。
enq -r-sr-sr-x	これらのアクセス権を確認するには、aclget /usr/bin/enq を入力します。 アクセス権をリセットするには、tcbck -y /usr/bin/enq を入力します。これらのアクセス権をリセットするには、root ユーザー権限が必要です。

qdaemon コマンドに関する問題が引き続き発生する場合は、次の手順を使用してキューイング・システム全体を再初期化できます。

1. **qdaemon** コマンドが実行されている場合は (このことを調べるには **ps -ef** コマンドを使用します)、stopsrc -s qdaemon を入力して終了します。
2. バックエンドが実行されている場合は、**kill** コマンドを使用して停止します。詳しくは、**kill** コマンドを参照してください。
3. 次のディレクトリーの内容を削除します。
   - /var/spool/lpd/stat
   - /var/spool/lpd/qdir

   注: 印刷のためにキューに入れられているすべてのジョブが取り消され、これらのジョブを再実行依頼する必要があります。
4. startsrc -s qdaemon と入力して、**qdaemon** コマンドを再始動します。詳しくは、**startsrc** コマンドを参照してください。

キューイング・システムの問題

プリンター上でキューが待機していないことが確認されると、キューイング・システムは、1 つ以上のキューが **DEV_WAIT** 状況であることを示します。プリンターがオフラインになっているか、用紙切れまたはジャム、ケーブルのゆるみ、不良、あるいは配線の誤りが起きているために、キューが待機しない場合があり、そのキューは、TIMEOUT 期間内に **DOWN** 状況に変わりません。このシナリオでは、以下の方式でキューイング・システムをクリアして再始動してください。

この方式では、**qdaemon** を停止し、キューに入れられたすべてのジョブを除去し、**qdaemon** を再始動します。root 権限が必要です。

qdaemon デーモンを停止するには、以下のコマンドを使用します。

```
stopsrc -s qdaemon
ps -e | fgrep qd
kill -9 PIDNumbers
```

ここで、*PIDNumbers* は、**ps**コマンドから得られる PID です。

```
ps -e | fgrep pio
kill -9 PIDNumbers
rm /var/spool/lpd/stat/_dev_DEVICE
```

ここで、*DEVICE* は、DEV_WAIT 状況を示しているデバイスです。

```
rm /var/spool/lpd/stat/s.QUEUE.DEVICE
```

ここで、*QUEUE* は問題のキューであり、*DEVICE* は、DEV_WAIT 状況を示しているデバイスです。

```
mkdir /tmp/QDIR
mv /var/spool/lpd/qdir/NNUSER:QUEUE /tmp/QDIR
```

ここで、*NN* は番号、*USER* はジョブをキューに入れたユーザー、*QUEUE* は DEV_WAIT 状況を示しているキューです。

qdaemon デーモンを開始するには、以下のコマンドを使用します。

```
startsrc -s qdaemon
```

キューイング・システムがクリアされて正しく機能したら、**qdaemon** を停止し、jdf ファイルを /tmp/QDIR パスから /var/spool/lpd/qdir パスにコピーした後で、**qdaemon** を再始動する必要があります。

qdaemon のテスト

ジョブをスプーラーに実行依頼してもスプーラーのアクティビティーに違いが認められない場合は、次の情報を使用して問題を判別し、解決してください。

asc という名前のローカル印刷キューがあるとします。

qdaemon は実行されていますか?

コマンド enq -Pasc /etc/motd を発行します。 **qdaemon** がアクティブでない場合は、次のような種類のメッセージが表示されます。

```
enq: (WARNING): Cannot awaken qdaemon. (request accepted anyway)
enq: errno = 2:  No such file or directory
enq: (WARNING): Cannot awaken qdaemon. (request accepted anyway)
enq: errno = 2:  No such file or directory
```

コマンド ps -ef | grep qdaemon を発行して、**qdaemon** が非アクティブであることを確認します。**qdaemon** が非アクティブの場合は、**grep** 自体を表す出力行のみが表示されます。これは次のような行です。

```
root  2992 18792   0 12:46:39  pts/2  0:00 grep qdaemon
```

qdaemon がアクティブならば (アクティブでないことはほぼ確実ですが)、次のような種類の行が表示されます。

```
root  2980 3652    0 12:41:25   -   0:00 /usr/sbin/qdaemon
```

qdaemon が非アクティブの場合は、コマンド startsrc -s qdaemon を発行して **qdaemon** を再始動します。**qdaemon** が非活動状態になった場合、本来は **srcmstr** プロセスによって自動的に再始動されますが、これが機能しない場合があり、そのときは手動で再始動する必要があります。次のような種類のメッセージが表示されます。

```
0513-059 The qdaemon Subsystem has been started. Subsystem PID is 3000.
```

1 分ほど待った後、コマンド ps -ef | grep qdaemon を再発行します。**qdaemon** はまだアクティブですか、それとも開始した後で終了しましたか?

qdaemon を再始動し、**qdaemon** のプロセス ID (PID) を示すメッセージを受け取ったにもかかわらず、**qdaemon** がアクティブではなくなっている場合があります。/var/spool/lpd/stat/pid という名前のファイルの存在を確認します。コマンド cat /var/spool/lpd/stat/pid を発行して、この確認を行うことがで

きます。このファイルには、アクティブ **qdaemon** の PID が含まれています。**qdaemon** がアクティブでない場合、このファイルは本来なら 除去されているものです。

cat コマンドがディスプレイに番号を出力する場合、その番号はアクティブ **qdaemon** の PID であることが必要です。 **qdaemon** がアクティブでないと既に判断した場合は、ファイル /var/spool/lpd/stat/pid を除去してください。これは、**qdaemon** の前のインスタンスが何らかの理由でこのファイルの除去を行わずに終了したからです。そのファイルが存在しない場合は、次のようなメッセージが表示されます。

```
cat: cannot open /var/spool/lpd/stat/pid
```

qdaemon が非アクティブになっていて、**qdaemon** を再始動しても再び終了し、ファイル /var/spool/lpd/stat/pid が存在していて、そのファイルを除去した場合を考えます。コマンド startsrc -s qdaemon を使用して、もう一度 **qdaemon** を再始動します。1 分ほど待った後、コマンド ps -ef | grep qdaemon を再度発行して、**qdaemon** がアクティブのままかどうか調べます。また、コマンド cat /var/spool/lpd/stat/pid を再度発行することによって、ファイルが再作成され、有効な PID を含んでいるかどうか調べることもできます。

最初の質問「デーモンは実行されていますか?」の答えが「はい」ならば、**qdaemon** は新規ジョブを受け入れる徴候を示す前に、現在実行中のジョブがすべて完了するまで待っている可能性があります。このシナリオは、基本オペレーティング・システムを実行しているマシンの非同期アダプター (64 ポートや 128 ポートのアダプターなど) に、多数のプリンター (25 台を超える) が接続されている場合によく起こります。

qdaemon が他のジョブを実行する前にいずれかのジョブの完了を待っているかどうか調べるには、**lpstat** コマンドを使用して、状況が **RUNNING** になっているジョブが存在するかどうか調べます。存在する場合は、**RUNNING** ジョブを示しているプリンターを物理的に検査して、少なくとも 1 つのジョブが実際に実行中であることを確認します。用紙詰まりまたは用紙切れのために 1 つ以上のプリンターが **DEV_WAIT** を示している場合は、問題を修正して、プリンターが印刷を開始するかどうか確認します。印刷を開始しない場合は、**lpstat** コマンドを再度使用して、キューの状況が **RUNNING** かどうか調べます。ここで説明した状況で、プリンターを検査する目的は、**qdaemon** が新規ジョブを開始していなくても、少なくとも 1 つのプリンターが実際に印刷を行っているかどうか確認することです。

ここで、コマンド enq -Pasc /etc/motd を使用して、新規ジョブをスプーラーに実行依頼します。

lpstat コマンドを使用して、キューの状況を検査します。新規ジョブのジョブ番号が **NEW** ならば、**qdaemon** は何らかの理由で他のジョブの実行に集中しており、現行ジョブが完了するまで新規ジョブを開始しません。待つほかに方法はありません。実行中のジョブを取り消すこともできません。これは、ジョブ取り消し要求もまたジョブであり、**qdaemon** は新規ジョブを受け入れないからです。

スプーラー・キューのテスト

アプリケーションからジョブのスプーリングを行う際に、ジョブが実際にスプーラーに到達したかどうか明確でないことがよくあります。

asc という名前のキューに問題があるとします。

コマンド disable asc を発行して、スプーラー・キューを使用不可に設定します。キューが **DOWN** 状態かどうか検査するには、コマンド lpstat -pasc を発行します。次に、アプリケーションを使用してジョブをキューに実行依頼します。

lpstat を使用して、ジョブが **asc** キューにあるかどうか検査します (キュー状況が一時的な **DOWN** 状態である限り、**qdaemon** はジョブをキューに入れますが、ジョブの処理は許可しません。) ジョブがキューに入っていない場合は、ご自身の知識、アプリケーション資料、またはアプリケーションのテクニカル・サ

ポートを利用して、問題の原因を調べてください。可能な場合は、アプリケーションが使用している正確なジョブ実行依頼コマンドまたはメソッドを調べて、コマンド・ラインから試行します。 **enq** または **qdaemon** から戻されるエラー・メッセージをアプリケーションが非表示にしている可能性があります。

スプール印刷ジョブのコピー

特にリモート・スプーリング環境では、スプール印刷ジョブのコピーを作成すると役に立つことがあります。

ジョブをスプーラーに実行依頼すると、ジョブ記述ファイル (JDF) が作成され、`/var/spool/lpd/qdir` に格納されます。キューがリモート・キューであり、**rembak** などをバックエンドとしている場合、ジョブは印刷サーバーに転送されます。このサーバーで、**enq** は別の JDF を作成し、ジョブを指定された印刷サーバー・キューに入れます。

ジョブが印刷サーバーで消滅しているように見える場合は、印刷サーバー・キューを使用不可にして (ASCII キューの例では `disable asc`)、ジョブを再実行依頼します。 **asc** が停止しているため、ジョブはキューに入れられているものとして **lpstat** コマンドに表示されますが、キューは **DOWN** 状態になり、ジョブは処理されません。このジョブの JDF については、`/var/spool/lpd/qdir` を調べてください。JDF の最終行は、入力データ・ストリームのスプール・コピーの絶対パス名です。そのファイルを `/tmp/myfile` などの一時ファイルにコピーします。このファイルをコピーする際に、ジョブに関連したフラグがすべて失われます。コピーされるものは印刷データ・ストリーム自体のみです。

asc キューを使用可能にして (`enable asc`)、ジョブを処理できるようにします。ジョブが消滅する場合は、作成したコピーを実行依頼します (`enq -Pasc /tmp/myfile`)。このジョブも消滅する場合は、プリンターが何らかの理由でそのジョブを印刷していないので、入力データ・ストリームのエラーを調べる必要があります。コピーが印刷された場合は、元のジョブに関連したフラグに問題があることが考えられます。

クリーンアップと再始動

印刷スプーラーをクリアし、再始動できます。

この手順では、スプーラー・サブシステムを完全にクリアし、再始動します。処理のためにキューに入れられているすべてのジョブが削除され、これらのジョブを再実行依頼する必要があります。操作不可のスプーラーのトラブルシューティングができない場合に、この方法を使用してください。この作業を実行するには、root ユーザーであることが必要です。

1. **qdaemon** を停止します。

   ```
   stopsrc -s qdaemon
   ```

2. 関連したプロセスを停止します。

   ```
   ps -ef | grep qd
   kill -9 PIDNumbers
   ```

   ここで、*PIDNumbers* は **ps** コマンドから得られる PID です。**qdfork** が検出される可能性があります。

3. 関連したプロセスを停止します。

   ```
   ps -ef | grep pio
   kill -9 PIDNumbers
   ```

   ここで、*PIDNumbers* は **ps** コマンドから得られる PID です。**pioformat** または **pioout** が検出される可能性があります。

4. キューとデバイス状況のディレクトリーを空にします。

```
rm /var/spool/lpd/stat/*_dev_*
rm /var/spool/lpd/stat/s*
```

ファイル /var/spool/lpd/stat/numfile には、最後に割り当てられたジョブ番号を示す整数が含まれています。ジョブ番号付け体系を最初に戻して構わなければ、次のように入力します。

```
rm /var/spool/lpd/stat/*
```

5. スプール・ジョブを除去します。

```
rm /var/spool/lpd/qdir/*
rm /var/spool/qdaemon/*
```

6. qdaemon を再始動します。

```
startsrc -s qdaemon
```

ps コマンドを発行すると、親プロセス ID (PPID) が **1** であるプロセスが見つかることがあります。これらのプロセスを **kill -9** によって強制終了できない場合は、これらのプロセスを削除するためにシステムをリブートする必要があります。

印刷用語

印刷に関してよく使用される用語がいくつかあります。

フォーマッター・フィルター

入力パラメーターに基づいて、入力印刷ファイルのフォーマットを設定するか、または変更せずに引き渡す機能を備えています。フィルターが入力ファイルを変更せずに引き渡す場合でも、入力ファイルの印刷前にプリンターを初期化するプリンター・コマンドを送信し、印刷の完了後にはプリンターを復元します。

フォーマッター・フィルターは、次のコンポーネントから構成されます。

- デバイス独立のフォーマッター・ドライバー
- デバイス依存のフォーマッター

入力データのそれぞれのタイプ (またはタイプのグループ) ごとに、フォーマッターが存在します。例えば、サポートされる IBM プロプリンターすべてに対して、フォーマッターが 1 つ存在します。

フォーマッター・ドライバーは、パイプラインによって呼び出され、駆動するフォーマッターの名前を渡されます。フォーマッター・ドライバーは、フォーマッターを動的にロードしてリンクし、フォーマッターの **setup** 機能を呼び出します。この機能は、データ・フォーマットまたはデータ・パススルーのどちらが要求されているかを指示します。フォーマッターの **setup** 機能が変更またはパススルーする入力ファイルを準備した後、処理はフォーマッター・ドライバーに戻ります。フォーマッター・ドライバーは次に、**initialize** 機能を呼び出します。この機能は、プリンターを開始する一連のプリンター・コマンドを出力します。詳しくは、**initialize** 機能および **setup** 機能を参照してください。

次にフォーマッター・ドライバーは、**passthru** 機能を 1 回呼び出すか、印刷ファイルの各行ごとに **lineout** 機能を呼び出します。これは、**setup** 機能からの戻りコードによって決まります。**lineout** 機能が呼び出されると、フォーマッター・ドライバーはすべての垂直スペーシングを自動的に行うか (用紙送り、上部マージン、および下部マージン)、または **lineout** 機能を介して行います (行送り、垂直タブ)。詳しくは、**passthru** 機能および **lineout** 機能を参照してください。

処理が完了すると、フォーマッター・ドライバーは **restore** 機能を呼び出します。**restore** 機能は、プリンターをデフォルト状態に復元する一連のプリンター・コマンドを出力します。このデフォルト状態は、データベース属性値によって定義されます。詳しくは、**restore** 機能を参照してください。

印刷フォーマッターがプリンター・フォーマッターのサブルーチンと相互作用する方法について詳しくは、84 ページの『印刷フォーマッターの例』を参照してください。

ローカル・プリンター
ノードまたはホストに接続されたプリンター。

印刷ジョブ
プリンター上で実行される作業単位。印刷ジョブの要求方法に応じて、印刷ジョブには 1 つ以上のファイルの印刷が含まれます。システムは、印刷するそれぞれのジョブに固有のジョブ番号を割り当てます。

印刷スプーラー

プリンターへのキューに入れられた印刷ジョブを含む、さまざまなタイプのジョブのキューイングに使用できる汎用スプーリング機能。スプーラーは通常、キューに入れるジョブのタイプを区別しません。システム管理者が、キューに対して指定されたスプーラー・バックエンド・プログラムに基づいて、スプーラー・キューを定義します。例えば、スプーラー・バックエンド・プログラムが **piobe** コマンド (プリンター入出力バックエンド) ならば、キューは印刷キューです。

同様に、スプーラー・バックエンド・プログラムがコンパイラーならば、キューはコンパイル・ジョブ用です。スプーラーの **qdaemon** プロセスがスプーラー・キューからジョブを選択すると、キューの定義時にシステム管理者によって指定されるバックエンド・プログラムを呼び出して、そのジョブを実行します。

ネットワークが、基本オペレーティング・システム・マシンと、他のタイプのクライアントおよびサーバーで構成されている場合は、ネットワーク全体でサポートされないリモート印刷要求があります。場合によっては、印刷ジョブを一度に 1 ファイルずつ実行依頼したり、印刷ジョブとして実行依頼する前にファイルを連結したりする必要が生じることがあります。

基本のスプーラー・コマンドは **enq** コマンドです。このコマンドを直接呼び出して印刷ジョブをキューに入れることもできますが、3 つのフロントエンド・コマンドが、印刷ジョブを実行依頼するためのコマンドとして定義されています (**lp**、**lpr**、および **qprt** の各コマンドを参照)。これらのコマンドのいずれかによって発行された印刷要求は、まず **enq** プログラムに渡され、次にこのプログラムが、**qdaemon** の処理するファイルに関する情報をキューに入れます。このキューは `/var/spool/lpd/qdir` ディレクトリーです。

ジョブがファイルでなければ (つまり、**enq** へのコマンドのパイプ出力)、印刷されるデータを含む実ファイルが `/var/spool/qdaemon` 内に作成されます。`/var/spool/lpd/qdir` ファイルの情報は、`/var/spool/qdaemon` 内のファイルを指しています。

プリンター・バックエンド

印刷対象としてキューに入れられた印刷ジョブを管理するために、スプーラーの **qdaemon** プロセスによって呼び出されるプログラムの集合。プリンター・バックエンドは、次の機能を実行します。

- 印刷される 1 つ以上のファイルのリストを **qdaemon** プロセスから受け取ります。
- データベースから取得したプリンターおよびフォーマットの属性値 (コマンド・ラインで入力されたフラグによって指定変更される) を使用します。
- ファイルを印刷する前にプリンターを初期化します。

- 必要に応じてフィルターを実行し、印刷データ・ストリームをプリンターがサポートするフォーマットに変換します。
- ASCII 文書の単純なフォーマット設定用のフィルターを提供します。
- 各国語文字の印刷のサポートを提供します。
- フィルター済み印刷データ・ストリームをプリンター・デバイス・ドライバーに渡します。
- ヘッダー・ページとトレーラー・ページを生成します。
- 複数部印刷のコピーを生成します。
- 用紙切れや要介入エラーなどのプリンター・エラー条件を報告します。
- フィルターによって検出された問題を報告します。
- 印刷ジョブが取り消された後にクリーンアップを行います。
- システム管理者が特定の印刷要件を満たすためにカスタマイズできる印刷環境を提供します。

mkvirprt コマンドは、プリンター・バックエンドに対して仮想プリンターを定義します。特定タイプのプリンター用の事前定義属性セットをコピーして、カスタマイズした属性セットを作成します。カスタマイズした属性は、**lsvirprt** コマンドを使用してリストできます。また、**chvirprt** コマンド、SMIT (「**Change / Show Print Queue Characteristics (印刷キューの特性の変更/表示)**」オプション) を使用して、属性を変更できます。 **mkvirprt** コマンドまたは **chvirprt** コマンドを使用するたびに、ダイジェスト・ユーティリティー (**piodigest** コマンド) が自動的に実行され、属性値とルックアップ・テーブルのメモリー・イメージを構成します。これらは、印刷処理中に読み込まれて使用されます。

qdaemon コマンドは、**piobe** コマンド (印刷ジョブ・マネージャー) を呼び出し、フラグ・オプションと印刷される 1 つ以上のファイルの名前を渡します。渡されないフラグ・オプションは、**enq** コマンドによって除去されるスプーラー・フラグ・オプションのみです。これは、**qdaemon** コマンドが既にプリンター・デバイスを開き、標準出力をプリンターにリダイレクトしているからです。**qdaemon** とバックエンドの間でのやり取りは、状況ファイルによって行われます。

ヘッダー・ページが必要な場合、**piobe** コマンドはヘッダー・ページを生成するヘッダー・ページ・パイプラインを取得します。ヘッダー・ページ・パイプラインは、シェルに渡されます。パイプライン内では、ヘッダー・ページ・フィルターからの標準出力が、フォーマッター・フィルターの標準入力になります。フォーマッター・フィルターはヘッダー・ページを処理し、結果を標準出力に書き込みます。フォーマッター・フィルターの標準出力は、デバイス・ドライバー・インターフェース・プログラムの標準入力となり、このプログラムがフィルター済みヘッダー・ページをプリンター・デバイス・ドライバーに書き込みます。

プリンター/プロッター・デバイス

`/dev` ディレクトリー内にある、デバイス用のスペシャル・ファイル。このファイルは、リダイレクトによって使用できます (例: `cat FileName > /dev/lp0`)。このデバイス・ドライバーの設定値は、**chdev** コマンドを使用して表示および変更できます。プリンター・コマンドがプリンター・デバイスにアクセスできるように、事前にそのデバイス用の印刷キューを作成するか、または `/etc/qconfig` 内でプリンター・バックエンドに対してそのプリンターを構成する必要があります。

qdaemon

qdaemon は、バックグラウンドで実行されてキューを制御するプロセスです。通常は、システムがオンになったときに、**startsrc** コマンドによって開始されます。

qdaemon は、`/var/spool/lpd/qdir` ディレクトリー内の印刷要求を追跡し、ジョブが適時に正しいプリンターに送られるようにします。また、プリンターの状況を追跡し、システム・アカウンテ

ィングのためにプリンター使用率のデータを保管します。この情報は /var/spool/lpd/stat ディレクトリーに格納され、**lpstat** コマンドと **enq -A** コマンドを使用してこの情報にアクセスできます。

qdaemon が停止した場合は、**srcmstr** プロセスによって再始動されます。

注: **srcmstr** プロセスは停止しないでください。このプロセスは、システム上で稼働する他のデーモンを制御します。

キュー 印刷ジョブを送る先の場所。これは、キューの名前と一致する /etc/qconfig ファイル内のスタンザで、関連したキュー・デバイスを指定します。次に例を示します。

```
Msa1:
   device = lp0
```

キュー・デバイス

/etc/qconfig ファイル内のスタンザで、通常はローカル・キュー・スタンザの後に指定されます。印刷先の /dev ファイル (プリンター・デバイス) と、使用するバックエンドを指定します。次に例を示します。

```
lp0:
      file = /dev/lp0
   header = never
   trailer = never
   access = both
      backend = /usr/lpd/piobe
```

前記の例では、lp0 がデバイス名で、後続の行はデバイスの使用方法を定義します。

実プリンター

固有のハードウェア・デバイス・アドレスをもつ、シリアル・ポートまたはパラレル・ポートに接続されたプリンター・ハードウェア。カーネル内のプリンター・デバイス・ドライバーがプリンター・ハードウェアとやり取りし、プリンター・ハードウェアと仮想プリンター間のインターフェースを提供します。

コマンド・ラインで **mkdev** コマンドを使用することにより、実プリンターを追加することができます。詳しくは、**mkdev** コマンドを参照してください。

リモート・プリンター

ローカル・システムに直接接続していないプリンター。リモート印刷システムを使用すれば、プリンターに直接リンクしていないノードからプリンターにアクセスできます。

リモート印刷機能を使用するには、個々のノードが TCP/IP を使用してネットワークに接続されていて、必要な TCP/IP アプリケーションをサポートしていることが必要です。

シリアル・プリンター

機能を順次実行するプリンター。例えば、一度に 1 文字ずつ印刷するプリンターです。

シリアル・プリンターは通常は DTE として構成されます。つまり、*受信* データ・ライン上でデータを受信し、*送信* データ・ライン上でデータを送信する必要があります。シリアル・プリンターはデフォルトで EIA-232 接続を行い、DB-25 D 型コネクターを使用します。多くのプリンターは EIA-422 接続もサポートします。

仮想プリンター

仮想プリンター定義 とも呼ばれます。これは、特定のデータ・ストリーム (ASCII や PostScript など) を特定のプリンターに対して記述する属性値のセットを含むファイルです。これには、プリンター・ハードウェアをホスト・コンピューターに接続する方法や、データのバイトをプリンターとの間で送受信するために使用されるプロトコルに関する情報は含まれません。仮想プリンター

は、印刷キューに関連付けられます。プリンターがサポートするそれぞれのデータ・ストリームごとに、印刷キューを定義できます。複数の印刷キューが同じ実プリンターを使用できます。

印刷ジョブをキューに入れるには、印刷キューとキュー・デバイスの両方に対して、仮想プリンター定義が既に存在している必要があります。詳しくは、**mkvirprt** コマンドを参照してください。

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒103-8510
東京都中央区日本橋箱崎町19番21号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
Dept. LRAS/Bldg. 903
11501 Burnet Road
Austin, TX 78758-3400
USA

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性がありますが、その測定値が、一般に利用可能なシステムのものと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態で提供されるものであり、いかなる保証も提供されません。 IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。

このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。

プライバシー・ポリシーに関する考慮事項

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品 (「ソフトウェア・オファリング」) では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。 IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらのCookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。

この「ソフトウェア・オファリング」は、Cookie もしくはその他のテクノロジーを使用して個人情報を収集することはありません。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie などの各種テクノロジーの使用について詳しくは、『IBM オンラインでのプライバシー・ステートメントのハイライト』(http://www.ibm.com/privacy/jp/ja/)、『IBM オンラインでのプライバシー・ステートメント』(http://www.ibm.com/privacy/details/jp/ja/) の『クッキー、ウェブ・ビーコン、その他のテクノロジー』というタイトルのセクション、および『IBM Software Products and Software-as-a-Service Privacy Statement』(http://www.ibm.com/software/info/product-privacy) を参照してください。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標です。 他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。 現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml の「Copyright and trademark information」をご覧ください。

UNIX は、The Open Group の米国およびその他の国における登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ハ行]

[マ行]

[ヤ行]

[ラ行]

[数字]

A

C

D

E

G

H

I

L

M

P

Q

R

S

T

［特殊文字］

Printed in Japan

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21