(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2005-295515
(P2005-295515A)
(43)公開日　平成17年10月20日(2005.10.20)

(51)Int.Cl.[7]	F I		テーマコード（参考）
H04N 1/00	H04N 1/00	C	2C061
B41J 29/38	B41J 29/38	Z	2H027
B41J 29/42	B41J 29/42	F	5B021
G03G 21/00	G03G 21/00	386	5B042
G06F 3/00	G06F 3/00	652A	5C062

審査請求 未請求　請求項の数 38　ＯＬ　（全 51 頁）　最終頁に続く

(21)出願番号　特願2005-57890(P2005-57890)
(22)出願日　平成17年3月2日(2005.3.2)
(31)優先権主張番号　特願2004-64048(P2004-64048)
(32)優先日　平成16年3月8日(2004.3.8)
(33)優先権主張国　日本国(JP)
(31)優先権主張番号　特願2004-64049(P2004-64049)
(32)優先日　平成16年3月8日(2004.3.8)
(33)優先権主張国　日本国(JP)

(71)出願人　000006747
株式会社リコー
東京都大田区中馬込１丁目３番６号
(74)代理人　100070150
弁理士　伊東　忠彦
(72)発明者　高橋　久憲
東京都大田区中馬込１丁目３番６号　株式会社リコー内

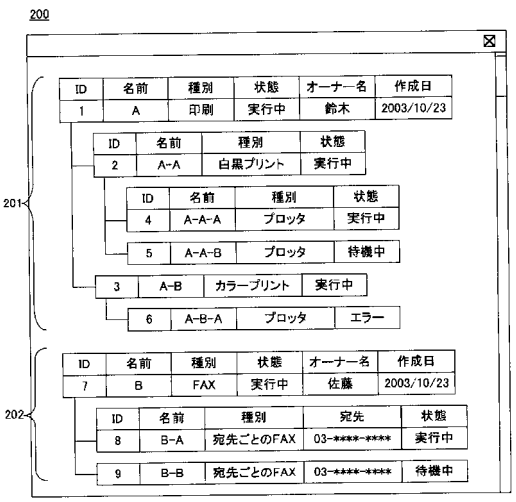
Ｆターム(参考)　2C061 AP07 CQ23 CQ34 HK11 HQ02 HQ03
2H027 FA30 FA35 GA34 GA52 GA56 GB05 GB14 ZA07
5B021 AA01 BB06
5B042 GA23 HH30 MA13 MC21 NN16

最終頁に続く

(54)【発明の名称】電子機器、ジョブ表示方法、ジョブ表示プログラム

(57)【要約】
【課題】　画像形成処理のジョブの管理に好適なユーザインタフェースを提供する電子機器、ジョブ表示方法と、その方法を実行するジョブ表示プログラムを提供する。
【解決手段】　所定の処理により発生するジョブに関するジョブ情報を表示する電子機器において、任意のジョブと他のジョブとの関連を示すジョブ構造情報と、前記ジョブ構造情報に基づいて、前記ジョブ情報を表示するＧＵＩ表示手段とを有する。
【選択図】　図６０

ユーザインタフェースを示す図である(その1)



【特許請求の範囲】
【請求項１】
所定の処理により発生するジョブに関するジョブ情報を表示する電子機器において、
任意のジョブと他のジョブとの関連を示すジョブ構造情報と、
前記ジョブ構造情報に基づいて、前記ジョブ情報を表示するＧＵＩ表示手段と
を有することを特徴とする電子機器。
【請求項２】
前記ＧＵＩ表示手段は、前記ジョブが属する機器を、該ジョブに関連付けて表示することを特徴とする請求項１に記載の電子機器。
【請求項３】
前記ＧＵＩ表示手段は、全てのジョブのジョブ情報を表示することを特徴とする請求項１に記載の電子機器。
【請求項４】
前記ジョブを検索するためのジョブ検索手段を有することを特徴とする請求項３に記載の電子機器。
【請求項５】
前記ＧＵＩ表示手段は、前記ジョブのうちの根幹ジョブのみのジョブ情報を表示することを特徴とする請求項１に記載の電子機器。
【請求項６】
前記ＧＵＩ表示手段は、前記ジョブのうちの末端ジョブのみのジョブ情報を表示することを特徴とする請求項１に記載の電子機器。
【請求項７】
前記ＧＵＩ表示手段は、前記ジョブ同士の関係を明示するように前記ジョブ情報を表示することを特徴とする請求項１に記載の電子機器。
【請求項８】
前記ＧＵＩ表示手段は、各ジョブに対して実行可能な所定の操作項目を表示することを特徴とする請求項１に記載の電子機器。
【請求項９】
前記操作項目は、ジョブの中止、ジョブの中断、ジョブの再実行、前記ジョブ情報の編集、前記ジョブを実行する機器の変更のうちのいずれか１つ以上の項目であることを特徴とする請求項８に記載の電子機器。
【請求項１０】
前記ジョブ情報は、前記ジョブの処理内容を示す種別と、前記ジョブの状態と、前記ジョブを識別するための識別情報と、前記ジョブのオーナー名と、前記ジョブが作成された日のうちのいずれか１つ以上の情報を含むことを特徴とする請求項１から９のいずれか１項に記載の電子機器。
【請求項１１】
前記ジョブ構造情報は、前記ジョブの親子関係を示す情報であることを特徴とする請求項１から１０のいずれか１項に記載の電子機器。
【請求項１２】
前記ジョブ構造情報を、１つ以上の機器から取得するジョブ構造情報取得手段を有することを特徴とする請求項１から１１のいずれか１項に記載の電子機器。
【請求項１３】
前記各機器から取得したジョブ構造情報を結合して１つのジョブ構造情報とする全体ジョブ構造情報作成手段を有することを特徴とする請求項１２に記載の電子機器。
【請求項１４】
前記ジョブ構造情報は、前記ジョブごとに作成されるジョブ関連情報に基づき作成されることを特徴とする請求項１から１３のいずれか１項に記載の電子機器。
【請求項１５】
前記ジョブ関連情報は、前記ジョブの祖先にあたる全てのジョブの識別情報を含むことを

特徴とする請求項１４に記載の電子機器。
【請求項１６】
前記ジョブ関連情報は、前記ジョブの親ジョブの識別情報を含むことを特徴とする請求項１４に記載の電子機器。
【請求項１７】
前記ジョブ関連情報は、前記ジョブの子孫にあたる全てのジョブの識別情報を含むことを特徴とする請求項１４に記載の電子機器。
【請求項１８】
前記ジョブ関連情報は、前記ジョブの子ジョブの識別情報を含むことを特徴とする請求項１４に記載の電子機器。
【請求項１９】
画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとをさらに有することを特徴とする請求項１から１８に記載の電子機器。
【請求項２０】
前記ＧＵＩ表示手段からの通知により、他の機器に属していたジョブを実行することを特徴とする請求項１９に記載の電子機器。
【請求項２１】
前記ＧＵＩ表示手段からの通知により、他の機器に自らが実行すべきジョブを実行させることを特徴とする請求項１９に記載の電子機器。
【請求項２２】
所定の処理により発生するジョブに関するジョブ情報を表示するジョブ表示方法において、
任意のジョブと他のジョブとの関連を示すジョブ構造情報を受信するジョブ構造受信段階と、
前記ジョブ構造情報に基づいて、前記ジョブ情報を表示するＧＵＩ表示段階と
を有することを特徴とするジョブ表示方法。
【請求項２３】
前記ＧＵＩ表示段階は、前記ジョブが属する機器を、該ジョブに関連付けて表示することを特徴とする請求項２２に記載のジョブ表示方法。
【請求項２４】
前記ＧＵＩ表示段階は、全てのジョブのジョブ情報を表示することを特徴とする請求項２２に記載のジョブ表示方法。
【請求項２５】
前記ＧＵＩ表示段階は、前記ジョブのうちの根幹ジョブのみのジョブ情報を表示することを特徴とする請求項２２に記載のジョブ表示方法。
【請求項２６】
前記ＧＵＩ表示段階は、前記ジョブのうちの末端ジョブのみのジョブ情報を表示することを特徴とする請求項２２に記載のジョブ表示方法。
【請求項２７】
前記ＧＵＩ表示段階は、前記ジョブ同士の関係を明示するように前記ジョブ情報を表示することを特徴とする請求項２２に記載のジョブ表示方法。
【請求項２８】
前記ＧＵＩ表示段階は、各ジョブに対して実行可能な所定の操作項目を表示することを特徴とする請求項２２に記載のジョブ表示方法。
【請求項２９】
前記操作項目は、ジョブの中止、ジョブの中断、ジョブの再実行、前記ジョブ情報の編集、前記ジョブを実行する機器を変更のうちのいずれか１つ以上の項目であることを特徴とする請求項２８に記載のジョブ表示方法。
【請求項３０】
前記ジョブ情報は、前記ジョブの処理内容を示す種別と、前記ジョブの状態と、前記ジョ

ブを識別するための識別情報と、前記ジョブのオーナー名と、前記ジョブが作成された日のうちのいずれか１つ以上の情報を含むことを特徴とする請求項２２から２９のいずれか１項に記載のジョブ表示方法。
【請求項３１】
前記ジョブ構造情報は、前記ジョブの親子関係を示す情報であることを特徴とする請求項２２から３０のいずれか１項に記載のジョブ表示方法。
【請求項３２】
前記ジョブ構造情報を、１つ以上の機器から取得するジョブ構造情報取得段階を有することを特徴とする請求項２２から３１のいずれか１項に記載のジョブ表示方法。
【請求項３３】
前記ジョブ構造情報は、前記ジョブごとに作成されるジョブ関連情報に基づき作成されることを特徴とする請求項２２から３２のいずれか１項に記載のジョブ表示方法。
【請求項３４】
前記ジョブ関連情報は、前記ジョブの祖先にあたる全てのジョブの識別情報を含むことを特徴とする請求項３３に記載のジョブ表示方法。
【請求項３５】
前記ジョブ関連情報は、前記ジョブの親ジョブの識別情報を含むことを特徴とする請求項３３に記載のジョブ表示方法。
【請求項３６】
前記ジョブ関連情報は、前記ジョブの子孫にあたる全てのジョブの識別情報を含むことを特徴とする請求項３３に記載のジョブ表示方法。
【請求項３７】
前記ジョブ関連情報は、前記ジョブの子ジョブの識別情報を含むことを特徴とする請求項３３に記載のジョブ表示方法。
【請求項３８】
コンピュータに、請求項２２から請求項３７に記載のジョブ表示方法を実行させるためのジョブ表示プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、画像形成に係るジョブを表示する電子機器、ジョブ表示方法、及びその方法を実行するジョブ表示プログラムに関する。
【背景技術】
【０００２】
近年、ファクシミリ、プリンタ、コピーおよびスキャナなどの各装置の機能を１つの筐体内に収納した画像形成装置が知られるようになった。この画像形成装置は、１つの筐体内に表示部、印刷部および撮像部などを設けると共に、ファクシミリ、プリンタ、コピーおよびスキャナにそれぞれ対応する４種類のアプリケーションを設け、そのアプリケーションを切り替えることより、ファクシミリ、プリンタ、コピーおよびスキャナとして動作させるものである。
【０００３】
このような画像を出力、変更、移動動作などの画像形成に係る処理は、ジョブといわれる処理単位で実行される。例えばコピーを行う場合、画像形成装置では原稿を読み取る処理と、読み取った原稿を印刷する処理が行われるため、原稿読み取りジョブと印刷のジョブが発生する。このジョブが発生しない例として、画像形成装置の設定を変更する操作を行う場合があり、このときは画像に対する処理が行われないので、ジョブは発生しない。
【０００４】
このジョブは、処理の過程でより細かいジョブとなることがある。このことを、図１を用いて説明する。例えばコピーを行う場合、画像形成装置には、まず「コピージョブ」が発生する。そして、このコピージョブから「読み取りジョブ」と「印刷ジョブ」の２つの

ジョブが派生する。このように１つのジョブから新たなジョブが派生した場合、前者を「親ジョブ」、後者を「子ジョブ」と呼ぶことにする。

【０００５】

親ジョブ、子ジョブと表現すると、両者が同時に存在するようなイメージとなるが、従来では子ジョブが派生すると、親ジョブは消滅する。上の例で言えば、「読み取りジョブ」と「印刷ジョブ」が派生すると、「コピージョブ」は消滅する。従って、従来において派生するとは、親ジョブが分裂して子ジョブとなるイメージである。

【０００６】

上述したコピージョブでは、１回の派生であったが、処理によってはジョブの派生が複数回行われる場合がある。例えば、図２に示されるような、カラーページと白黒ページが混在した文書を印刷する際に、カラーページと白黒ページを別のプロセスで印刷する場合を考える。

【０００７】

まず、画像形成装置がユーザから受けた要求により、「印刷ジョブ」が発生する。そして、印刷ジョブから「カラープリントジョブ」と「白黒プリントジョブ」が派生する。さらに、印刷を行うページ数の単位で「プロッタジョブ」が派生する。

【０００８】

このように派生していく様子を図にすると、図２に示されるようなジョブ構造となるが、このジョブ構造における最も根元にあるジョブを「根幹ジョブ」、末端にあるジョブを「末端ジョブ」と呼ぶ。図２の場合、根幹ジョブは、プリントジョブであり、末端ジョブは、プロッタジョブである。この根幹ジョブは、従来、スプールで処理されると消滅するようになっている。

【０００９】

また、従来技術におけるジョブ管理では、各アプリケーション（以下、アプリケーションをアプリと表現することがある）が根幹ジョブ用のキューをそれぞれ持っており、ジョブを処理する順番を管理している。例えば、図３に示されるように、プリンタ用のアプリケーションであるプリンタアプリは、プリンタアプリ根幹ジョブ用キューで、キューに積まれた根幹ジョブＡ、Ｂ、Ｃのように、順番を管理している。

【００１０】

また、プロッタなどのハードウェアに対する末端ジョブも専用のキューがあり、それにより処理する順番の管理を行っている。例えば、図４に示されるように、各アプリや末端のジョブを管理するサービス層のモジュールは、プロッタジョブ用キューで、キューに積まれた末端ジョブＡ、Ｂ、Ｃのように、順番を管理する。

【００１１】

ジョブを管理するモジュールはそれらのキューにアクセスすることでジョブの情報を取得し、その情報をパソコン上のアプリに渡すことで、アプリが図５のようなユーザインタフェースを表示することができる。図５に示される従来のユーザインタフェースは、根幹ジョブを表示するもので、根幹ジョブがスプールで処理されると消滅するようになっている。

【発明の開示】

【発明が解決しようとする課題】

【００１２】

以上のように、従来は、根幹ジョブがスプールで処理されると消滅するので、ある子ジョブがどの親ジョブから派生したものか判別できない。そのため、例えば、ある末端ジョブでエラーが発生したが、それがどの要求で生じたエラーなのか判別できなかった。また、ある要求に対して発生した根幹ジョブを途中で中止しようとしても、派生した子ジョブの中止ができないため、処理が中止できないという問題点がある。

【００１３】

本発明は、このような問題点に鑑み、画像形成処理のジョブの管理に好適なユーザインタフェースを提供する電子機器、ジョブ表示方法と、その方法を実行するジョブ表示プロ

グラムを提供することを目的とする。
【課題を解決するための手段】
【００１４】
上記課題を解決するために、本発明は、所定の処理により発生するジョブに関するジョブ情報を表示する電子機器において、任意のジョブと他のジョブとの関連を示すジョブ構造情報と、前記ジョブ構造情報に基づいて、前記ジョブ情報を表示するＧＵＩ表示手段とを有することを特徴とする。
【００１５】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示手段は、前記ジョブが属する機器を、該ジョブに関連付けて表示することを特徴とする。
【００１６】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示手段は、全てのジョブのジョブ情報を表示することを特徴とする。
【００１７】
また、上記課題を解決するために、本発明は、前記ジョブを検索するためのジョブ検索手段を有することを特徴とする。
【００１８】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示手段は、前記ジョブのうちの根幹ジョブのみのジョブ情報を表示することを特徴とする。
【００１９】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示手段は、前記ジョブのうちの末端ジョブのみのジョブ情報を表示することを特徴とする。
【００２０】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示手段は、前記ジョブ同士の関係を明示するように前記ジョブ情報を表示することを特徴とする。
【００２１】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示手段は、各ジョブに対して実行可能な所定の操作項目を表示することを特徴とする。
【００２２】
また、上記課題を解決するために、本発明は、前記操作項目は、ジョブの中止、ジョブの中断、ジョブの再実行、前記ジョブ情報の編集、前記ジョブを実行する機器を変更のうちのいずれか１つ以上の項目であることを特徴とする。
【００２３】
また、上記課題を解決するために、本発明は、前記ジョブ情報は、前記ジョブの処理内容を示す種別と、前記ジョブの状態と、前記ジョブを識別するための識別情報と、前記ジョブのオーナー名と、前記ジョブが作成された日のうちのいずれか１つ以上の情報を含むことを特徴とする。
【００２４】
また、上記課題を解決するために、本発明は、前記ジョブ構造情報は、前記ジョブの親子関係を示す情報であることを特徴とする。
【００２５】
また、上記課題を解決するために、本発明は、前記ジョブ構造情報を、１つ以上の機器から取得するジョブ構造情報取得手段を有することを特徴とする。
【００２６】
また、上記課題を解決するために、本発明は、前記各機器から取得したジョブ構造情報を結合して１つのジョブ構造情報とする全体ジョブ構造情報作成手段を有することを特徴とする。
【００２７】
また、上記課題を解決するために、本発明は、前記ジョブ構造情報は、前記ジョブごとに作成されるジョブ関連情報に基づき作成されることを特徴とする。

【００２８】
また、上記課題を解決するために、本発明は、前記ジョブ関連情報は、前記ジョブの祖先にあたる全てのジョブの識別情報を含むことを特徴とする。
【００２９】
また、上記課題を解決するために、本発明は、前記ジョブ関連情報は、前記ジョブの親ジョブの識別情報を含むことを特徴とする。
【００３０】
また、上記課題を解決するために、本発明は、前記ジョブ関連情報は、前記ジョブの子孫にあたる全てのジョブの識別情報を含むことを特徴とする。
【００３１】
また、上記課題を解決するために、本発明は、前記ジョブ関連情報は、前記ジョブの子ジョブの識別情報を含むことを特徴とする。
【００３２】
また、上記課題を解決するために、本発明は、画像形成処理で使用されるハードウェア資源と、画像形成に係る処理を行うプログラムとをさらに有することを特徴とする。
【００３３】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示手段からの通知により、他の機器に属していたジョブを実行することを特徴とする。
【００３４】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示手段からの通知により、他の機器に自らが実行すべきジョブを実行させることを特徴とする。
【００３５】
また、上記課題を解決するために、本発明は、所定の処理により発生するジョブに関するジョブ情報を表示するジョブ表示方法において、任意のジョブと他のジョブとの関連を示すジョブ構造情報を受信するジョブ構造受信段階と、前記ジョブ構造情報に基づいて、前記ジョブ情報を表示するＧＵＩ表示段階とを有することを特徴とする。
【００３６】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示段階は、前記ジョブが属する機器を、該ジョブに関連付けて表示することを特徴とする。
【００３７】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示段階は、全てのジョブのジョブ情報を表示することを特徴とする。
【００３８】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示段階は、前記ジョブのうちの根幹ジョブのみのジョブ情報を表示することを特徴とする。
【００３９】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示段階は、前記ジョブのうちの末端ジョブのみのジョブ情報を表示することを特徴とする。
【００４０】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示段階は、前記ジョブ同士の関係を明示するように前記ジョブ情報を表示することを特徴とする。
【００４１】
また、上記課題を解決するために、本発明は、前記ＧＵＩ表示段階は、各ジョブに対して実行可能な所定の操作項目を表示することを特徴とする。
【００４２】
また、上記課題を解決するために、本発明は、前記操作項目は、ジョブの中止、ジョブの中断、ジョブの再実行、前記ジョブ情報の編集、前記ジョブを実行する機器を変更のうちのいずれか１つ以上の項目であることを特徴とする。
【００４３】
また、上記課題を解決するために、本発明は、前記ジョブ情報は、前記ジョブの処理内

容を示す種別と、前記ジョブの状態と、前記ジョブを識別するための識別情報と、前記ジョブのオーナー名と、前記ジョブが作成された日のうちのいずれか１つ以上の情報を含むことを特徴とする。

【００４４】

また、上記課題を解決するために、本発明は、前記ジョブ構造情報は、前記ジョブの親子関係を示す情報であることを特徴とする。

【００４５】

また、上記課題を解決するために、本発明は、前記ジョブ構造情報を、１つ以上の機器から取得するジョブ構造情報取得段階を有することを特徴とする。

【００４６】

また、上記課題を解決するために、本発明は、前記ジョブ構造情報は、前記ジョブごとに作成されるジョブ関連情報に基づき作成されることを特徴とする。

【００４７】

また、上記課題を解決するために、本発明は、前記ジョブ関連情報は、前記ジョブの祖先にあたる全てのジョブの識別情報を含むことを特徴とする。

【００４８】

また、上記課題を解決するために、本発明は、前記ジョブ関連情報は、前記ジョブの親ジョブの識別情報を含むことを特徴とする。

【００４９】

また、上記課題を解決するために、本発明は、前記ジョブ関連情報は、前記ジョブの子孫にあたる全てのジョブの識別情報を含むことを特徴とする。

【００５０】

また、上記課題を解決するために、本発明は、前記ジョブ関連情報は、前記ジョブの子ジョブの識別情報を含むことを特徴とする。

【００５１】

また、上記課題を解決するために、本発明は、コンピュータに、請求項２２から請求項３７に記載のジョブ表示方法を実行させるためのジョブ表示プログラム。

【発明の効果】

【００５２】

本発明は以上説明したように、画像形成処理のジョブの管理に好適なユーザインタフェースを提供する電子機器、ジョブ表示方法と、その方法を実行するジョブ表示プログラムを提供することができる。

【発明を実施するための最良の形態】

【００５３】

以下、図面を参照し、本発明を実施するための最良の形態について説明する。本実施の形態では、最初に処理の概要を説明し、その後、処理の詳細な説明を行い、ユーザインタフェースについて説明する。また、電子機器をＭＦＰ（Multi Function Printer）と記すことにする。なお、ジョブを表示するのは、ＭＦＰに限らず、ＭＦＰと接続されたパソコンであっても良い。

【００５４】

最初に処理の概要について説明する。本実施の形態は、ジョブに関する内容が多くを占めるので、まずジョブに対する操作とジョブの状態について説明する。ジョブに対する操作には、「ジョブの中止」と、「ジョブの中断」と、「ジョブの再実行」と、「ジョブ情報の編集」と、「ジョブを実行する機器の変更」とがある。

【００５５】

「ジョブの中止」とは、ジョブの実行を取り止める操作である。この「ジョブの中止」は、ジョブの状態が「実行中」、「エラー発生中」、「待機中」、「中断中」であるジョブに対して行うことができる。中止されたジョブは、それ以降実行されることはない。

【００５６】

「ジョブの中断」とは、ジョブの実行を一時的に止める操作である。この「ジョブの中

断」は、ジョブの状態が「実行中」もしくは「待機中」であるジョブに対して行うことができる。

【００５７】

「ジョブの再実行」とは、「エラー発生中」、「中断中」の状態のジョブを再び実行する操作である。再実行時に他のジョブが実行されていて、再実行されたジョブをすぐに実行できない場合、そのジョブの状態は「待機中」となる。

【００５８】

「ジョブ情報の編集」とは、ジョブを実行する時の条件を変更する操作である。例えばＦＡＸ送信の宛先を別の宛先に変更する操作などが、ジョブ情報の編集にあたる。ジョブ情報の編集は、ジョブの状態が「エラー発生中」、「待機中」、「中断中」であるジョブに対して行うことができる。

【００５９】

「ジョブを実行する機器の変更」とは、ジョブを実行する機器を変更する操作である。例えば、２台のＭＦＰがあり、現在印刷しているＭＦＰにかえて、もう１つのＭＦＰで印刷を実行するような操作がジョブを実行する機器の変更にあたる。

【００６０】

次に、ジョブの状態について説明する。ジョブの状態には、「実行中」と、「待機中」と、「エラー発生中」と、「中断中」と、「完了」と、「中止」とがある。

【００６１】

「実行中」とは、ジョブが実行されている状態である。「待機中」とは、ジョブが実行されるのを待っている状態である。この状態において、実行の順番が回ってくると、ジョブの状態は「実行中」になる。「エラー発生中」とは、ジョブの実行中に何らかのエラーが発生し、ジョブの実行ができなくなっている状態である。この状態において、エラーが取り除かれるとジョブが再開され、そのジョブの状態は「実行中」になる。

【００６２】

「中断中」とは、ユーザからの操作によりジョブの実行が一時的に止まっている状態である。「完了」とは、ジョブが正常に終了した状態である。「中止」とは、ユーザからの操作でジョブの実行が取り止められた状態である。

【００６３】

次に、図６を用いてジョブの状態遷移について説明する。図６には、状態とイベントが示されており、この図はそれらイベントによる状態遷移を示す図である。イベントには、ジョブの実行、ジョブの完了、再実行、中断、エラー発生、中止の６つある。このうち、ジョブの実行と、ジョブの完了と、エラー発生は、内部イベントである。

【００６４】

基本的に、イベントが中止の場合、ジョブの状態が何であってもジョブの状態は中止６０５に遷移し、そのままジョブが終了するだけとなる。またイベントが再実行の場合、ジョブの状態が何であってもジョブの状態は待機中６０１に遷移する。

【００６５】

通常、ジョブの状態は待機中６０１から、実行中６０２、そして完了６０３と遷移する。状態が待機中６０１のときは、ジョブの実行により実行中６０２に状態が遷移する。また、イベントが中断のときは、状態が中断中６０４に遷移する。

【００６６】

状態が実行中６０２のときは、中断、エラー発生、中止のいずれかイベントにより状態が遷移する。まず、イベントが中断のときは、状態が中断中６０４に遷移する。イベントがエラー発生のときは、状態がエラー発生中６０６に遷移する。

【００６７】

状態が中断中６０４のときは、再実行または中止のイベントにより状態が遷移する。状態がエラー発生中のときは、再実行または中止のイベントにより状態が遷移する。

【００６８】

ジョブの状態遷移は以上のようになっている。次に、ジョブの派生について説明する。

図７は、根幹ジョブから派生した子ジョブを示す図である。また、図７に示されるような、ジョブとジョブとの関係を示したものをジョブ構造といい、ジョブ構造を示す情報をジョブ構造情報という。このジョブ構造情報は、本実施の形態ではジョブ構造テーブルと表現され、ジョブ構造テーブルについては後に詳しく説明する。

【００６９】

従来技術において、根幹ジョブはスプールで処理されると消滅するので、根幹ジョブから派生した子ジョブとの親子関係が保持されることはなかった。そこで、本実施例では、どのジョブも自分から派生した全ての子ジョブの処理が終了するまで消滅しないように保持しておくようにする。

【００７０】

例えば、図７において、従来技術ではジョブＡ４００は、ジョブＡ-Ａ４０１とジョブＡ-Ｂ４０２を派生すると消滅していた。そのため図７のようなジョブ構造がＭＦＰ内では分からなかった。しかし、本実施例において、ジョブＡ４００は子ジョブであるジョブＡ-Ａ４０１とジョブＡ-Ｂ４０２が処理されるまで保持される。また、ジョブＡ-Ａ４０１は子ジョブであるジョブＡ-Ａ-Ａ４０３とジョブＡ-Ａ-Ｂ４０４が処理されるまで保持される。さらに、ジョブＡ-Ｂ４０２は、ジョブＡ-Ｂ-Ａ４０５が処理されるまで保持される。このようにすることにより、図７に示すジョブ構造をＭＦＰ内で構築することが可能となる。

【００７１】

次に、ジョブの派生パターンについて説明する。図８は、１つのジョブから１つのジョブが派生するパターンを示す図である。図８には、１つのプリントジョブ４１０から１つの印刷ジョブ４１１が派生し、その印刷ジョブ４１１から１つのプロッタジョブ４１２が派生していることが示されている。

【００７２】

次に、図９を用いて、１つのジョブから複数のジョブが派生するパターンについて説明する。図９には、１つのプリントジョブ４１５から１つの印刷ジョブ４１６が派生し、その印刷ジョブ４１６から白黒プリントジョブ４１７、カラープリントジョブ４１８という２つのジョブが派生している。さらに白黒プリントジョブ４１７からは２つのプロッタジョブ４１９、４２０が派生している。また、カラープリントジョブ４１８からプロッタジョブ４２１が派生している。

【００７３】

次に、複数のＭＦＰが連携した場合のジョブ構造について説明する。図１０は、２つのＭＦＰが連携した場合のジョブ構造を示している。図１０には、ＭＦＰＡが実行するジョブのジョブ構造２７１と、ＭＦＰＢが実行するジョブのジョブ構造２７２とが示されている。

【００７４】

ジョブ構造２７１には、根幹ジョブである印刷ジョブ２５９と、白黒プリントジョブ２５４と、プロッタジョブ２５５、２５６がある。ジョブ構造２７２には、カラープリントジョブ２５７と、プロッタジョブ２５８がある。

【００７５】

このように、印刷ジョブ２５９から派生した白黒プリントジョブ２５４は、ＭＦＰＡに、同様に派生したカラープリントジョブ２５７は、ＭＦＰＢに割り当てることで、ジョブの分担が可能となる。

【００７６】

なお、図１０には、ジョブを特定するためのジョブＩＤが示されている。ジョブ構造２７１、２７２に属するジョブのジョブＩＤは、そのジョブＩＤをジョブＩＤ全体でユニークなものとされる。

【００７７】

このジョブＩＤは、６ビットとされ、上位３ビットをＭＦＰの識別コードとし、下位３ビットを、そのＭＦＰ内のジョブのなかでユニークな値としている。実際、図１０におい

て、ＭＦＰＡ２５１の識別コードは「００１」とされ、ＭＦＰＢの識別コードは、「００２」とされている。また、各ジョブ構造に属するジョブＩＤの下位３ビットは、各ＭＦＰ内のジョブのなかでユニークなものとなっている。

【００７８】

以上説明したように、ジョブの派生には、１つのジョブから１つのジョブが派生するパターンと、１つのジョブから複数のジョブが派生するパターンの２つのパターンがある。

【００７９】

次に、ジョブが派生する条件について説明する。あるジョブから他のジョブが派生する条件には、処理を委譲することで他のジョブを派生する場合と、処理を分割して管理するために他のジョブを派生する場合とがある。

【００８０】

最初に、図１１を用いて、処理を委譲することで他のジョブを派生する場合について説明する。処理を委譲するとは、あるソフトウェアが自分のジョブを遂行するため、他のソフトウェアに処理の一部を委譲することである。この委譲によりジョブが派生する。

【００８１】

図１１は、コピージョブ４３０の一部の処理を読み取り制御モジュールに委譲することで読み取りジョブ４３１が派生、印刷制御モジュールに委譲することで印刷ジョブ４３２が派生した例を示している。

【００８２】

次に、図１２を用いて、処理を分割して管理するために他のジョブを派生する場合について説明する。処理を分割して管理するとは、１つのソフトウェア内で、あるジョブの一部をそのジョブとは別に管理することである。この分割によりジョブが派生する。図１２は処理を行った印刷ジョブ４３２の一部にエラーが発生したため、その部分を、正常なジョブ４３３、４３５とは別のジョブ４３４として分割し、個別に状態（実行中／エラーなど）を管理、操作（再実行／中止など）を行う場合の例を示している。

【００８３】

処理を分割することによるジョブの派生の他の例を、図１３を用いて説明する。図１３は、複数の宛先に同じ文書をＦＡＸ送信する場合に、ＦＡＸ送信ジョブ４４１を、宛先ごとのＦＡＸ送信ジョブ４４２、４４３、４４４として状態を管理、操作を行う場合の例を示している。

【００８４】

以上がジョブの派生に関する説明である。今までの説明に示されるように、本実施の形態におけるジョブとは、印刷処理、プロッタ処理、白黒プリント処理、カラープリント処理、読み取り処理、ページごとの印刷処理、宛先ごとのＦＡＸ送信処理である。しかし、これらのジョブは、例示列挙であり、他にも多くのジョブが存在する。これらのジョブを本実施の形態で示すような管理を行うことにより、ジョブの中止など、ジョブに対する操作に加え、ジョブごとに課金が可能なるなどのメリットが生じる。

【００８５】

次に、ジョブをどのようにソフトウェアが処理するかは実装方法によって異なるので、その実装方法について説明する。

【００８６】

ここではシングルプロセス・シングルスレッドで実装する場合、シングルプロセス・マルチスレッドで実装する場合、マルチプロセス・シングルスレッドで実装する場合、マルチプロセス・マルチスレッドで実装する場合の４例の実装方法を示す。

【００８７】

また、ここでは、ユーザのジョブへ再実行や中止などの操作をどのように実装するかも示す。なお、シングルプロセス・マルチスレッドとは、プロセスは１つだが、そのなかでは複数のスレッドがあることをいう。また、マルチプロセス・シングルスレッドとは、プロセスは複数あるが、個々のプロセスにはスレッドは１つしかないことをいう。

【００８８】

これらの例で用いられるジョブのジョブ構造を図１４に示す。図１４には、処理の過程でジョブＡ４５０からジョブＢ４５１とジョブＣ４５２が派生し、ジョブＣ４５２からジョブＤ４５３が派生していることが示されている。

【００８９】

まず、図１５を用いて、シングルプロセス・シングルスレッドの場合のジョブ処理の例をについて説明する。図１５には、プロセスＡ５００と、スレッドＡ５０１とが示されている。このソフトウェアのプロセスは１つ（プロセスＡ５００）であり、スレッドもプロセスＡ５００のメインスレッド（スレッドＡ５０１）だけである。この場合、ステップＳ１６０１でジョブＡが実行される。ステップＳ１６０２でジョブＢが実行される。ステップＳ１６０３でジョブＣが実行される。ステップＳ１６０１でジョブＤが実行される。

【００９０】

このように、図１５のようなシングルプロセス・シングルスレッドの場合、ジョブＡ～ジョブＤは全てスレッドＡ５０１で処理される。

【００９１】

次に、図１６を用いて、シングルプロセス・マルチスレッドの場合のジョブ処理の例を説明する。図１６には、プロセスＡ５００と、スレッドＡ５０１、スレッドＢ５０２、スレッドＣ５０３とが示されている。

【００９２】

この場合、ステップＳ１７０１で、プロセスＡ５００のメインスレッドであるスレッドＡ５０１がジョブＡを実行する。このジョブＡから派生したサブスレッドであるジョブＢに対し、スレッドＡがステップＳ１７０２で処理を依頼する。

【００９３】

そして、ステップＳ１７０３で、スレッドＢ５０２がジョブＢを実行し、ステップＳ１７０４で、スレッドＡ５０１に応答を返す。さらにスレッドＡ５０１は、サブスレッドであるスレッドＣ５０３に、ステップＳ１７０５で処理を依頼する。スレッドＣ５０３は、ステップＳ１７０６でジョブＣを実行し、ステップＳ１７０７で、ジョブＤを実行する。そして、スレッドＣ５０３は、ステップＳ１７０８でスレッドＡに応答を返す。

【００９４】

このように、シングルプロセス・マルチスレッドの場合、スレッドＡ５０１は、サブスレッドであるスレッドＢ５０２にジョブＢを実行させ、ジョブＣをサブスレッドであるスレッドＣ５０３に実行させている。さらに、ジョブＣから派生したジョブＤもスレッドＣ５０３に実行させている。

【００９５】

次に、図１７を用いて、マルチプロセス・シングルスレッドの場合の例を説明する。図１７には、プロセスＡ５００と、プロセスＢ５０５と、プロセスＣ５０６と、スレッドＡ５０１、スレッドＢ５０２、スレッドＣ５０３とが示されている。

【００９６】

ステップＳ１８０１で、スレッドＡ５０１は、ジョブＡを実行する。ステップＳ１８０２で、スレッドＡ５０１は、プロセスＢ５０５のスレッドＢ５０２に、処理を依頼する。スレッドＢ５０２は、ステップＳ１８０３で、ジョブＢを実行し、ステップＳ１８０４で、スレッドＡ５０１に応答を返す。

【００９７】

スレッドＡ５０１は、ステップＳ１８０５で、プロセスＣ５０６のスレッドＣ５０３に処理を依頼する。スレッドＣ５０３は、ステップＳ１８０６で、ジョブＣを実行し、さらにステップＳ１８０７で、ジョブＤを実行し、ステップＳ１８０８で、スレッドＡに応答を返す。

【００９８】

このように、マルチプロセス・シングルスレッドの場合、プロセスＡ５００のメインスレッドであるスレッドＡ５０１でジョブＡを実行する。そして、スレッドＡ５０１は、ジョブＡから派生したジョブＢをプロセスＢ５０５のメインスレッドであるスレッドＢ５０

２に、ジョブＣをプロセスＣ５０６のメインスレッドであるスレッドＣ５０３に実行させている。さらに、スレッドＡ５０１は、ジョブＣから派生したジョブＤもスレッドＣ５０３に実行させている。

【００９９】

次に、図１８を用いて、マルチプロセス・マルチスレッドの場合の例を説明する。図１８には、プロセスＡ５００と、プロセスＢ５０５と、スレッドＡ５０１、スレッドＢ５０２、スレッドＣ５０３、スレッドＤ５０７とが示されている。

【０１００】

ステップＳ１９０１で、プロセスＡ５００のスレッドＡ５０１は、ジョブＡを実行する。ステップＳ１９０２でスレッドＡ５０１は、同じプロセスＡ５００内のスレッドＢ５０２に処理を依頼する。スレッドＢ５０２は、ステップＳ１９０３で、ジョブＢを実行し、ステップＳ１９０４で、スレッドＡ５０１に応答を返す。

【０１０１】

スレッドＡ５０１は、ステップＳ１９０５で、プロセスＢ５０５のスレッドＣ５０３に処理を依頼する。スレッドＣ５０３は、ステップＳ１９０６で、ジョブＣを実行する。ステップＳ１９０７でスレッドＣ５０３は、同じプロセスＢ５０５内のスレッドＤ５０７に処理を依頼する。スレッドＤ５０７は、ステップＳ１９０８で、ジョブＤを実行し、ステップＳ１９０９で、スレッドＣ５０３に応答を返す。スレッドＣ５０３は、ステップＳ１９１０で、スレッドＡ５０１に応答を返す。

【０１０２】

このように、マルチプロセス・マルチスレッドの場合、プロセスＡ５００のメインスレッドであるスレッドＡ５０１でジョブＡが処理される。また、スレッドＡ５０１は、ジョブＡから派生したジョブＢをプロセスＡ５００のサブスレッドであるスレッドＢ５０２に実行させる。また、スレッドＡ５０１は、ジョブＡから派生したジョブＣをプロセスＢ５０５のメインスレッドであるスレッドＣ５０３に実行させ、ジョブＣから派生したジョブＤをプロセスＢ５０５のサブスレッドであるスレッドＤ５０７に実行させている。

【０１０３】

以上が４例の実装方法である。これらの実装方法に加え、さらにマルチプロセス・マルチスレッドという実装方法において、ユーザからの要求によりジョブの処理を操作する例を、図１９を用いて説明する。図１９には、プロセスＡ５００と、プロセスＢ５０５と、プロセスＣ５０６と、スレッドＡ５０１と、スレッドＢ５０２と、スレッドＣ５０３と、スレッドＥ５０８とが示されている。

【０１０４】

このうち、プロセスＣがユーザからの要求が行われるプロセスとなっている。ステップＳ２００１で、プロセスＡ５００のスレッドＡ５０１は、ジョブＡを実行する。ステップＳ２００２でスレッドＡ５０１は、同じプロセスＡ５００内のスレッドＢ５０２に処理を依頼する。スレッドＢ５０２は、ステップＳ２００３で、ジョブＢを実行し、ステップＳ２００４で、スレッドＡ５０１に応答を返す。

【０１０５】

スレッドＡ５０１は、ステップＳ２００５で、プロセスＢ５０５のスレッドＣ５０３に処理を依頼する。スレッドＣ５０３は、ステップＳ２００６で、ジョブＣを実行する。

【０１０６】

このとき、ステップＳ２００７でプロセスＣ５０６のスレッドＥ５０８は、スレッドＣ５０３に処理の中止を依頼する。スレッドＣ５０３は、ステップＳ２００８で、ジョブＣの中止処理を実行し、ステップＳ２００９で、スレッドＡ５０１に応答を返す。

【０１０７】

このように、ユーザからの操作の場合、操作の受付用のプロセス（プロセスＣ５０６）が存在し、ユーザからの中止要求を受け付け、該当するジョブに中止依頼を出すようになっている。

【０１０８】

次に、各ジョブに関する情報であるジョブ情報について、図２０を用いて説明する。ジョブ情報は、図２０に示されるように、識別情報に対応するジョブＩＤと、文書名と、ジョブ種別と、状態と、オーナー名と、作成日と、ジョブ関連情報とを有する。

【０１０９】

このうち、ジョブ関連情報が、ジョブのジョブ構造をＭＦＰ内で構築することを可能とするものであり、このジョブ関連情報については後に詳しく説明することとする。ジョブ構造を構築することが可能であれば、ジョブの親子関係が分かる。

【０１１０】

上記ジョブ情報のうち、ジョブＩＤは、「１」や「２」など、各ジョブを識別するもので、各ジョブに一意的に割り当てられるものである。以下の説明では、ジョブＩＤがｎのジョブを単にジョブｎと表現することがある。

【０１１１】

文書名は、印刷する文書の名前を示すもので、例えば文字列で表現される。ジョブ種別は、「印刷」、「ＦＡＸ送信」など、どのような処理のジョブかを示すものである。「状態」は、「実行中」や「エラー」など、ジョブの状態を示すものである。「オーナー名」は、「鈴木」、「佐藤」など、ジョブの実行を依頼したユーザ名を示すものである。「作成日」は、「2004.02.17.12：15」など、ジョブが作成された日時を示すものである。

【０１１２】

ジョブ情報の他の例を、図２１を用いて説明する。図２１に示されるジョブ情報と、図２０に示されるジョブ情報とが異なる大きな点は、「ジョブの操作の許可情報」と「ジョブを操作できる条件」なる２つの項目である。

【０１１３】

これらの項目は、ジョブの操作、ジョブの状態閲覧を可能とするための項目である。個々のジョブは実行中やエラー発生中などの状態を持ち、そのジョブに対してユーザが再実行や中止などの操作を行うことができる。

【０１１４】

しかし、ジョブによってはユーザに操作させなかったり、ある条件を満たす時に限り操作を許したりするものもある。そのような形にジョブを実装する場合、ジョブ情報に図２１に示すような「ジョブの操作の許可情報」や「ジョブを操作できる条件」という属性を付加する必要がある。

【０１１５】

このうち、「ジョブの操作の許可情報」について説明する。この「ジョブの操作の許可情報」が取り得る値には「許可」、「不許可」、「条件発生時のみ許可」がある。

【０１１６】

「許可」とは、どのような場合でもそのジョブの状態に見合った操作（再実行／中止／中断など）が行えることを示す。「不許可」とは、どのような場合でもそのジョブに対しては操作が行えないことを示す。「条件発生時のみ許可」とは、図２１に示される「ジョブを操作できる条件」が成就した時に、その条件に見合った操作を行えることを示す。具体的に、操作として、エラー発生の場合の再実行または中止が挙げられる。

【０１１７】

これらの値のうち、「不許可」の場合のジョブ情報を示すのが、図２２に示されるジョブ情報である。このジョブ情報の「ジョブの操作の許可情報」は「不許可」となっている。また、「条件発生時のみ許可」の場合のジョブ情報を示すのが、図２３に示されるジョブ情報である。このジョブ情報の「ジョブの操作の許可情報」は「条件発生時のみ許可」となっており、「ジョブを操作できる条件」が「実行中＆エラー発生中」であることが示されている。

【０１１８】

なお、「ジョブを操作できる条件」は、「エラー発生中」、「実行中またはエラー発生中」、「待機中またはエラー発生中」、「実行中または待機中＆エラー発生中」の値もとり得る。「エラー発生中」は、ジョブがエラー発生中という条件である。「実行中または

エラー発生中」は、ジョブが実行中もしくはジョブがエラー発生中という条件である。「待機中またはエラー発生中」は、ジョブが待機中もしくはジョブがエラー発生中という条件である。「実行中または待機中またはエラー発生中」は、ジョブが実行中、ジョブが待機中、ジョブがエラー発生中のうちのいずれか条件である。

【０１１９】

次に、ジョブ関連情報に基づき作成されるジョブ構造テーブルについて、図２４を用いて説明する。ジョブ構造テーブル１００は、親ジョブと子ジョブのＩＤからなるものである。「子ジョブ」は、ジョブＩＤを示し、「親ジョブ」は、子ジョブで示されたジョブＩＤを有するジョブの親ジョブのジョブＩＤを示している。従って、親ジョブが記入されていないジョブは、親ジョブが存在しないジョブなので、根幹ジョブということになる。

【０１２０】

また、ジョブ構造１０１、１０２は、ジョブ構造テーブル１００をジョブ構造として表現したものである。

【０１２１】

図２４の場合、ジョブ１と、ジョブ８は根幹ジョブであることが分かる。また、ジョブ構造テーブル１００には、ジョブ２、３のジョブの親ジョブのＩＤが１と示されてあるので、ジョブ２、３は、ジョブ１から派生したジョブであることが分かる。同様に、他のジョブに関しても、ジョブ構造テーブル１００に示される各ジョブＩＤにより、図２４に示されるジョブ構造を生成できる。

【０１２２】

このように、ジョブの親ジョブのジョブＩＤを各ジョブごとに保持しておくとことで、ジョブ構造テーブルを作成することができる。また、このジョブ構造テーブルにより、パソコンの画面、あるいはＭＦＰに設けられているオペレーションパネルにジョブ構造を表示することができる。

【０１２３】

なお、ジョブ構造テーブルは、図２５に示されるように、ＸＭＬで表現することもできる。図２５に示されるＸＭＬ文は、図２４で示した根幹ジョブで、ジョブ８から派生したジョブのジョブ構造１０２を生成するためものである。

【０１２４】

この図２５より、ジョブ８は、ジョブ９、１０、１１のジョブを、子ジョブあるいは孫ジョブとしていることが分かる。同様に、ジョブ９は、ジョブ８を親ジョブとし、ジョブ１０、１１を子ジョブとしていることが分かる。さらにジョブ１０、１１は、ともにジョブ９を親ジョブとしていることが分かる。

【０１２５】

このようにジョブ構造テーブルは、ジョブＩＤを入れ子とするＸＭＬ文で表現することができる。また、ジョブ構造テーブルは、任意のジョブと他のジョブとの関連を示していることがわかる。

【０１２６】

以上が処理の概要である。次に、各処理の詳細な説明をする。まず、図２６を用いてＭＦＰに発生するジョブを、コピーを例にして説明する。図２６には、コピー蓄積ジョブ１１０と、読み取りジョブ１１１と、プリントジョブ１１２と、蓄積ジョブ１１３と、プロッタジョブ１１４、１１５が示されている。このジョブ構造における根幹ジョブは、コピー蓄積ジョブ１１０である。

【０１２７】

コピー蓄積の要求に対して、コピー蓄積ジョブ１１０が発生し、それを実現するために読み取りジョブ１１１、プリントジョブ１１２、蓄積ジョブ１１３が派生する。プリントジョブ１１２からは、さらにプロッタジョブ１１４、１１５が派生する。

【０１２８】

次に、図２７を用いて、ＭＦＰのソフトウェアブロックについて説明する。この図２７で説明される「アプリ」、「モジュール」、「ハンドラ」をまとめて、プログラムという

ことにする。

【０１２９】

図２７に示されるソフトウェアブロックは、アプリケーション層５と、サービス層７と、ハンドラ層９の３つの層に分けられる。アプリケーション層５は、コピーアプリ２１と、プリンタアプリ２０と、ＦＡＸアプリ２２などのプログラムを含む。サービス層７は、印刷制御モジュール２３と、ＦＡＸ制御モジュール２４と、読み取り制御モジュール２５と、ジョブ管理モジュール２６のプログラムを含む。ハンドラ層９は、プロッタハンドラ２７と、ＦＡＸユニットハンドラ２８と、スキャナハンドラ２９と、メモリ管理モジュール３０のプログラムを含む。

【０１３０】

アプリケーション層５のコピーアプリ２１、プリンタアプリ２０、ＦＡＸアプリ２２は、それぞれ、コピー用、プリンタ用、ＦＡＸ用のアプリケーションである。

【０１３１】

次に、サービス層７のモジュールについて説明する。印刷制御モジュール２３は、印刷処理を制御するモジュールである。ＦＡＸ制御モジュール２４は、ＦＡＸ処理を制御するモジュールである。読み取り制御モジュール２５は、読み取り処理を制御するモジュールである。全体ジョブ構造情報作成手段とジョブ検索手段に対応するジョブ管理モジュール２６は、要求によりジョブ構造テーブル（ジョブ構造情報）を作成して要求元に渡すモジュールである。

【０１３２】

次に、ハンドラ層９のプログラムについて説明する。ハンドラ層９のプログラムは、ハードウェアであるプロッタやスキャナなどのハンドラなどである。

【０１３３】

プロッタハンドラ２７は、プロッタのハンドラである。ＦＡＸユニットハンドラ２８は、ＦＡＸユニットのハンドラである。スキャナハンドラ２９は、スキャナのハンドラである。メモリ管理モジュール３０は、メモリやハードディスクの管理を行うハンドラである。

【０１３４】

次に、ＭＦＰのハードウェア構成図を、図２８を用いて説明する。ＭＦＰは、コントローラボード６０と、オペレーションパネル５３と、ＦＣＵ６８と、エンジン７１と、スキャナ５１と、プロッタ５２とを含む。また、ＦＣＵ６８は、Ｇ３規格対応ユニット６９と、Ｇ４規格対応ユニット７０とを有する。

【０１３５】

また、コントローラボード６０は、ＣＰＵ６１と、ＡＳＩＣ６６と、ＨＤＤ６５と、ローカルメモリ（ＭＥＭ－Ｃ）６４と、システムメモリ（ＭＥＭ－Ｐ）６３と、ノースブリッジ（以下、ＮＢと記す）６２と、サウスブリッジ（以下、ＳＢと記す）７３と、ＮＩＣ７４（Network Interface Card）と、ＵＳＢデバイス７５と、ＩＥＥＥ１３９４デバイス７６と、セントロニクスデバイス７７とを含む。

【０１３６】

オペレーションパネル５３は、コントローラボード６０のＡＳＩＣ６６に接続されている。また、ＳＢ７３と、ＮＩＣ７４と、ＵＳＢデバイス７５と、ＩＥＥＥ１３９４デバイス７６と、セントロニクスデバイス７７と、ＮＢ６２にＰＣＩバスで接続されている。

【０１３７】

また、ＦＣＵ６８と、エンジン７１と、スキャナ５１、プロッタ５２は、コントローラボード６０のＡＳＩＣ６６にＰＣＩバスで接続されている。

【０１３８】

なお、コントローラボード６０は、ＡＳＩＣ６６にローカルメモリ６４、ＨＤＤ６５などが接続されると共に、ＣＰＵ６１とＡＳＩＣ６６とがＣＰＵチップセットのＮＢ６２を介して接続されている。このように、ＮＢ６２を介してＣＰＵ６１とＡＳＩＣ６６とを接続すれば、ＣＰＵ６１のインタフェースが公開されていない場合に対応できる。

【０１３９】
なお、ＡＳＩＣ６６とＮＢ６２とはＰＣＩバスを介して接続されているのでなく、ＡＧＰ（Accelerated Graphics Port）６７を介して接続されている。このように、図２７のアプリケーション層５などを形成する一つ以上のプロセスを実行制御するため、ＡＳＩＣ６６とＮＢ６２とを低速のＰＣＩバスでなくＡＧＰ６７を介して接続し、パフォーマンスの低下を防いでいる。
【０１４０】
ＣＰＵ６１は、ＭＦＰの全体制御を行うものである。ＣＰＵ６１は、アプリケーション層５、サービス層７、ハンドラ層９に含まれるプログラムをＯＳ上にそれぞれプロセスとして起動して実行させる。
【０１４１】
ＮＢ６２は、ＣＰＵ６１、システムメモリ６３、ＳＢ７３およびＡＳＩＣ６６を接続するためのブリッジである。システムメモリ６３は、ＭＦＰの描画用メモリなどとして用いるメモリである。ＳＢ７３は、ＮＢ６２とＰＣＩバス、周辺デバイスとを接続するためのブリッジである。また、ローカルメモリ６４はコピー用画像バッファ、符号バッファとして用いるメモリである。
【０１４２】
ＡＳＩＣ６６は、画像処理用のハードウェア要素を有する画像処理用途向けのＩＣである。ＨＤＤ６５は、画像データの蓄積、文書データの蓄積、プログラムの蓄積、フォントデータの蓄積、フォームの蓄積などを行うためのストレージである。また、オペレーションパネル５３は、ユーザからの入力操作を受け付けると共に、ユーザに向けた表示を行う操作部である。
【０１４３】
次に、上述した各ソフトウェアの関係と、各ソフトウェア間でやり取りされる情報について、図２９と図３０を用いて説明する。図２９は、ジョブ情報がＨＤＤ６５に記憶される場合を示す図であり、図３０は、ジョブ情報がメモリ上に記憶される場合を示す図である。
【０１４４】
まず、図２９の説明をする。図２９には、各プログラム１３０と、メモリ管理モジュール３０と、ＨＤＤ６５と、ジョブ情報１３２と、ジョブ管理モジュール２６と、表示アプリ３２と、専用キュー１３１とが示されている。
【０１４５】
各プログラム１３０は、アプリ層、サービス層、ハンドラ層の各プログラムである。具体的には、ジョブ管理モジュールとメモリ管理モジュール以外のプログラムであり、ジョブを発生・処理するプログラムである。表示アプリ３２は、パソコンで表示するＰＣアプリや、上述したオペレーションパネル用のアプリである。ジョブ情報１３２は、図２０で説明したジョブ情報であり、図２９の場合、ＨＤＤ６５に記憶されている。専用キュー１３１は、各プログラム用のメモリ空間に記憶される各プログラムがジョブを管理する専用のキューである。
【０１４６】
なお、各プログラムのメモリ空間とは、そのプログラムにそれぞれ割り当てられたメモリ領域を指す。ここに各モジュールは専用のキューを持っており、処理するジョブの順番を管理している。
【０１４７】
各プログラム１３０からメモリ管理モジュール３０へはジョブ情報が提供される。メモリ管理モジュール３０は、ジョブ情報をＨＤＤ６５に記憶する。また、メモリ管理モジュール３０は、ジョブ管理モジュール２６にジョブ情報を提供する。ジョブ管理モジュール２６は、表示アプリ３２にジョブ構造テーブルとジョブ情報を提供する。各プログラム１３０は、ジョブ管理モジュール２６にキューの情報を提供する。また、各プログラム１３０は、専用キュー１３１にジョブＩＤを記憶させる。

【０１４８】
次に、図３０の説明をする。なお、図２９で説明した参照符号と同じものは、説明を省略する。図３０と図２９で異なるのは、キュー１３３とジョブ情報１３２である。キュー１３３とジョブ情報１３２は、図３０に示されるように、各プログラム用のメモリ空間に記憶される。従って、キュー１３３とジョブ情報１３２は、各プログラム専用のものとなる。
【０１４９】
各プログラム１３０は、キュー１３３とジョブ情報１３２にそれぞれジョブＩＤとジョブ情報を記憶させる。また、各プログラム１３０は、ジョブ管理モジュール２６にキューの情報とジョブ情報を提供する。ジョブ管理モジュール２６は、表示アプリ３２にジョブ構造テーブルとジョブ情報を提供する。この表示アプリ３２は、ＧＵＩ表示手段とジョブ構造情報取得手段に対応する。
【０１５０】
この図３０の場合、ジョブ情報はＨＤＤではなく各プログラムが管理するメモリ空間で保存されている。よって、ジョブ管理モジュール６２はジョブを持っている各プログラムからジョブ情報を取得し、ジョブ構造テーブルを作成する。
【０１５１】
次に、ジョブ関連情報について説明する。このジョブ関連情報は上述したように、ジョブ構造テーブルの作成に必要なものである。このジョブ関連情報は、ジョブ構造テーブルを作成できればよいので、いくつかの種類が考えられる。以下、７種類のジョブ関連情報の例とそのときの処理を示すシーケンス図について説明する。シーケンス図で説明する処理は、印刷要求を行ったときのジョブ登録処理と、ジョブの参照を行ったときの処理である。
【０１５２】
なお、それぞれで示されるジョブ情報は図２０または図２１で説明した構造を例としている。また、以下の説明で用いられるジョブ構造は、図３１に示されるジョブ構造とする。図３１に示されるジョブ構造は、ジョブＡから派生したジョブで構成されるジョブのジョブ構造である。
【０１５３】
第１のジョブ関連情報例について、図３２を用いて説明する。図３２に示されるジョブ関連情報は、子ジョブの親ジョブのジョブＩＤから根幹ジョブのジョブＩＤまで、というように、子ジョブが全ての先祖のジョブのジョブＩＤを持っているものである。
【０１５４】
具体的に、図３３に示されるジョブ情報を用いて説明する。この図３３に示されるジョブ情報は、図３１のジョブ７のジョブ情報である。
【０１５５】
図３３には、ジョブ情報１５０と、ジョブ関連情報１５１とが示されている。ジョブ情報１５０は、図２０で説明したものである。ジョブ関連情報１５１は、項目に「根幹ジョブのジョブＩＤ」と「２世代目のジョブＩＤ」と、「親ジョブのジョブＩＤ」があり、それぞれ値が「１」、「２」、「４」となっている。
【０１５６】
実際に、ジョブ７の親ジョブはジョブ４であり、２世代目のジョブはジョブ２であり、根幹ジョブはジョブ１である。
【０１５７】
このように、子ジョブが全ての先祖のジョブのジョブＩＤを持っており、ジョブ管理モジュールが末端ジョブを識別する仕組みが構築されると、ジョブ管理モジュールは末端ジョブの情報を取得するだけで、ジョブ構造テーブルを作成できる。また、アプリ層、サービス層、ハンドラ層の各プログラムは、ジョブ生成時に、親ジョブのジョブ構造テーブルに、親のジョブＩＤを追記するだけで、ジョブ関連情報を作成できる。なお、ジョブ管理モジュールが末端ジョブを識別する仕組みとして、ジョブ情報に末端ジョブかを判別する

フラグを設けるなどの仕組みが挙げられる。

【０１５８】

以上説明した第１のジョブ関連情報例の場合の処理を、シーケンス図を用いて説明する。図３４は、ジョブ登録時の動作シーケンスを示すもので、ユーザ１６０と、プリンタアプリ２０と、印刷制御モジュール２３と、プロッタハンドラ２７と、メモリ管理モジュール３０との間で行われる処理を示している。このことは、以降のジョブ登録時の動作シーケンス図でもほぼ同じである。

【０１５９】

ステップＳ１０１で、プリンタアプリ２０は、ユーザ１６０から印刷要求を通知される。プリンタアプリ２０は、ステップＳ１０２で、ジョブの生成を行い、ステップＳ１０３で、ジョブＩＤをキューに追加する。ステップＳ１０４で、プリンタアプリ２０は、メモリ管理モジュールにジョブ情報を通知することで、ジョブ情報を保存する。

【０１６０】

ステップＳ１０５で、プリンタアプリ２０は、キューからジョブＩＤを取得する。ステップＳ１０７で、メモリ管理モジュール３０は、プリンタアプリ２０にジョブ情報を通知する。ステップＳ１０８は、プリンタアプリ２０の処理の実行である。

【０１６１】

ステップＳ１０９で、プリンタアプリ２０は、印刷制御モジュール２３に印刷処理要求を通知する。このとき、親ジョブのジョブ関連情報と、親ジョブのジョブＩＤも通知される。印刷モジュール２３は、ステップＳ１１０でジョブの生成を行い、ステップＳ１１１でジョブＩＤをキューに追加する。そして、ステップＳ１１２で印刷制御モジュール２３は、ジョブ情報を保存する。次のステップＳ１１３で、印刷制御モジュール２３は、ジョブＩＤをキューから取得する。

【０１６２】

印刷制御モジュール２３は、ステップＳ１１４で、メモリ管理モジュール３０にジョブ情報の取得を要求する。メモリ管理モジュール３０は、ステップＳ１１５で、印刷制御モジュール２３にジョブ情報を通知する。

【０１６３】

ステップＳ１１６は、印刷制御モジュール２３の処理の実行である。ステップＳ１１７で印刷制御モジュール２３は、プロッタハンドラ２７にプロット要求を通知する。このとき、親ジョブのジョブ関連情報と親ジョブＩＤも通知される。

【０１６４】

ステップＳ１１８で、プロッタハンドラ２７は、ジョブの生成を行う。ステップＳ１１９で、プロッタハンドラ２７は、ジョブＩＤをキューに追加し、ステップＳ１２０で、ジョブ情報を保存する。ステップＳ１２１で、プロッタハンドラ２７は、ジョブＩＤをキューから取得する。ステップＳ１２２で、プロッタハンドラ２７は、メモリ管理モジュール３０にジョブ情報の取得を要求する。ステップＳ１２３で、メモリ管理モジュール３０は、プロッタハンドラ２７にジョブ情報を通知する。

【０１６５】

プロッタハンドラ２７は、ステップＳ１２４で、プロットを実行する。プロッタハンドラ２７は、印刷制御モジュール２３に、ステップＳ１２５で、プロット終了を通知する。印刷制御モジュール２３は、ステップＳ１２６で、プリンタアプリ２０に印刷終了を通知する。プリンタアプリ２０は、ステップＳ１２７で、ユーザ１６０に印刷終了を通知（表示）する。

【０１６６】

次に、図３５を用いて、ジョブ参照時の動作シーケンスを説明する。図３５は、ユーザ１６０と、オペパネアプリ１６１と、ジョブ管理モジュール２６と、メモリ管理モジュール３０と、プリンタアプリ２０との間で行われる処理を示している。このことは、以降のジョブ参照時の動作シーケンス図でもほぼ同じである。

【０１６７】

ステップＳ２０１で、オペパネアプリ１６１は、ユーザ１６０からジョブ参照要求を通知される。ステップＳ２０２で、オペパネアプリ１６１は、ジョブ管理モジュール２６にジョブ一覧要求を通知する。ジョブ管理モジュール２６は、ステップＳ２０３で、メモリ管理モジュール３０に末端ジョブの検索を通知する。ステップＳ２０４で、メモリ管理モジュール３０は、ジョブ管理モジュール２６に末端ジョブの検索結果を通知する。ステップＳ２０５で、ジョブ管理モジュール２６は、メモリ管理モジュール３０にジョブ情報の取得を要求する。ステップＳ２０６で、メモリ管理モジュール３０は、ジョブ管理モジュール２６にジョブ情報を通知する。ジョブ管理モジュール２６は、ステップＳ２０７で、ジョブ一覧の作成を行う。

【０１６８】

ジョブ構造受信段階に対応するステップＳ２０８で、ジョブ管理モジュール２６は、オペパネアプリ１６１にジョブ一覧を通知する。オペパネアプリ１６１は、ＧＵＩ表示段階に対応するステップＳ２０９で、ユーザ１６０に、ジョブ一覧を通知（表示）する。ステップＳ２１０で、オペパネアプリ１６１は、ユーザ１６０からジョブ情報を要求される。ステップＳ２１１で、オペパネアプリ１６１は、ジョブ管理モジュール２６にジョブ情報の取得を要求する。ジョブ管理モジュール２６は、ステップＳ２１２で、メモリ管理モジュール３０にジョブ情報の取得を要求する。メモリ管理モジュール３０は、ステップＳ２１３で、ジョブ管理モジュール２６にジョブ情報を通知する。

【０１６９】

ステップＳ２１４で、ジョブ管理モジュール２６は、プリンタアプリ２０に、キューの順番取得を要求する。プリンタアプリ２０は、ステップＳ２１５で、ジョブ管理モジュール２６にキューの順番を通知する。ジョブ管理モジュール２６は、ステップＳ２１６で、オペパネアプリ１６１にジョブ情報を通知する。ステップＳ２１７で、オペパネアプリ１６１は、ユーザ１６０にジョブ情報を通知（表示）する。

【０１７０】

次に、第２のジョブ関連情報例について、図３６を用いて説明する。第２のジョブ関連情報例は、図３６に示されるように、ジョブ情報内のジョブ関連情報に、親ジョブのジョブＩＤのみを持っているものである。

【０１７１】

具体的に、図３７に示されるジョブ情報を用いて説明する。この図３７に示されるジョブ情報は、図３１のジョブ７のジョブ情報である。ジョブ関連情報は、項目に「親ジョブのジョブＩＤ」のみであり、値が「４」となっている。

【０１７２】

このように、ジョブ関連情報が親ジョブのジョブＩＤのみであれば、ジョブ情報の情報量を抑えることが可能となる。この場合、ジョブ管理モジュールが全ジョブにアクセスする仕組みが必要となる。

【０１７３】

以上説明した第２のジョブ関連情報例の場合のジョブ登録時の動作を、図３８のシーケンス図を用いて説明する。なお、このシーケンス図は、図３４で示したシーケンス図でのステップのうち、２つのステップ以外は同じであるので、異なるステップ以外の説明は省略する。

【０１７４】

異なるステップは、図３４におけるステップＳ１０９とステップＳ１１７であり、図３８では、それぞれステップＳ３０９とステップＳ３１７に対応する。

【０１７５】

ステップＳ１０９では、親ジョブのジョブ関連情報が通知されるが、ステップＳ３０９では、親ジョブのジョブＩＤのみが通知される。また、ステップＳ１１７も同様に、親ジョブのジョブ関連情報と親ジョブのジョブＩＤの２つが通知されるが、ステップＳ３１７では、親ジョブのジョブＩＤのみが通知される。

【０１７６】

いずれのステップの処理も、ジョブ関連情報が親ジョブのジョブＩＤとなったことによるものである。

【０１７７】

次に、図３９を用いて、ジョブ参照時の動作シーケンスを説明する。なお、このシーケンス図は、図３５で示したシーケンス図のステップＳ２０１、２０２は、ステップＳ４０１、４０２に対応し、ステップＳ２０５からステップＳ２１７は、ステップＳ４０４からステップＳ４１５に対応する。

【０１７８】

従って、図３９に示される処理は、図３５の処理から、末端ジョブの検索処理であるステップＳ２０３、２０４の処理を除いたものである。これは、ジョブ管理モジュールが全ジョブにアクセスするため、特定のジョブの検索の必要がないためである。

【０１７９】

次に、第３のジョブ関連情報例について、図４０を用いて説明する。図４０に示されるジョブ関連情報は、ジョブが全ての子孫ジョブのジョブＩＤを持っているものである。

【０１８０】

具体的に、図４１に示されるジョブ情報を用いて説明する。図４１には、ジョブ情報１６７と、ジョブ関連情報１６８とが示されている。ジョブ情報１６７に示されるように、このジョブ情報は、図３１に示されているジョブ７のジョブ情報である。

【０１８１】

ジョブ関連情報１６８は、値に「子ジョブ」と「親ジョブ」がある。子ジョブで示されるジョブＩＤの親ジョブのジョブＩＤが親ジョブの欄に記載される。

【０１８２】

実際に、ジョブ２、３の親ジョブはジョブ１であり、ジョブ４、５の親ジョブは、ジョブ２であり、ジョブ６の親ジョブはジョブ３であり、ジョブ７の親ジョブはジョブ４である。

【０１８３】

このように、ジョブ関連情報が子孫ジョブのジョブＩＤあれば、ジョブ管理モジュールは、根幹ジョブのジョブ情報を取得するだけでジョブ構造テーブルの作成が可能となる。この場合、ジョブを生成したモジュールが、先祖のジョブを生成した全モジュールにジョブＩＤを通知する仕組みが必要である。さらに、ジョブ管理モジュールが保存されているジョブから根幹ジョブを探す仕組みが必要である。

【０１８４】

以上説明した第３のジョブ関連情報例の場合のジョブ登録時の動作を、図４２のシーケンス図を用いて説明する。なお、このシーケンス図は、図３４で示したシーケンス図のステップＳ１０１からステップＳ１０８までの処理と、ステップＳ５０１からステップＳ５０８までの処理が同じなので説明を省略する。

【０１８５】

ステップＳ５０９で、プリンタアプリ２０は、印刷制御モジュール２３に印刷処理要求を通知する。このとき、親ジョブのジョブ関連情報と、親ジョブのジョブＩＤは通知されない。印刷モジュール２３は、ステップＳ５１０でジョブの生成を行い、ステップＳ５１１でプリンタアプリ２０に子ジョブ生成を通知する。次のステップＳ５１２で、プリンタアプリ２０は、ジョブＩＤをキューに追加する。そして、ステップＳ５１３で印刷制御モジュール２３は、ジョブ情報を保存する。次のステップＳ５１４で、印刷制御モジュール２３は、ジョブＩＤをキューから取得する。

【０１８６】

印刷制御モジュール２３は、ステップＳ５１５で、メモリ管理モジュール３０にジョブ情報の取得を要求する。メモリ管理モジュール３０は、ステップＳ５１６で、印刷制御モジュール２３にジョブ情報を通知する。

【０１８７】

ステップＳ５１７は、印刷制御モジュール２３の処理の実行である。ステップＳ５１８

で印刷制御モジュール２３は、プロッタハンドラ２７にプロット要求を通知する。ステップＳ５１９で、プロッタハンドラ２７は、ジョブの生成を行う。

【０１８８】

ステップＳ５２０で、プロッタハンドラ２７は、印刷制御モジュール２３に、ジョブＩＤとともに子ジョブの生成を通知する。ステップＳ５２１で、印刷制御モジュール２３は、プリンタアプリ２０に子孫ジョブの生成を通知する。このとき、親ジョブＩＤと子ジョブＩＤも通知される。

【０１８９】

ステップＳ５２２で、プロッタハンドラ２７は、ジョブＩＤをキューに追加し、ステップＳ５２３で、ジョブ情報を保存する。ステップＳ５２４で、プロッタハンドラ２７は、ジョブＩＤをキューから取得する。ステップＳ５２５で、プロッタハンドラ２７は、メモリ管理モジュール３０にジョブ情報の取得を要求する。ステップＳ５２６で、メモリ管理モジュール３０は、プロッタハンドラ２７にジョブ情報を通知する。

【０１９０】

プロッタハンドラ２７は、ステップＳ５２７で、プロットを実行する。プロッタハンドラ２７は、印刷制御モジュール２３に、ステップＳ５２８で、プロット終了を通知する。印刷制御モジュール２３は、ステップＳ５２９で、プリンタアプリ２０に印刷終了を通知する。プリンタアプリ２０は、ステップＳ５３０で、ユーザ１６０に印刷終了を通知（表示）する。

【０１９１】

次に、図４３を用いて、ジョブ参照時の動作シーケンスを説明する。なお、このシーケンス図は、図３５で示したシーケンス図でのステップのうち、２つのステップ以外は同じであるので、異なるステップ以外の説明は省略する。

【０１９２】

異なるステップは、図３５におけるステップＳ２０３とステップＳ２０４であり、図４３では、それぞれステップＳ６０３とステップＳ６０４に対応する。図３５の場合、検索の対象は、末端ジョブであったが、図４３では、検索の対象が根幹ジョブとなる。

【０１９３】

次に、第４のジョブ関連情報例について、図４４を用いて説明する。図４４に示されるジョブ関連情報は、ジョブが全ての子ジョブのジョブＩＤを持っているものである。

【０１９４】

具体的に、図４５に示されるジョブ情報を用いて説明する。図４５には、ジョブ情報１６３と、ジョブ関連情報１６４とが示されている。ジョブ情報１６３に示されるように、このジョブ情報は、図３１に示されているジョブ１のジョブ情報である。

【０１９５】

ジョブ関連情報１６４には、項目に２つの「子ジョブのジョブＩＤ」があり、それぞれ値が「２」、「３」となっている。実際に、ジョブ１の子ジョブはジョブ２とジョブ３である。

【０１９６】

このように、ジョブ関連情報が子ジョブのジョブＩＤのみであれば、ジョブ情報の情報量を抑えることが可能となる。この場合、ジョブ管理モジュールが全ジョブにアクセスする仕組みが必要となる。

【０１９７】

以上説明した第４のジョブ関連情報例の場合の処理を、図４６のシーケンス図を用いて説明する。なお、このシーケンス図は、図４２で示したシーケンス図のステップＳ５２１の処理を行わないこと以外は、処理が同じなので説明を省略する。これは、第４のジョブ関連情報は、子孫ジョブを必要としないので、子孫ジョブの生成を通知する図４２のステップＳ５２１の処理は不要となるからである。

【０１９８】

ジョブ参照時の動作シーケンスは、図３９と同じであるため、説明を省略する。

【０１９９】
次に、第１、４のジョブ関連情報を組み合わせた第５のジョブ関連情報例について、図４７を用いて説明する。図４７に示されるジョブ関連情報は、ジョブが全ての先祖のジョブＩＤと、全ての子ジョブのジョブＩＤからなる。
【０２００】
具体的に、図４８に示されるジョブ情報を用いて説明する。図４８には、ジョブ情報１６６と、ジョブ関連情報１６９とが示されている。ジョブ情報１６６に示されるように、このジョブ情報は、図３１に示されているジョブ４のジョブ情報である。
【０２０１】
ジョブ関連情報１６９には、項目に「根幹ジョブのジョブＩＤ」と、「親ジョブのジョブＩＤ」と、「子ジョブのジョブＩＤ」があり、それぞれ値が「１」、「２」、「７」となっている。実際に、ジョブ４の根幹ジョブはジョブ１であり、親ジョブはジョブ２であり、子ジョブは、ジョブ７である。
【０２０２】
このようなジョブ関連情報であれば、ジョブ管理モジュールは末端ジョブの情報を取得するだけで、ジョブ構造テーブルが作成できる。また、アプリ層、サービス層、ハンドラ層の各プログラムは、親のジョブ関連情報に親のジョブＩＤを追記するだけなので、ジョブ生成時にジョブ関連情報を作成するのが容易となる。さらに、子ジョブのジョブＩＤの項目に値がない場合は末端ジョブであるので、末端ジョブの検索が子ジョブのジョブＩＤの項目に値が入っているかどうかを見ることで可能となる。なお、値が入っていないとは、実質的な値が入っていないことであり、例えばＮＵＬＬもしくは0ｘｆｆｆｆなど、設計でよく用いられる値が入っていることであっても良い。なお、ジョブを生成するモジュールは、親ジョブを生成したモジュールに子ジョブのジョブＩＤを通知する必要がある。
【０２０３】
以上説明した第５のジョブ関連情報例の場合のジョブ登録時の動作を、図４９のシーケンス図を用いて説明する。なお、このシーケンス図は、図３４で示したシーケンス図と一部異なるので、その異なる部分を説明することにする。
【０２０４】
ステップＳ８０９は、ステップＳ１０９に対応するが、親ジョブ用のジョブ関連情報のみが通知される。また、ステップＳ８１０でジョブを生成した後、ステップＳ８１１で、プリンタアプリ２０に子ジョブの生成を通知する処理が追加される。
【０２０５】
またステップＳ８１８のプロット要求では、親ジョブのジョブ関連情報のみが通知される。さらに、ステップＳ８１９のジョブの生成後に、プロッタハンドラ２７から印刷制御モジュール２３へ子ジョブの生成が通知される。
【０２０６】
ジョブ参照時の動作シーケンスは、図３５と同じであるため、説明を省略する。
【０２０７】
次に、第２、３のジョブ関連情報を組み合わせた第６のジョブ関連情報例について、図５０を用いて説明する。図５０に示されるジョブ関連情報は、親ジョブのジョブＩＤと、全ての子孫ジョブのジョブＩＤからなる。
【０２０８】
具体的に、図５１に示されるジョブ情報を用いて説明する。図５１には、ジョブ情報１７０と、ジョブ関連情報１７１とが示されている。ジョブ情報１７０に示されるように、このジョブ情報は、図３１に示されているジョブ１のジョブ情報である。
【０２０９】
ジョブ関連情報１７１には、項目に「子ジョブ」と、「親ジョブ」がある。子ジョブで示されるジョブＩＤの親ジョブのジョブＩＤが親ジョブの欄に記載される。
【０２１０】
実際、ジョブ１は根幹ジョブであるので、親ジョブは存在せず、ジョブ２、３の親ジョ

ブはジョブ１であり、ジョブ４、５の親ジョブはジョブ２であり、ジョブ６の親ジョブはジョブ３であり、ジョブ７の親ジョブは、ジョブ４である。

【０２１１】

この第６のジョブ関連情報では、根幹ジョブの情報を取得するだけで、ジョブ構造テーブルが作成できる。なお、根幹ジョブのジョブ関連情報には、親ジョブのジョブＩＤの項目に値が入っていないことから、根幹ジョブを判別することができる。

【０２１２】

以上説明した第６のジョブ関連情報例の場合のジョブ登録時の動作を、図５２のシーケンス図を用いて説明する。なお、このシーケンス図は、図３４で示したシーケンス図と一部異なるので、その異なる部分を説明することにする。

【０２１３】

ステップＳ９０９は、ステップＳ１０９に対応するが、親ジョブのジョブＩＤのみ通知される。また、ステップＳ９１０でジョブを生成した後、ステップＳ９１１で、プリンタアプリ２０に子ジョブの生成を通知する処理が追加される。また、ステップＳ９１８は、ステップＳ１１７に対応するが、親ジョブのジョブＩＤのみ通知される。さらに、ステップＳ９２０で子ジョブの生成通知後、ステップＳ９２１で、印刷制御モジュール２３からプリンタアプリ２０に子孫ジョブの生成が通知される処理が加わる。

【０２１４】

ジョブ参照時の動作シーケンスは、図４３と同じであるため、説明を省略する。

【０２１５】

次に、ジョブ管理モジュール２６で一括管理する第７のジョブ関連情報例について、図５３を用いて説明する。

【０２１６】

第７のジョブ関連情報例は、各モジュールでの処理において子ジョブが派生した場合は、各モジュールがジョブ管理モジュールに派生した子ジョブのジョブＩＤと親のジョブＩＤの対応を通知することで生成される。ジョブ管理モジュール２６は、全ジョブの対応関係を保持しており、表示アプリにジョブ構造を表示させる場合はその情報を表示アプリに通知するようになっている。

【０２１７】

図５３には、ジョブの関係図１７２、１７３、１７４と、プリンタアプリ２０と、印刷制御モジュール２３と、ジョブ管理モジュール２６とが示されている。

【０２１８】

関係図１７２は、プリンタアプリ２０が生成したジョブの関係図である。関係図１７３は、印刷制御モジュール２３が生成したジョブの関係図である。関係図１７４は、関係図１７２、１７３に基づき、ジョブ管理モジュール２６が生成した関係図である。

【０２１９】

関係図１７２で示されているジョブの関係と、関係図１７３で示されているジョブの関係が、関係図１７４に反映されていることが分かる。この場合、ジョブ情報は、ジョブ関連情報を持たなくても良い。

【０２２０】

以上説明した第７のジョブ関連情報例の場合のジョブ登録時の動作を、図５４のシーケンス図を用いて説明する。なお、このシーケンス図は、図３４で示したシーケンス図と一部異なるので、その異なる部分を説明することにする。

【０２２１】

ステップＳ１００２のプリンタアプリ２０がジョブを生成する処理の後に、ステップＳ１００３と、ステップＳ１００４の処理が追加される。ステップＳ１００３の処理は、プリンタアプリ２０からジョブ管理モジュール２６へのジョブ生成の通知である。このとき、親ジョブＩＤと子ジョブＩＤが通知される。また、ステップＳ１００４は、ジョブ管理モジュール２６によるジョブ構造テーブルの更新である。

【０２２２】

ステップＳ１０１１は、ステップＳ１０９に対応するが、親ジョブのジョブＩＤのみの通知となる。

【０２２３】

ステップＳ１０１２の印刷制御モジュール２３がジョブを生成する処理の後に、ステップＳ１０１３と、ステップＳ１０１４の処理が追加される。ステップＳ１０１３の処理は、印刷制御モジュール２３からジョブ管理モジュール２６へのジョブ生成の通知である。このとき、親ジョブＩＤと子ジョブＩＤが通知される。また、ステップＳ１０１４は、ジョブ管理モジュール２６によるジョブ構造テーブルの更新である。

【０２２４】

ステップＳ１０２１は、ステップＳ１１７に対応するが、親ジョブのジョブＩＤのみの通知となる。

【０２２５】

ステップＳ１０２２のプロッタハンドラ２７がジョブを生成する処理の後に、ステップＳ１０２３と、ステップＳ１０２４の処理が追加される。ステップＳ１０２３の処理は、プロッタハンドラ２７からジョブ管理モジュール２６へのジョブ生成の通知である。このとき、親ジョブＩＤと子ジョブＩＤが通知される。また、ステップＳ１０２４は、ジョブ管理モジュール２６によるジョブ構造テーブルの更新である。

【０２２６】

次に、図５５を用いて、ジョブ参照時の動作シーケンスを説明する。図５５は、ユーザ１６０と、オペパネアプリ１６１と、ジョブ管理モジュール２６と、プリンタアプリ２０と、メモリ管理モジュール３０との間で行われる処理を示している。

【０２２７】

ステップＳ１１０１で、オペパネアプリ１６１は、ユーザ１６０からジョブ参照要求を通知される。ステップＳ１１０２で、オペパネアプリ１６１は、ジョブ管理モジュール２６にジョブ一覧要求を通知する。ステップＳ１１０３で、ジョブ管理モジュール２６は、オペパネアプリ１６１にジョブ一覧を通知する。オペパネアプリ１６１は、ステップＳ１１０４で、ユーザ１６０に、ジョブ一覧を通知（表示）する。ステップＳ１１０５で、オペパネアプリ１６１は、ユーザ１６０からジョブ情報を要求される。ステップＳ１１０６で、オペパネアプリ１６１は、ジョブ管理モジュール２６にジョブ情報の取得を要求する。

【０２２８】

ジョブ管理モジュール２６は、ステップＳ１１０７で、メモリ管理モジュール３０に、ジョブ情報の取得を要求する。メモリ管理モジュール３０は、ステップＳ１１０８で、ジョブ管理モジュール２６にジョブ情報を通知する。

【０２２９】

ジョブ管理モジュール２６は、ステップＳ１１０９で、プリンタアプリ２０に、キューの順番取得を要求する。プリンタアプリ２０は、ステップＳ１１１０で、ジョブ管理モジュール２６にキューの順番を通知する。ジョブ管理モジュール２６は、ステップＳ１１１１で、オペパネアプリ１６１にジョブ情報を通知する。ステップＳ１１１２で、オペパネアプリ１６１は、ユーザ１６０にジョブ情報を通知（表示）する。

【０２３０】

次に、ソフトウェアの構成が図３０で説明した構成であり、ジョブ関連情報が図３２に示したジョブ関連情報の場合の処理について説明する。図３０における構成は、ジョブ情報は各アプリ層、サービス層のモジュールが各自のメモリ空間に保持する構成である。この構成でのジョブ登録時の動作シーケンスを、図５６を用いて説明する。

【０２３１】

図５６には、ユーザ１６０と、プリンタアプリ２０と、印刷制御モジュール２３と、プロッタハンドラ２７との間で行われる処理を示している。

【０２３２】

ステップＳ１２０１で、プリンタアプリ２０は、ユーザ１６０から印刷要求を通知され

る。プリンタアプリ２０は、ステップＳ１２０２で、ジョブの生成を行い、ステップＳ１２０３で、ジョブＩＤをキューに追加する。ステップＳ１２０４で、プリンタアプリ２０は、キューからジョブＩＤを取得し、ステップＳ１２０５で、処理を実行する。

【０２３３】

ステップＳ１２０６で、プリンタアプリ２０は、印刷制御モジュール２３に印刷処理要求を通知する。このとき、親ジョブのジョブ関連情報と、親ジョブのジョブＩＤも通知される。印刷モジュール２３は、ステップＳ１２０７でジョブの生成を行い、ステップＳ１２０８でジョブＩＤをキューに追加する。そして、印刷制御モジュール２３は、ステップＳ１２０９でジョブＩＤをキューから取得し、ステップＳ１２１０で処理を実行する。

【０２３４】

ステップＳ１２１１で印刷制御モジュール２３は、プロッタハンドラ２７にプロット要求を通知する。このとき、親ジョブのジョブ関連情報と親ジョブＩＤも通知される。

【０２３５】

ステップＳ１２１２で、プロッタハンドラ２７は、ジョブの生成を行う。ステップＳ１２１３で、プロッタハンドラ２７は、ジョブＩＤをキューに追加し、ステップＳ１２１４でジョブＩＤをキューから取得する。

【０２３６】

プロッタハンドラ２７は、ステップＳ１２１５で、プロットを実行する。プロッタハンドラ２７は、印刷制御モジュール２３に、ステップＳ１２１６で、プロット終了を通知する。印刷制御モジュール２３は、ステップＳ１２１７で、プリンタアプリ２０に印刷終了を通知する。プリンタアプリ２０は、ステップＳ１２１８で、ユーザ１６０に印刷終了を通知（表示）する。

【０２３７】

次に、図５７を用いて、ジョブ参照時の動作シーケンスを説明する。図５７は、ユーザ１６０と、オペパネアプリ１６１と、ジョブ管理モジュール２６と、プリンタアプリ２０と、印刷制御モジュール２３と、プロッタハンドラ２７との間で行われる処理を示している。

【０２３８】

ステップＳ１３０１で、オペパネアプリ１６１は、ユーザ１６０からジョブ参照要求を通知される。ステップＳ１３０２で、オペパネアプリ１６１は、ジョブ管理モジュール２６にジョブ一覧要求を通知する。

【０２３９】

ジョブ管理モジュール２６は、ステップＳ１３０３で、プリンタアプリ２０にジョブ情報の取得を要求する。ステップＳ１３０４で、プリンタアプリ２０は、ジョブ管理モジュール２６にジョブ情報を通知する。

【０２４０】

ジョブ管理モジュール２６は、ステップＳ１３０５で、印刷制御モジュール２３にジョブ情報の取得を要求する。ステップＳ１３０６で、印刷制御モジュール２３は、ジョブ管理モジュール２６にジョブ情報を通知する。

【０２４１】

ジョブ管理モジュール２６は、ステップＳ１３０７で、プロッタハンドラ２７にジョブ情報の取得を要求する。ステップＳ１３０８で、プロッタハンドラ２７は、ジョブ管理モジュール２６にジョブ情報を通知する。

【０２４２】

ステップＳ１３０９で、ジョブ管理モジュール２６は、ジョブ一覧の作成を行う。ステップＳ１３１０で、ジョブ管理モジュール２６は、オペパネアプリ１６１にジョブ一覧を通知する。オペパネアプリ１６１は、ステップＳ１３１１で、ユーザ１６０に、ジョブ一覧を通知（表示）する。

【０２４３】

ステップＳ１３１２で、オペパネアプリ１６１は、ユーザ１６０からジョブ情報を要求

される。ステップＳ１３１３で、オペパネアプリ１６１は、ジョブ管理モジュール２６にジョブ情報の取得を要求する。ジョブ管理モジュール２６は、ステップＳ１３１４で、プリンタアプリ２０にジョブ情報の取得を要求する。プリンタアプリ２０は、ステップＳ１３１５で、ジョブ管理モジュール２６にジョブ情報を通知する。

【０２４４】

ステップＳ１３１６で、ジョブ管理モジュール２６は、プリンタアプリ２０に、キューの順番取得を要求する。プリンタアプリ２０は、ステップＳ１３１７で、ジョブ管理モジュール２６にキューの順番を通知する。ジョブ管理モジュール２６は、ステップＳ１３１８で、オペパネアプリ１６１にジョブ情報を通知する。ステップＳ１３１９で、オペパネアプリ１６１は、ユーザ１６０にジョブ情報を通知（表示）する。

【０２４５】

次に、図５８を用いて、ユーザからジョブを中止された場合の処理について説明する。ステップＳ１４０１で、オペパネアプリ１６１は、ユーザ１６０からジョブの中止を通知される。オペパネアプリ１６１は、ステップＳ１４０２で、ジョブ管理モジュール２６にジョブの中止を通知する。ジョブ管理モジュール２６は、ステップＳ１４０３で、プリンタアプリ２０に印刷要求の中止を通知する。

【０２４６】

プリンタアプリ２０は、ステップＳ１４０４でメモリ管理モジュール３０にジョブ情報の取得を要求し、ステップＳ１４０５で、中止処理を行う。プリンタアプリ２０は、ステップＳ１４０６で、印刷制御モジュール２３に印刷処理の中止を通知する。

【０２４７】

印刷制御モジュール２３は、ステップＳ１４０７で、メモリ管理モジュール３０にジョブ情報の取得を要求し、ステップＳ１４０８で、中止処理を行う。印刷制御モジュール２３は、ステップＳ１４０９で、プロッタハンドラ２７にプロットの中止を通知する。プロッタハンドラ２７は、ステップＳ１４１０で、メモリ管理モジュール３０にジョブ情報の取得を要求し、ステップＳ１４１１で、中止処理を行う。

【０２４８】

次に、図５９を用いて、ユーザからジョブの再実行を要求された場合の処理について説明する。ステップＳ１５０１で、オペパネアプリ１６１は、ユーザ１６０からジョブの再実行が通知される。オペパネアプリ１６１は、ステップＳ１５０２で、ジョブ管理モジュール２６にジョブの再実行を通知する。ジョブ管理モジュール２６は、ステップＳ１５０３で、プリンタアプリ２０に印刷再要求を通知する。プリンタアプリ２０は、ステップＳ１５０４で、メモリ管理モジュール３０にジョブ情報の取得を要求する。メモリ管理モジュール３０は、ステップＳ１５０５で、プリンタアプリ２０にジョブ情報を通知する。

【０２４９】

プリンタアプリ２０は、ステップＳ１５０６で、ジョブ情報を変更し、ステップＳ１５０７で、メモリ管理モジュール３０にジョブ情報を通知することで、ジョブ情報を保存する。ステップＳ１５０８は、プリンタアプリ２０の処理の実行である。

【０２５０】

プリンタアプリ２０は、ステップＳ１５０９で、印刷制御モジュール２３に印刷処理の再実行を通知する。印刷制御モジュール２３は、ステップＳ１５１０で、メモリ管理モジュール３０にジョブ情報の取得を要求する。メモリ管理モジュール３０は、ステップＳ１５１１で、印刷制御モジュール２３にジョブ情報を通知する。印刷制御モジュール２３は、ステップＳ１５１２で、ジョブ情報を変更し、ステップＳ１５１３で、メモリ管理モジュール３０にジョブ情報を通知することで、ジョブ情報を保存する。ステップＳ１５１４は、印刷制御モジュール２３の処理の実行である。

【０２５１】

ステップＳ１５１５で、印刷制御モジュール２３は、プロッタハンドラ２７にプロット再実行を通知する。プロッタハンドラ２７は、ステップＳ１５１６で、メモリ管理モジュール３０にジョブ情報の取得を要求する。メモリ管理モジュール３０は、ステップＳ１５

１７で、印刷制御モジュール２３にジョブ情報を通知する。印刷制御モジュール２３は、ステップＳ１５１８で、ジョブ情報を変更し、ステップＳ１５１９で、メモリ管理モジュール３０にジョブ情報を通知することで、ジョブ情報を保存する。ステップＳ１５２０は、プロッタの実行である。

【０２５２】

プロッタハンドラ２７は、印刷制御モジュール２３に、ステップＳ１５２１で、プロット終了を通知する。印刷制御モジュール２３は、ステップＳ１５２２で、プリンタアプリ２０に印刷終了を通知する。プリンタアプリ２０は、ステップＳ１５２３で、オペパネアプリ１６１に印刷終了を通知する。

【０２５３】

以上がＭＦＰ内部の処理に関する説明である。次に、ＭＦＰの連携とユーザインタフェース（以下、ＵＩと記す）について説明する。ＭＦＰの連携とは、例えば、通信可能な2台のＭＦＰにおいて、一方のＭＦＰがカラーページ印刷し、他方のＭＦＰが白黒ページを印刷するように、複数のＭＦＰが１つの根幹ジョブから派生したジョブを分担して実行することをいう。

【０２５４】

まず、ＵＩから説明することにする。図６０は、オペレーションパネルまたはパソコンの画面に表示されるＵＩ２００である。このＵＩ２００は、図に示されるように、ジョブ情報とジョブのジョブ構造を一度に表示する場合のＵＩである。この図６０に示されるように、このＵＩには、ジョブ同士の関係が木構造を用いて明示される。

【０２５５】

ＵＩ２００には、２つのジョブ構造２０１、２０２が表示されている。ジョブ構造２０１には、ジョブＩＤが１から６までのジョブが表示され、ジョブ構造２０２には、ジョブＩＤが７から９までのジョブが表示されている。これらのジョブのうち、ジョブ１、７が根幹ジョブで、ジョブ４、５、６、８、９が末端ジョブである。

【０２５６】

またジョブ情報には、図２０または図２１で説明したジョブ情報が表示されている。このジョブ情報には、図に示されるように、種別や状態、オーナー名などが表示されるので、各ジョブ情報が一目で分かるようになっている。

【０２５７】

次に、図６１を用いて、他のＵＩの例を説明する。図６１に示されるＵＩ２２０は、ＭＦＰ２１０のジョブのジョブ構造が示されているものである。このＵＩ２２０は、カーソルで、ジョブを選択すると、選択されたジョブの情報が表示されるものである。

【０２５８】

図６１には、印刷ジョブ２１２がカーソル２１１で選択され、選択された印刷ジョブ２１２のジョブ情報２１３が、下部に表示される様子が示されている。また、このＵＩ２２０は、さらにジョブ情報の編集、ジョブの再開、中止が可能となっている。それらは、編集ボタン２１４、再開ボタン２１５、中止ボタン２１６で行うことができる。なお、編集とは、ジョブ情報の編集であり、再開とは、停止していたジョブの実行を再び開始することであり、中止とは、ジョブの実行を中止することである。

【０２５９】

このように、ＵＩ２００、２２０は、ジョブのジョブ構造ならびにジョブ情報が分かりやすく表示されるものである。

【０２６０】

次に、上述したＵＩ２００、２２０で、実際にジョブの実行を中止する場合のＵＩについて説明する。まず、ＵＩ２００の場合について、図６２、６３、６４を用いて説明する。図６２には、エラーが発生しているジョブ３が示されている。ジョブ３の処理の実行を中断するユーザは、このジョブ３のジョブ情報２３０にカーソル２１１をあわせ、例えば右クリックする。

【０２６１】

右クリックにより、図６３に示されるＵＩとなる。このＵＩには、図に示されるように、ジョブに対して実行可能な操作項目であるメニュー２３１が表示されている。メニュー２３１には、「ジョブの中止」と、「ジョブの再開」と、「ジョブ情報編集」と、「プロパティ」が表示される。このとき、メニュー２３１のジョブの中止にカーソル２２１をあわせ、例えば左クリックする。

【０２６２】

これにより、ジョブ３の処理の実行が中止され、図６４に示されるように、ジョブ３が表示されなくなる。

【０２６３】

次に、ＵＩ２２０の場合について、図６５、６６、６７、６８を用いて説明する。図６５には、エラーが発生しているプロッタジョブ２４０が示されている。このプロッタジョブ２４０の処理の実行を中断するユーザは、このプロッタジョブ２４０にカーソル２１１をあわせ、例えば左クリックする。

【０２６４】

この左クリックによって、図６６に示されるように、ジョブ情報２４１と、中止ボタン２４２などが表示される。そこで、図６７に示されるように、プロッタジョブ２４０の処理の実行を中断するユーザは、中止ボタン２４２にカーソル２１１をあわせ、例えば左クリックする。

【０２６５】

これにより、プロッタジョブ２４０の処理の実行が中止され、図６８に示されるように、プロッタジョブ２４０が表示されなくなる。

【０２６６】

次に、ＭＦＰの連携について説明する。図６９は、ＭＦＰの連携を示す図である。図６９には、ユーザ１６０と、ＰＣ２５０と、ネットワーク２５３と、ＭＦＰＡ２５１と、ＭＦＰＢ２５２とが示されている。ＰＣ２５０と、ＭＦＰＡ２５１と、ＭＦＰＢ２５２は、ネットワーク２５３で接続され、通信可能である。

【０２６７】

この構成で、例えばユーザ１６０がＰＣ２５０からＭＦＰＡ２５１に、カラーと白黒が混在したページを有する文書の印刷を要求したとする。この場合、ＭＦＰＡ２５１は、自らが白黒ページを印刷するとともに、ＭＦＰＢ２５２にカラーページの印刷要求をし、ＭＦＰＢ２５２がカラーページを印刷する。

【０２６８】

この処理におけるジョブ構造は、図１０で説明したジョブ構造である。このようにジョブのジョブ構造が複数のＭＦＰにまたがって発生した場合、図７０に示されるような、ジョブを管理するジョブ管理サーバを用意すると、ネットワーク上にある全体のジョブのジョブ構造を管理することが可能になる。

【０２６９】

図７０は、２台のＭＦＰと、ジョブ管理サーバ２８０が設けられた構成が示されている。ジョブ構造テーブル２８２は、ＭＦＰＡ２５１で実行されているジョブのジョブ構造テーブルであり、ジョブ構造テーブル２８３は、ＭＦＰ２５２で実行されているジョブのジョブ構造テーブルである。

【０２７０】

これら２つのジョブ構造テーブルを各ＭＦＰがジョブ管理サーバ２８０に通知することにより、ジョブ管理サーバ２８０は、取得したジョブのジョブ構造テーブルを結合し、ネットワーク上にある全体のジョブのジョブ構造テーブル２８４を作成することができる。このときのジョブ構造テーブルを取得する処理が、ジョブ構造情報取得段階に対応する。なお、ジョブ構造情報取得手段は、上述したようにＭＦＰであればオペパネアプリであってもよいし、パソコンであれば、ＨＴＭＬブラウザであってもよい。また、ジョブ構造テーブルとともに、ジョブ情報を取得しても良い。

【０２７１】

以上の処理は、ジョブ構造テーブルを受信し、ジョブ構造テーブルに基づいたグラフィカルユーザインタフェースを表示する処理である。

【０２７２】

このようにジョブを管理することにより、ジョブ管理サーバ２８０に表示される管理画面には、図７１に示されるＵＩ２９０を表示することができる。このＵＩ２９０には、ＭＦＰＡ２５１とＭＦＰＢ２５２のジョブと、上述したジョブ情報２１３と、編集ボタン２１４、再開ボタン２１５、中止ボタン２１６とが表示される。ジョブ情報２１３は、カーソル２１１があっている印刷ジョブ２１２のジョブ情報である。

【０２７３】

このＵＩ２９０は、ＭＦＰＢ２５２のジョブであることを明示するために、根幹ジョブである印刷ジョブ２１２から派生し、ＭＦＰＢ２５２のジョブは、例えば点線２８６で囲まれるようになっている。

【０２７４】

このようなＵＩ上で、ＭＦＰに属するジョブを、他のＭＦＰで実行させる場合のＵＩについて説明する。図７２に示されるＵＩには、ＭＦＰＡ２５１で実行されている印刷ジョブから派生したジョブが表示されている。これらのジョブのうち、カラープリントジョブ２９４を、ＭＦＰＢ２５２で実行することとする。

【０２７５】

この場合、図７３に示されるように、カラープリントジョブ２９４をＭＦＰＢ２５２のアイコンにドラッグする。この場合、アイコン２９５に示されるように、カラープリントジョブ２９４をドラッグすれば、カラープリントジョブ２９４の子ジョブも一緒にドラッグされる。

【０２７６】

そして、図７４に示されるように、ドラッグされたカラープリントジョブ２９５と、その子ジョブが、ＭＦＰＢ２５２のジョブとして表示される。

【０２７７】

このように、ユーザの指示により、ジョブが属する機器を変更することができる。

【０２７８】

以上説明したようなネットワークを介したＭＦＰを利用すると、あるＭＦＰでエラーが発生したジョブを他のＭＦＰで処理し、エラーを回避するということなどが可能となる。

【０２７９】

次に、ジョブ構造のいろいろな表示方法について説明する。まず、図７５を用いて、根幹ジョブのみを表示するＵＩについて説明する。図７５に示されるＵＩは、図６０で説明したジョブのうち、根幹ジョブのみを表示したものである。図６０にしめされるジョブ構造の根幹ジョブは、ジョブ１と、ジョブ７の２つのジョブである。従って、図７５に示されるＵＩには、ジョブ１とジョブ７のジョブ情報が表示される。

【０２８０】

次に、末端ジョブのみを表示するＵＩについて説明する。図７６に示されるＵＩは、図６０で説明したジョブのうち、末端ジョブのみを表示したものである。図６０にしめされるジョブ構造の末端ジョブは、ジョブ４、ジョブ５、ジョブ６、ジョブ８、ジョブ９の５つのジョブである。従って、図７６に示されるＵＩには、上記５つのジョブ情報が表示される。

【０２８１】

次に、ジョブの一覧を表示するＵＩについて説明する。図６０に示されるＵＩは、ジョブをツリー構造で表示するものであった。図７７に示されるＵＩは、ツリー構造ではなく、一覧として表示するＵＩを示すものである。

【０２８２】

図７７のＵＩには、図６０に示される全てのジョブが一覧として表示されている。一覧として表示されるため、このＵＩは、各項目ごとにソートをかけることが可能なＵＩである。例えば、オーナー名でソートをかけると、そのオーナーが要求したジョブについて一

目瞭然となる。
【０２８３】
また、図７７のＵＩでは、検索を行うことができる。そのことを、図７８を用いて説明する。図７８には、検索条件画面７００と、検索対象ラジオボタン７０１と、検索文字列入力欄７０２と、検索ボタン７０３と、キャンセルボタン７０４とが表示されている。
【０２８４】
検索対象ラジオボタン７０１は、検索対象の項目を選択するためのラジオボタンである。この図では、項目として「状態」が選択されている。検索文字列７０２は、検索する文字列が入力されるテキストボックスである。この図では、文字列として「待機中」が入力されている。検索ボタン７０３は、選択された検索対象と、入力された文字列に基づき検索を実行させるボタンである。キャンセルボタン７０４は、何もせず、この検索条件画面７００を閉じるためのボタンである。
【０２８５】
この検索条件画面７００により、対象となるジョブを検索することができる。このような検索が可能であるのは、ＭＦＰ内で、ジョブ構造テーブルが作成されているからである。
【０２８６】
検索ボタン７０３をユーザがクリックすると、図７９に示される検索結果を示すＵＩが表示される。図７７において、状態が待機中のジョブは、ジョブ５とジョブ９であり、それらのジョブが、図７９のＵＩに表示されている。
【図面の簡単な説明】
【０２８７】
【図１】ジョブの派生を示す図である。
【図２】ページが混在した文書を印刷する際のジョブの派生を示す図である。
【図３】プリンタアプリでのジョブ管理を示す図である。
【図４】サービス層のモジュールでのジョブ管理を示す図である。
【図５】ユーザインタフェースを示す図である。
【図６】ジョブの状態遷移図である。
【図７】根幹ジョブから派生した子ジョブを示す図である。
【図８】１つのジョブから１つのジョブが派生する場合の例を示す図である。
【図９】１つのジョブから複数のジョブが派生する場合の例を示す図である。
【図１０】複数ＭＦＰでのジョブ構造を示す図である。
【図１１】処理の委譲によるジョブの派生の例を示す図である。
【図１２】処理の分割によるジョブの派生の例を示す図である（その１）。
【図１３】処理の分割によるジョブの派生の例を示す図である（その２）。
【図１４】ジョブの派生を示す図である。
【図１５】シングルプロセス・シングルスレッドでの実装例を示す図である。
【図１６】シングルプロセス・マルチスレッドでの実装例を示す図である。
【図１７】マルチプロセス・シングルスレッドでの実装例を示す図である。
【図１８】マルチプロセス・マルチスレッドでの実装例を示す図である。
【図１９】ユーザによるジョブへの操作の実装例を示す図である。
【図２０】ジョブ情報を示す図である（その１）。
【図２１】ジョブ情報を示す図である（その２）。
【図２２】ジョブの操作の許可情報の例を示す図である。
【図２３】ジョブを操作できる条件の例を示す図である。
【図２４】ジョブ構造テーブルを示す図である。
【図２５】ジョブ構造テーブル（ＸＭＬ）を示す図である。
【図２６】コピー蓄積処理で発生するジョブを示す図である。
【図２７】ＭＦＰのソフトウェアブロック図である。
【図２８】ＭＦＰのハードウェア構成図である。

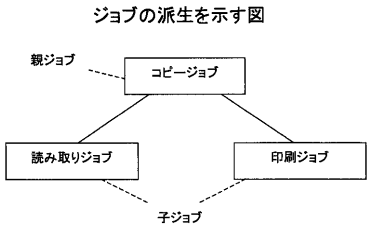
【図２９】ソフトウェアの関係を示す図（その１）である。
【図３０】ソフトウェアの関係を示す図（その２）である。
【図３１】ジョブ構造を示す図である。
【図３２】第１のジョブ関連情報例を示す図である。
【図３３】第１のジョブ関連情報例におけるジョブ情報を示す図である。
【図３４】第１のジョブ関連情報例におけるジョブ登録時の動作シーケンス図である。
【図３５】第１のジョブ関連情報例におけるジョブ参照時の動作シーケンス図である。
【図３６】第２のジョブ関連情報例を示す図である。
【図３７】第２のジョブ関連情報例におけるジョブ情報を示す図である。
【図３８】第２のジョブ関連情報例におけるジョブ登録時の動作シーケンス図である。
【図３９】第２のジョブ関連情報例におけるジョブ参照時の動作シーケンス図である。
【図４０】第３のジョブ関連情報例を示す図である。
【図４１】第３のジョブ関連情報例におけるジョブ情報を示す図である。
【図４２】第３のジョブ関連情報例におけるジョブ登録時の動作シーケンス図である。
【図４３】第３のジョブ関連情報例におけるジョブ参照時の動作シーケンス図である。
【図４４】第４のジョブ関連情報例を示す図である。
【図４５】第４のジョブ関連情報例におけるジョブ情報を示す図である。
【図４６】第４のジョブ関連情報例におけるジョブ登録時の動作シーケンス図である。
【図４７】第５のジョブ関連情報例を示す図である。
【図４８】第５のジョブ関連情報例におけるジョブ情報を示す図である。
【図４９】第５のジョブ関連情報例におけるジョブ登録時の動作シーケンス図である。
【図５０】第６のジョブ関連情報例を示す図である。
【図５１】第６のジョブ関連情報例におけるジョブ情報を示す図である。
【図５２】第６のジョブ関連情報例におけるジョブ登録時の動作シーケンス図である。
【図５３】第７のジョブ関連情報例を示す図である。
【図５４】第７のジョブ関連情報例におけるジョブ登録時の動作シーケンス図である。
【図５５】第７のジョブ関連情報例におけるジョブ参照時の動作シーケンス図である。
【図５６】図３０で示した構成におけるジョブ登録時の動作シーケンス図である。
【図５７】図３０で示した構成におけるジョブ参照時の動作シーケンス図である。
【図５８】ジョブを中止する場合の処理を示すシーケンス図である。
【図５９】ジョブを再実行する場合の処理を示すシーケンス図である。
【図６０】ユーザインタフェースを示す図である（その１）。
【図６１】ユーザインタフェースを示す図である（その２）。
【図６２】エラーの発生を示す図である。
【図６３】メニューの表示を示す図である。
【図６４】ジョブの中止を示す図である。
【図６５】エラーの発生を示す図である。
【図６６】ジョブ情報の表示を示す図である。
【図６７】ジョブの中止を示す図である。
【図６８】中止されたジョブの消滅を示す図である。
【図６９】ＭＦＰの連携を示す図である。
【図７０】２台のＭＦＰとジョブ管理サーバからなる構成を示す図である。
【図７１】ジョブ管理サーバに表示される管理画面を示す図である。
【図７２】ＭＦＰ　Ａのジョブ構造を示す図である。
【図７３】ジョブのアイコンをドラッグする様子を示す図である。
【図７４】ドラッグされたジョブが他のＭＦＰのジョブとなった様子を示す図である。
【図７５】根幹ジョブのみの表示を示す図である。
【図７６】末端ジョブのみの表示を示す図である。
【図７７】全てのジョブの一覧表示を示す図である。
【図７８】検索条件画面を示す図である。

【図７９】検索結果を示す図である。
【符号の説明】
【０２８８】
５　アプリケーション層
７　サービス層
９　ハンドラ層
２０　プリンタアプリ
２１　コピーアプリ
２２　ＦＡＸアプリ
２３　印刷制御モジュール
２４　ＦＡＸ制御モジュール
２５　読み取り制御モジュール
２６　ジョブ管理モジュール
２７　プロッタハンドラ
２８　ＦＡＸユニットハンドラ
２９　スキャナハンドラ
３０　メモリ管理モジュール
５１　スキャナ
５２　プロッタ
５３　オペレーションパネル
６０　コントローラボード
６１　ＣＰＵ
６２　ノースブリッジ（ＮＢ）
６３　システムメモリ（ＭＥＭ－Ｐ）
６４　ローカルメモリ（ＭＥＭ－Ｃ）
６５　ハードディスク装置（ＨＤＤ）
６６　ＡＳＩＣ
６７　ＡＧＰ（Accelerated Graphics Port）
６８　ファックスコントロールユニット（ＦＣＵ）
６９　Ｇ３
７０　Ｇ４
７１　エンジン
７３　サウスブリッジ（ＳＢ）
７４　ＮＩＣ
７５　ＵＳＢデバイス
７６　ＩＥＥＥ１３９４デバイス
７７　セントロニクス
１０３　ＦＡＸ送信ジョブ
１０４　宛先ごとのＦＡＸ送信ジョブ
１１０　コピー蓄積ジョブ
１１１　読み取りジョブ
１１２　プリントジョブ
１１３　蓄積ジョブ
１１４、１１５　プロッタジョブ
１３０　各プログラム
１３１　専用キュー
１３２、１５０、１６３、１６６、１６７、１７０、２１３、２３０、２４１　ジョブ情報
１３３　キュー
１５１、１６４、１６８、１６９、１７１　ジョブ関連情報

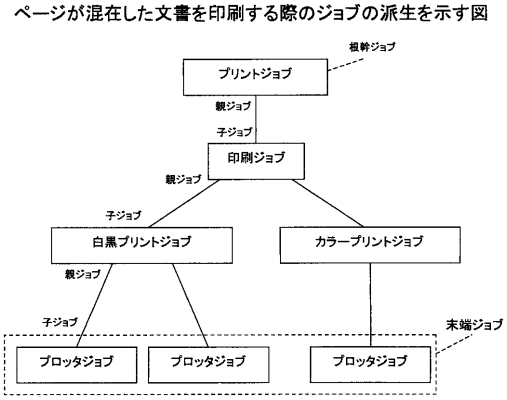
１７２、１７３、１７４　関係図
２００、２２０、２９０　ＵＩ
２０１、２０２、２７１、２７２　ジョブ構造
２１０　ＭＦＰ
２１１　カーソル
２１２　印刷ジョブ
２１４　編集ボタン
２１５　再開ボタン
２１６、２４２　中止ボタン
２３１　メニュー
２４０　プロッタジョブ
２５０　ＰＣ
２５１　ＭＦＰ　Ａ
２５２　ＭＦＰ　Ｂ
２５３　ネットワーク
２５４　白黒プリントジョブ
２５５、２５６、２５８　プロッタジョブ
２５７　カラープリントジョブ
２５９　印刷ジョブ
ジョブ構造
２８０　ジョブ管理サーバ
２８２、２８３、２８４　ジョブ構造テーブル
２８６　点線
２９４　カラープリントジョブ
２９５　アイコン
４００　ジョブＡ
４０１　ジョブＡ-Ａ
４０２　ジョブＡ-Ｂ
４０３　ジョブＡ-Ａ-Ａ
４０４　ジョブＡ-Ａ-Ｂ
４０５　ジョブＡ-Ｂ-Ａ
４１０、４１５　プリントジョブ
４１１、４１６、４３２　印刷ジョブ
４１２、４１９、４２０、４２１　プロッタジョブ
４１７　白黒プリントジョブ
４１８　カラープリントジョブ
４３０　コピージョブ
４３１　読み取りジョブ
４３３　１ページ～１０ページの印刷ジョブ
４３４　１１ページ目の印刷ジョブ
４３５　１２ページ～２０ページの印刷ジョブ
４４１　ＦＡＸ送信ジョブ
４４２、４４３、４４４　宛先ごとのＦＡＸ送信ジョブ
４５０　ジョブＡ
４５１　ジョブＢ
４５２　ジョブＣ
４５３　ジョブＤ
５００　プロセスＡ
５０１　スレッドＡ
５０２　スレッドＢ

５０３　スレッドＣ
５０５　プロセスＢ
５０６　プロセスＣ
５０７　スレッドＤ
５０８　スレッドＥ
６０１　待機中
６０２　実行中
６０３　完了
６０４　中断中
６０５　中止
６０６　エラー発生
７００　検索条件画面
７０１　検索対象ラジオボタン
７０２　検索文字列入力欄
７０３　検索ボタン
７０４　キャンセルボタン

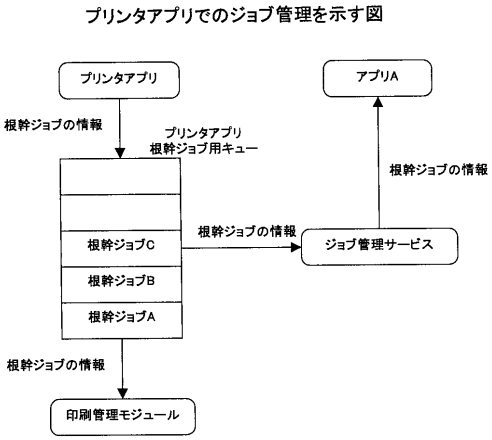
【図１】



【図２】

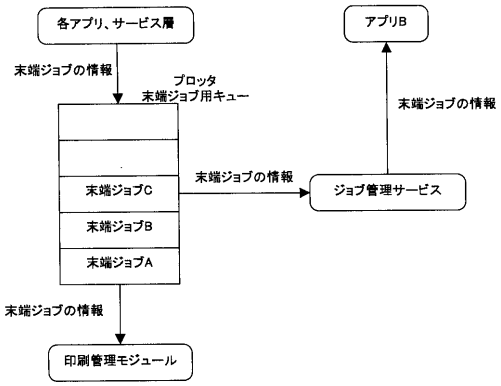


【図３】



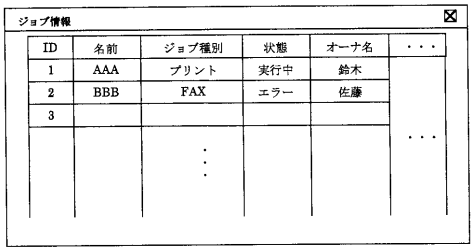
【図４】

サービス層のモジュールでのジョブ管理を示す図



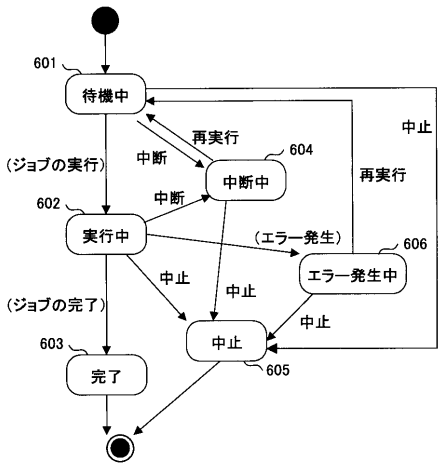
【図５】

ユーザインタフェースを示す図



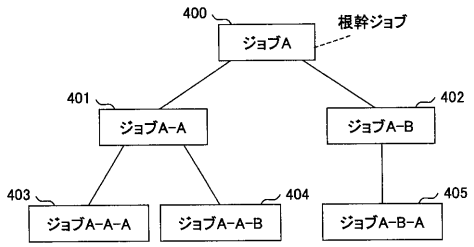
【図６】

ジョブの状態遷移図



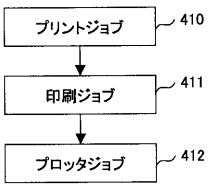
【図７】

根幹ジョブから派生した子ジョブを示す図



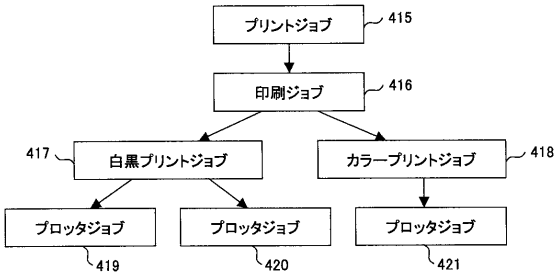
【図８】

1つのジョブから1つのジョブが派生する場合の例を示す図



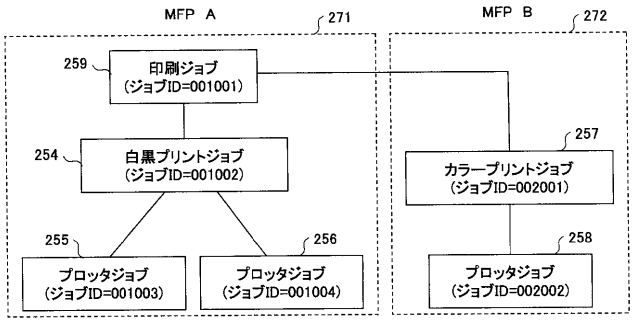
【図９】

1つのジョブから複数のジョブが派生する場合の例を示す図



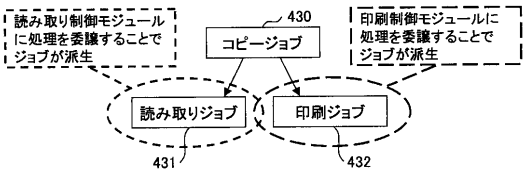
【図１０】

複数MFPでのジョブ構造を示す図



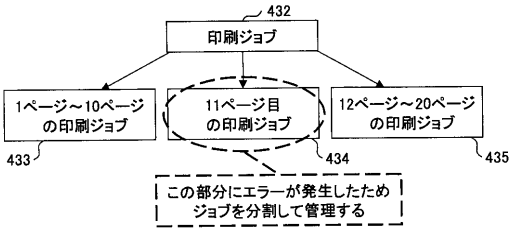
【図１１】

処理の委譲によるジョブの派生の例を示す図



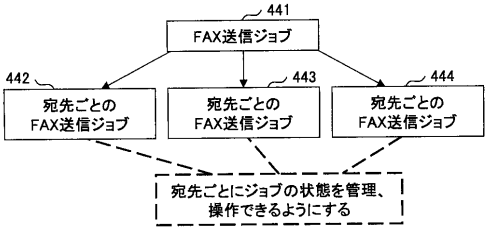
【図１２】

処理の分割によるジョブの派生の例を示す図である（その１）



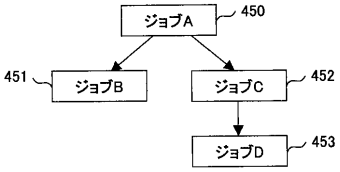
【図１３】

処理の分割によるジョブの派生の例を示す図である（その２）



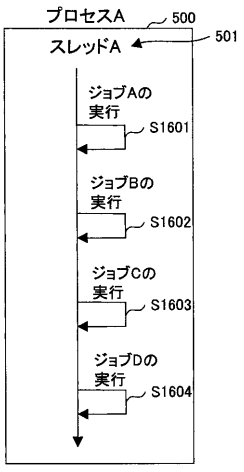
【図１４】

ジョブの派生を示す図



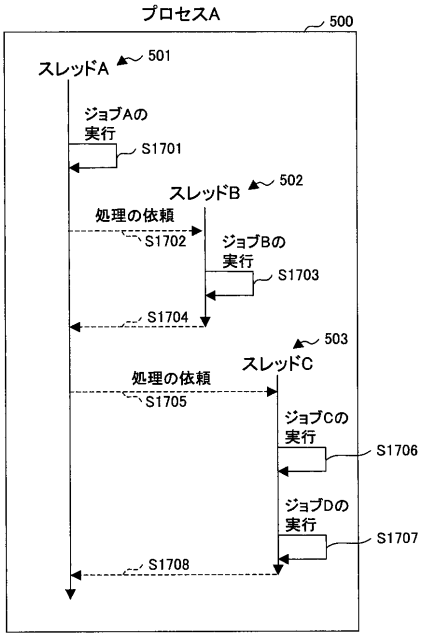
【図１５】

シングルプロセス・シングルスレッドでの実装例を示す図



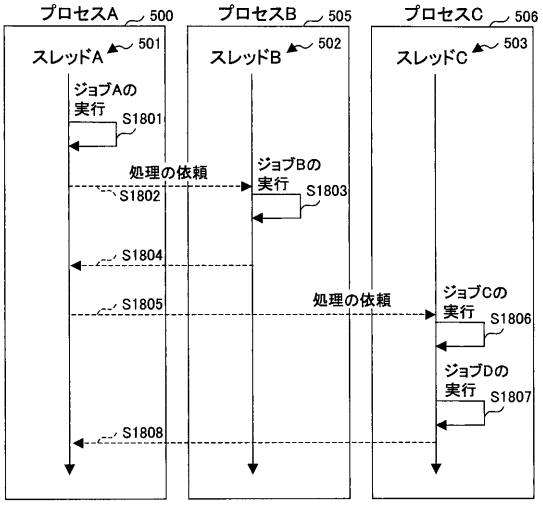
【図１６】

シングルプロセス・マルチスレッドでの実装例を示す図



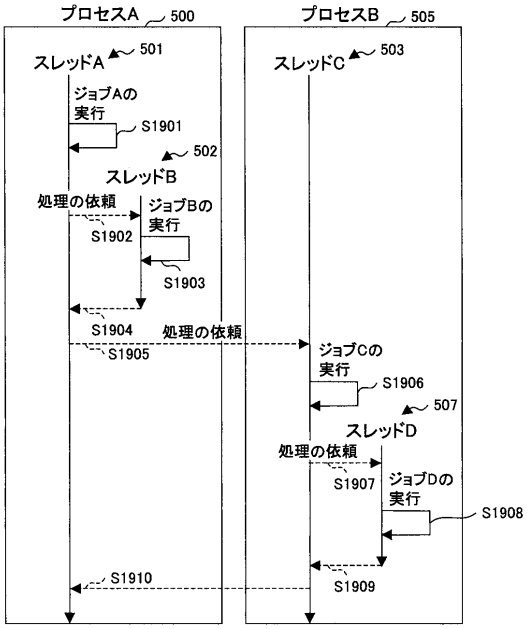
【図１７】

マルチプロセス・シングルスレッドでの実装例を示す図



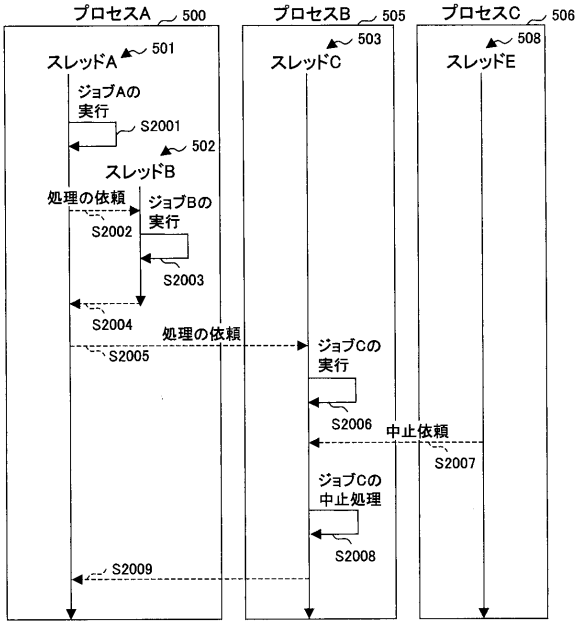
【図１８】

マルチプロセス・マルチスレッドでの実装例を示す図



【図１９】

ユーザによるジョブへの操作の実装例を示す図



【図２０】

ジョブ情報を示す図である(その1)

ジョブID
文書名
ジョブ種別
状態
オーナー名
作成日
ジョブ関連情報
・ ・ ・

【図２１】

ジョブ情報を示す図である(その2)

ジョブID
ジョブ種別
状態
ジョブ関連情報
ジョブの操作の許可情報
ジョブを操作できる条件
・ ・ ・

【図２２】

ジョブの操作の許可情報の例を示す図

項目	値
ジョブID	2
ジョブ種別	印刷ジョブ
状態	待機中
ジョブ関連情報(親ジョブID)	1
ジョブの操作の許可情報	不許可
ジョブを操作できる条件	−

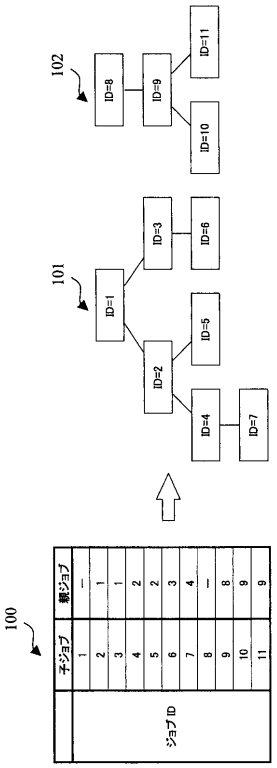
【図２３】

ジョブを操作できる条件の例を示す図

項目	値
ジョブID	2
ジョブ種別	印刷ジョブ
状態	実行中
ジョブ関連情報(親ジョブID)	1
ジョブの操作の許可情報	条件発生時のみ許可
ジョブを操作できる条件	実行中またはエラー発生中

【図２４】

ジョブ構造テーブルを示す図



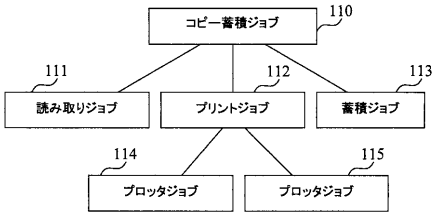
【図２５】

ジョブ構造テーブル(XML)を示す図

```
<?xml version="1.0"?>
<job id="8">
    <job id="9">
        <job id="10"/>
        <job id="11"/>
    </job>
</job>
```

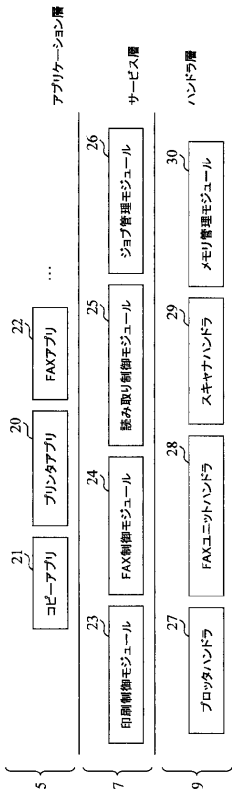
【図２６】

コピー蓄積処理で発生するジョブを示す図



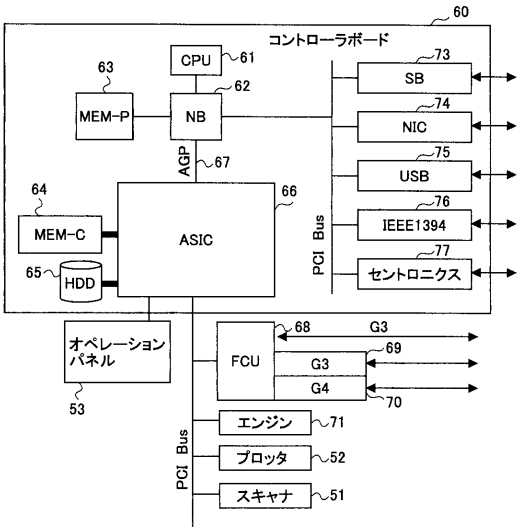
【図２７】

MFPのソフトウェアブロック図



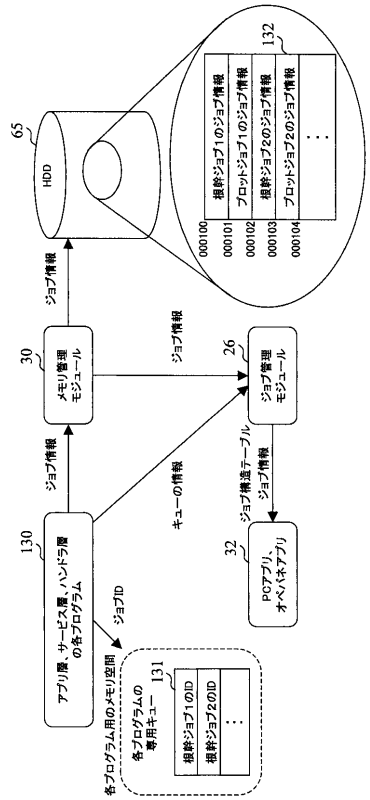
【図２８】

ＭＦＰのハードウェア構成図



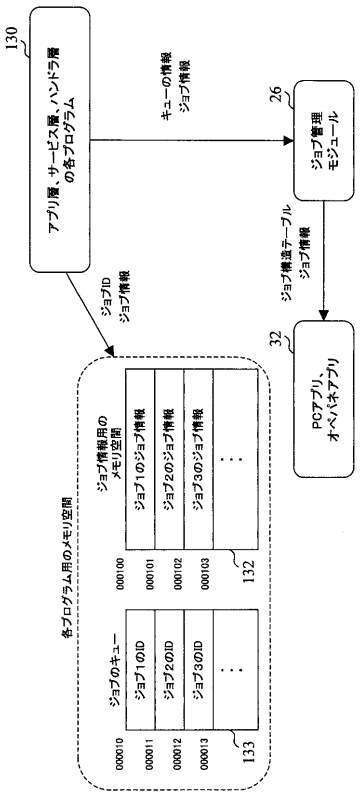
【図２９】

ソフトウェアの関係を示す図（その1）



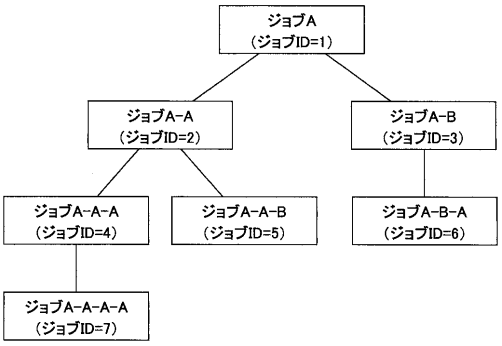
【図３０】

ソフトウェアの関係を示す図（その2）



【図３１】

ジョブ構造を示す図



【図３２】

第1のジョブ関連情報例を示す図

根幹ジョブのジョブID
2世代目のジョブID
・ ・ ・
親ジョブのジョブID

【図３３】

第1のジョブ関連情報例におけるジョブ情報を示す図

150

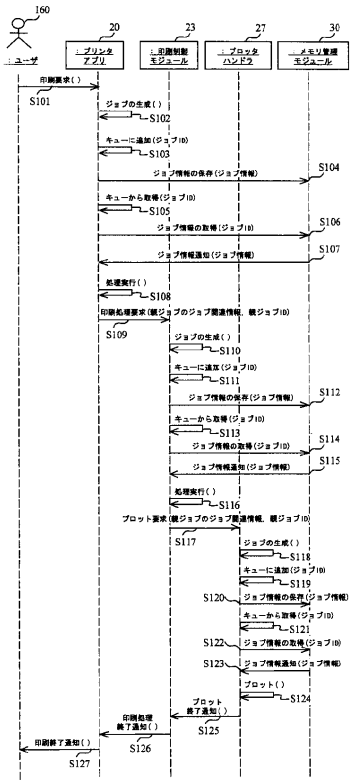
項目	値
ジョブID	7
ジョブ種別	プロッタ
状態	実行中
ジョブ関連情報	
・	・
・	・
・	・

151

項目	値
根幹ジョブのジョブID	1
2世代目のジョブID	2
親ジョブのジョブID	4

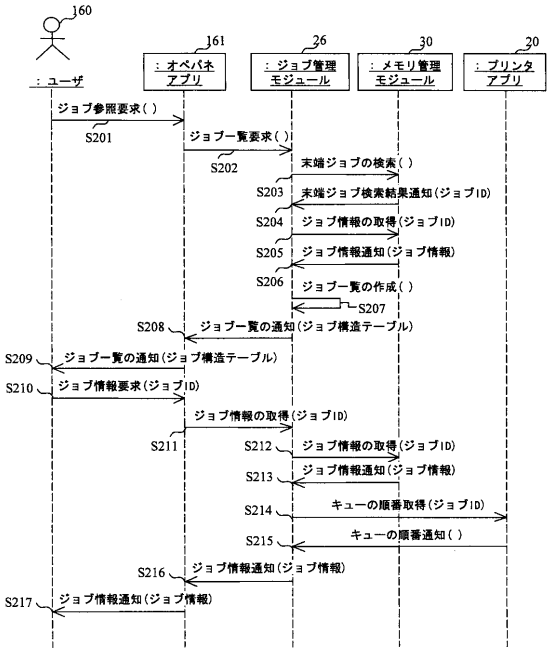
【図３４】

第1のジョブ関連情報例におけるジョブ登録時の動作シーケンス図



【図３５】

第1のジョブ関連情報例におけるジョブ参照時の動作シーケンス図



【図３６】

第2のジョブ関連情報例を示す図

ジョブID
ジョブ種別
状態
親ジョブのジョブID
・
・
・

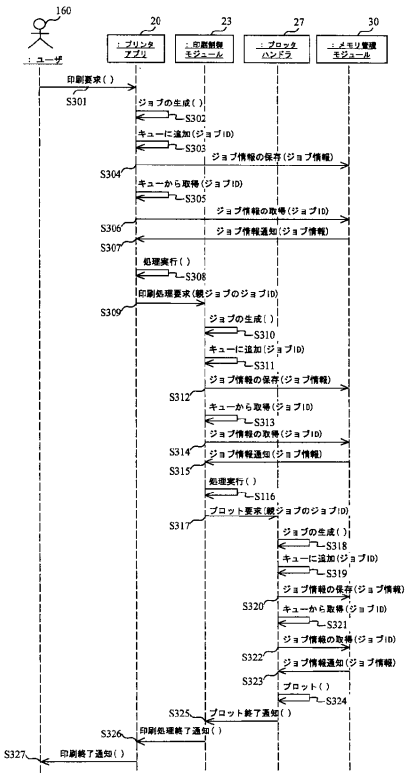
【図３７】

第2のジョブ関連情報例におけるジョブ情報を示す図

項目	値
ジョブID	7
ジョブ種別	プロッタ
状態	実行中
親ジョブのジョブID	4
・	・
・	・
・	・

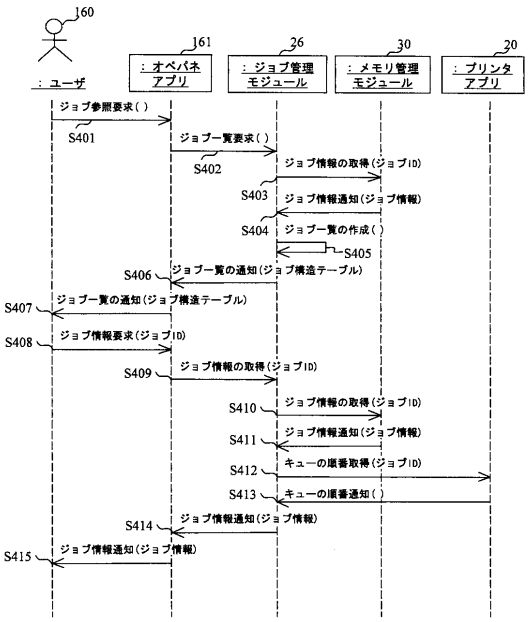
【図３８】

第2のジョブ関連情報例におけるジョブ登録時の動作シーケンス図



【図３９】

第2のジョブ関連情報例におけるジョブ参照時の動作シーケンス図



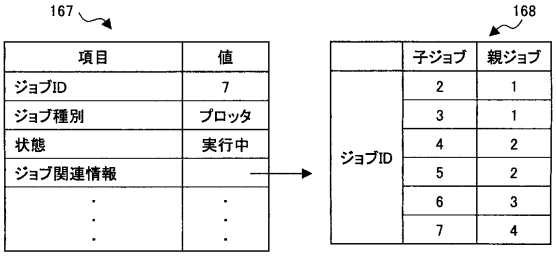
【図４０】

第3のジョブ関連情報例を示す図

子ジョブのIDとその親のID
孫ジョブのIDとその親のID
ひ孫ジョブのIDとその親のID
・ ・ ・

【図４１】

第3のジョブ関連情報例におけるジョブ情報を示す図

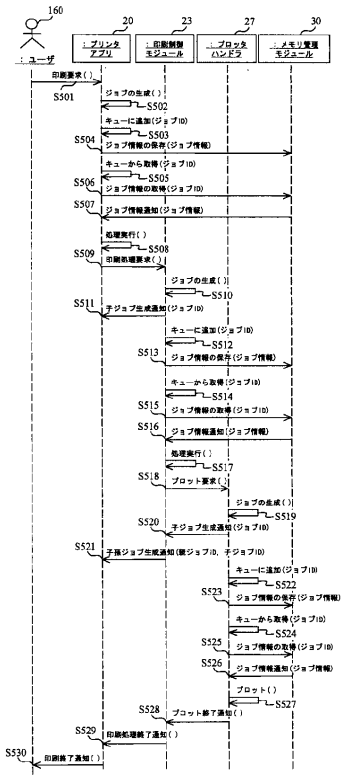


項目	値
ジョブID	7
ジョブ種別	プロッタ
状態	実行中
ジョブ関連情報	
・ ・ ・	・ ・ ・

	子ジョブ	親ジョブ
ジョブID	2	1
	3	1
	4	2
	5	2
	6	3
	7	4

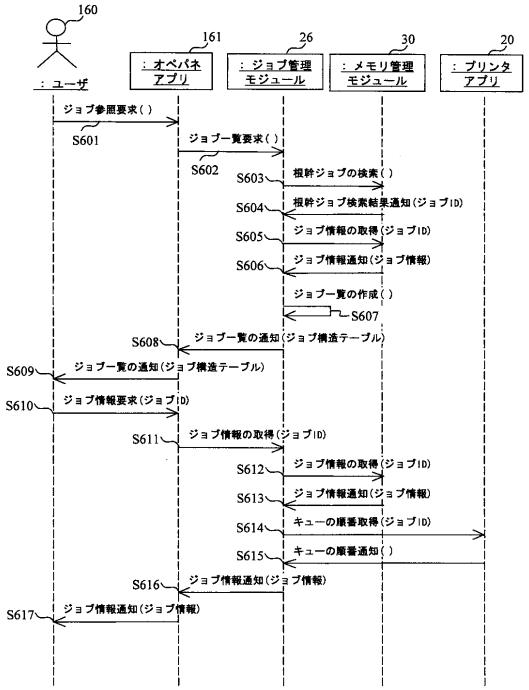
【図４２】

第3のジョブ関連情報例におけるジョブ登録時の動作シーケンス図



【図４３】

第3のジョブ関連情報例におけるジョブ参照時の動作シーケンス図



【図４４】

第4のジョブ関連情報例を示す図

子ジョブAのジョブID
子ジョブBのジョブID
・ ・ ・

【図４５】

第4のジョブ関連情報例におけるジョブ情報を示す図

163

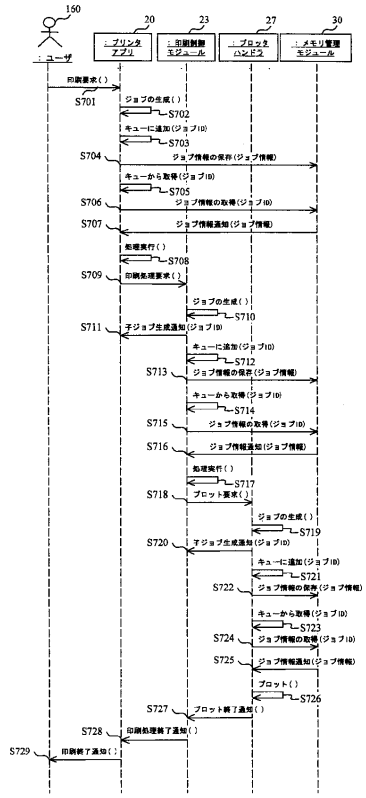
項目	値
ジョブID	1
ジョブ種別	プロッタ
状態	実行中
ジョブ関連情報	→
・ ・ ・	・ ・ ・

164

項目	値
子ジョブのジョブID	2
子ジョブのジョブID	3

【図４６】

第4のジョブ関連情報例におけるジョブ登録時の動作シーケンス図



【図４７】

第5のジョブ関連情報例を示す図

根幹ジョブのジョブID
2世代目のジョブID
・ ・ ・
親ジョブのジョブID
子ジョブのジョブID

【図４８】

第5のジョブ関連情報例におけるジョブ情報を示す図

167

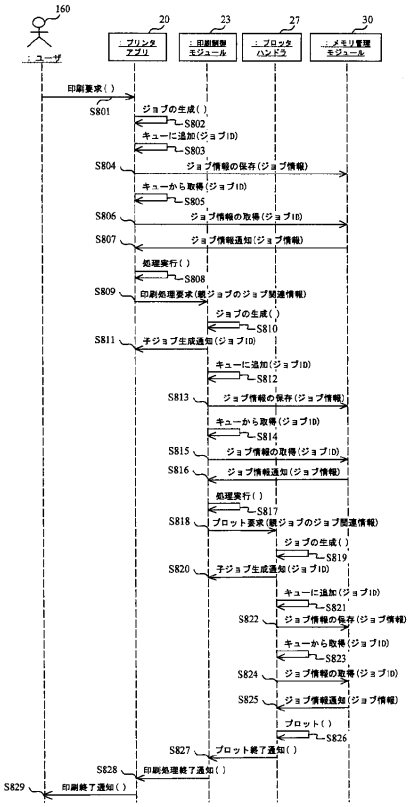
項目	値
ジョブID	4
ジョブ種別	プロッタ
状態	実行中
ジョブ関連情報	→
・ ・ ・	・ ・ ・

169

項目	値
根幹ジョブのジョブID	1
親ジョブのジョブID	2
子ジョブのジョブID	7

【図４９】

第5のジョブ関連情報例におけるジョブ登録時の動作シーケンス図



【図５０】

第6のジョブ関連情報例を示す図

自分のIDとその親ジョブのID
子ジョブのIDとその親のID
孫ジョブのIDとその親のID
ひ孫ジョブのIDとその親のID
・ ・ ・

【図５１】

第6のジョブ関連情報例におけるジョブ情報を示す図

170

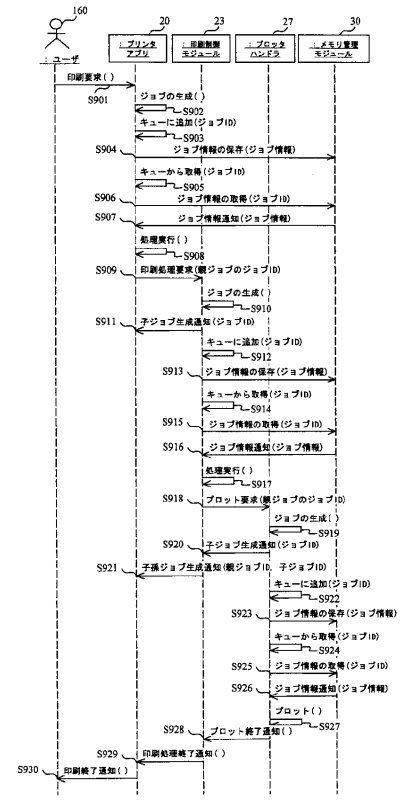
項目	値
ジョブID	1
ジョブ種別	プロッタ
状態	実行中
ジョブ関連情報	
・ ・ ・	・ ・ ・

171

	子ジョブ	親ジョブ
ジョブID	1	—
	2	1
	3	1
	4	2
	5	2
	6	3
	7	4

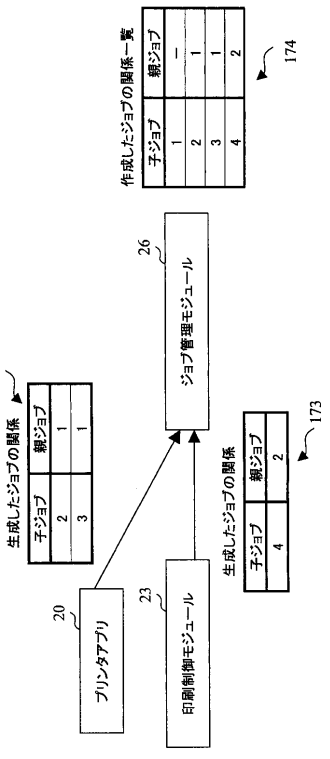
【図５２】

第6のジョブ関連情報例におけるジョブ登録時の動作シーケンス図



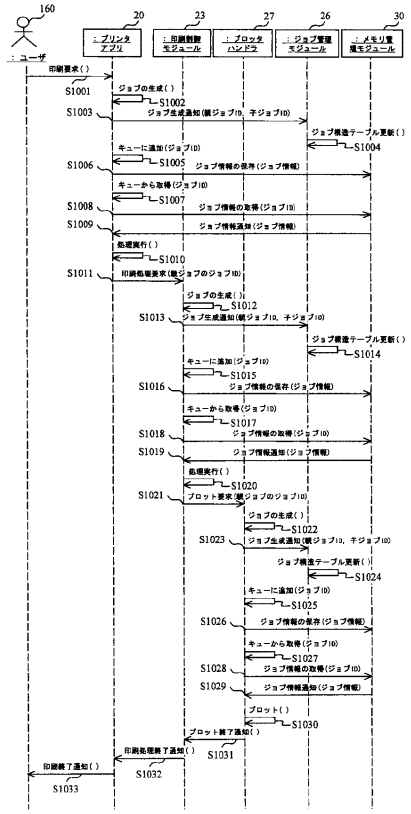
【図５３】

第7のジョブ関連情報例を示す図



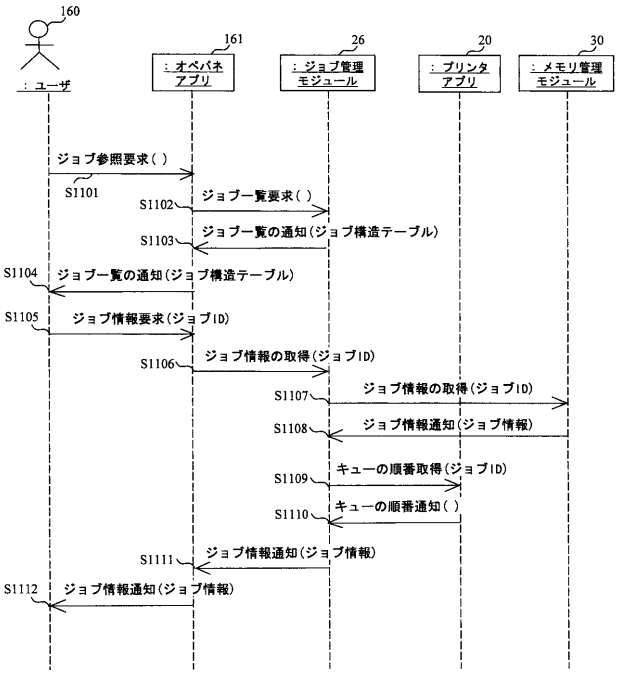
【図５４】

第7のジョブ関連情報例におけるジョブ登録時の動作シーケンス図



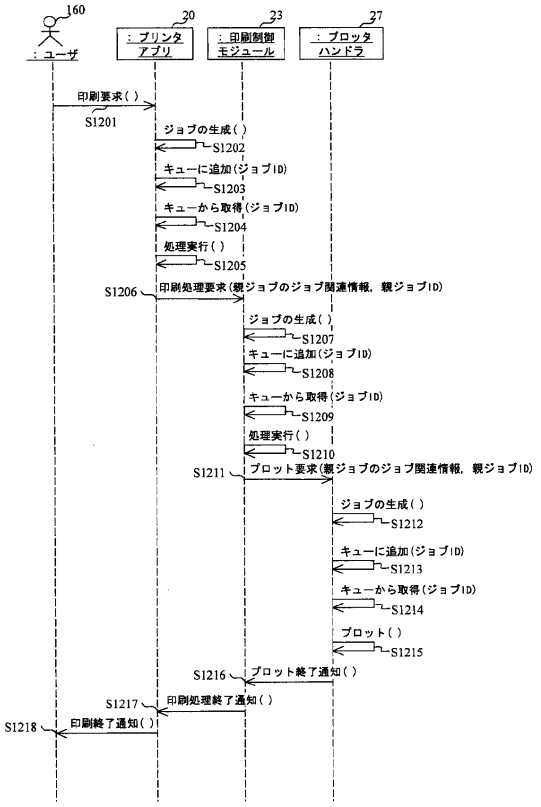
【図５５】

第7のジョブ関連情報例におけるジョブ参照時の動作シーケンス図



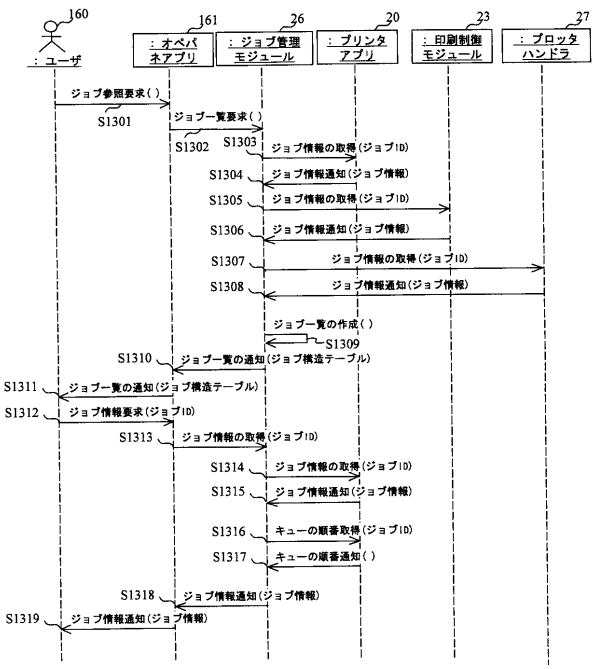
【図５６】

図30で示した構成におけるジョブ登録時の動作シーケンス図



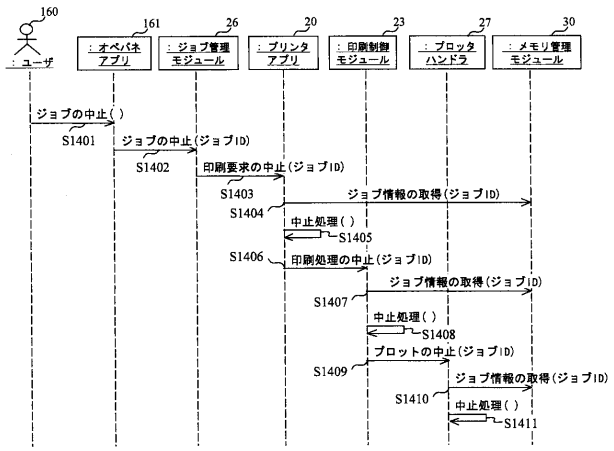
【図５７】

図30で示した構成におけるジョブ参照時の動作シーケンス図



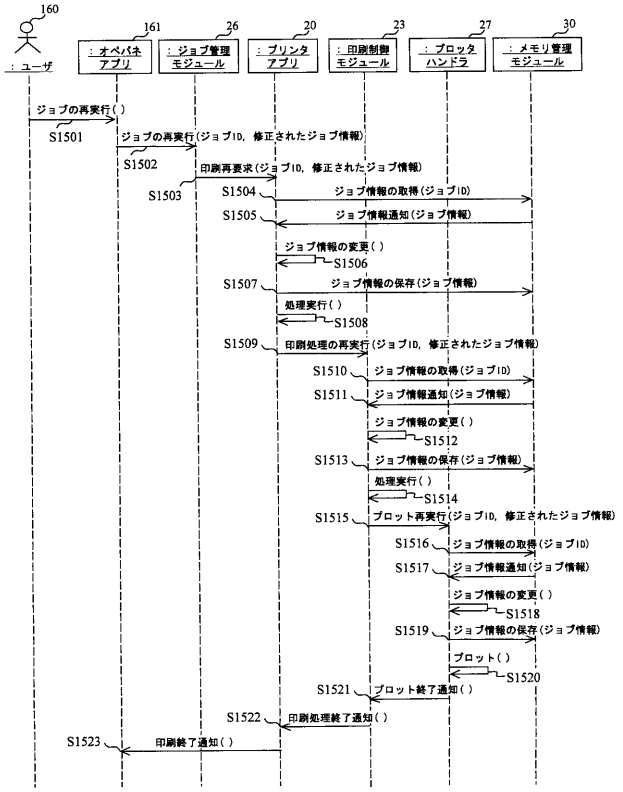
【図５８】

ジョブを中止する場合の処理を示すシーケンス図



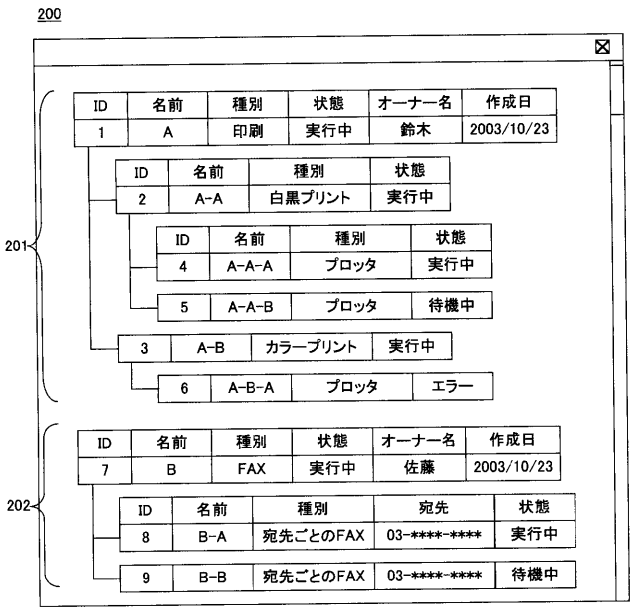
【図５９】

ジョブを再実行する場合の処理を示すシーケンス図



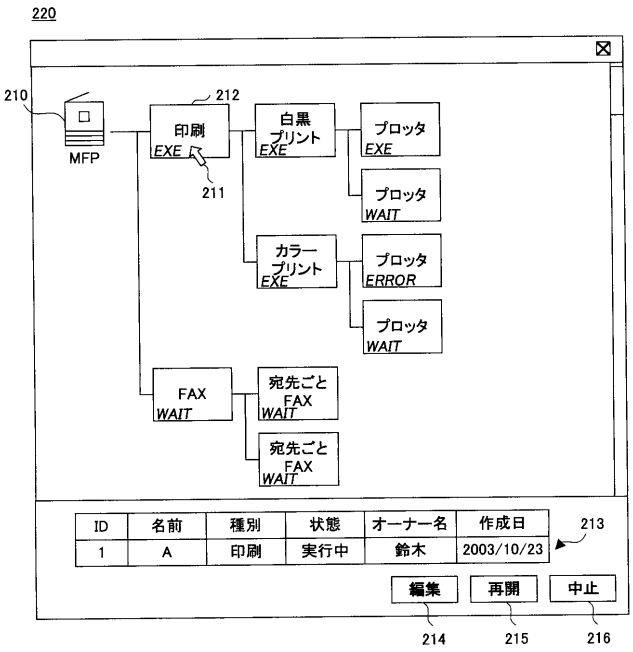
【図６０】

ユーザインタフェースを示す図である(その1)



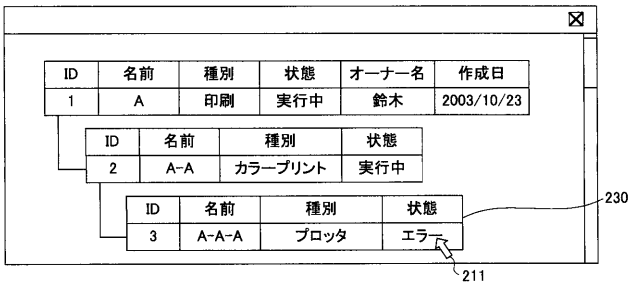
【図６１】

ユーザインタフェースを示す図である(その2)



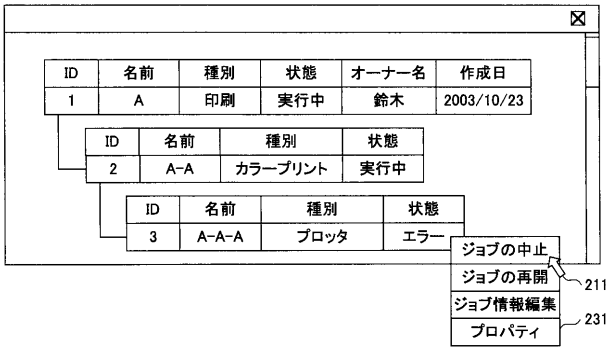
【図６２】

エラーの発生を示す図



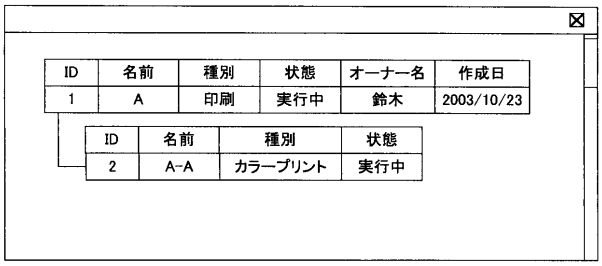
【図６３】

メニューの表示を示す図



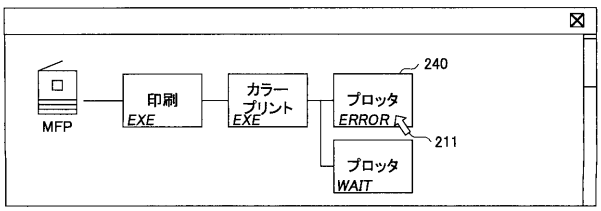
【図６４】

ジョブの中止を示す図



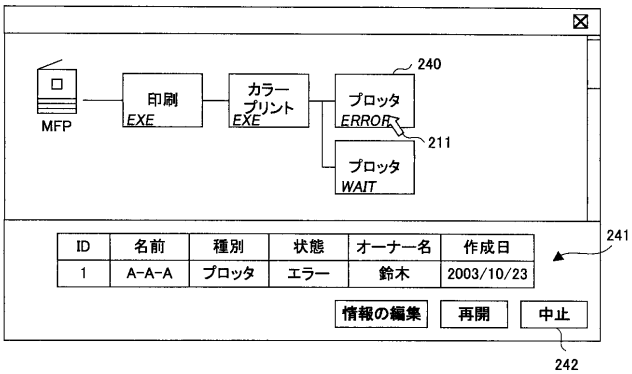
【図６５】

エラーの発生を示す図



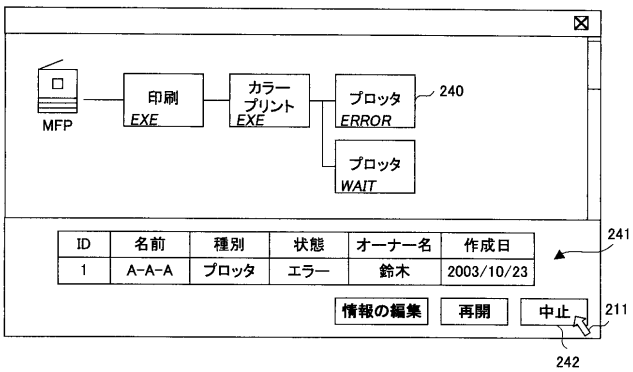
【図６６】

ジョブ情報の表示を示す図



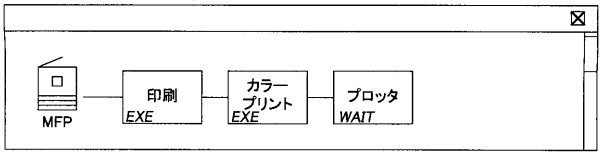
【図６７】

ジョブの中止を示す図



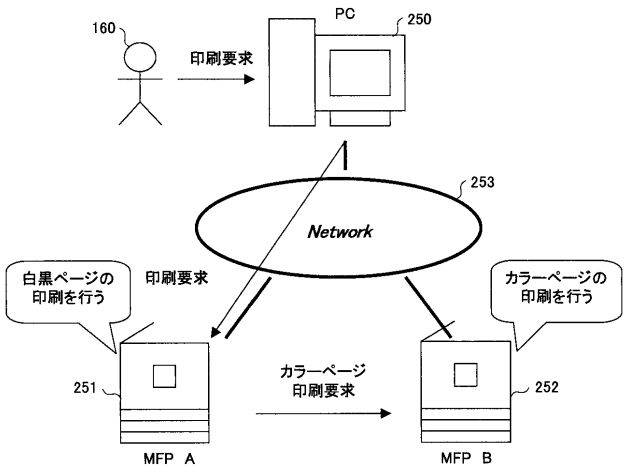
【図６８】

中止されたジョブの消滅を示す図



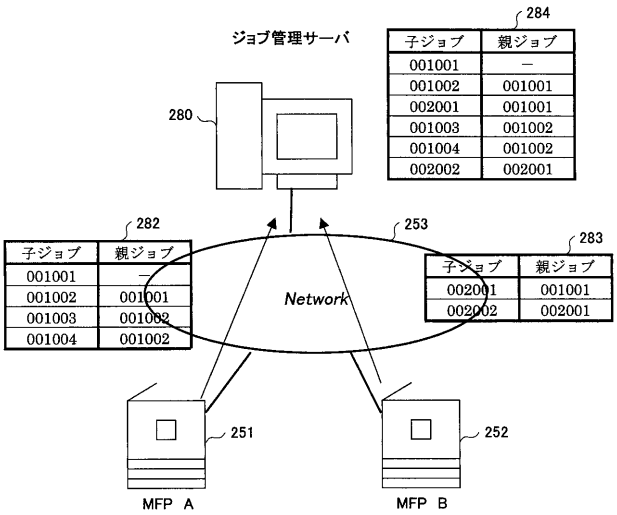
【図６９】

MFPの連携を示す図



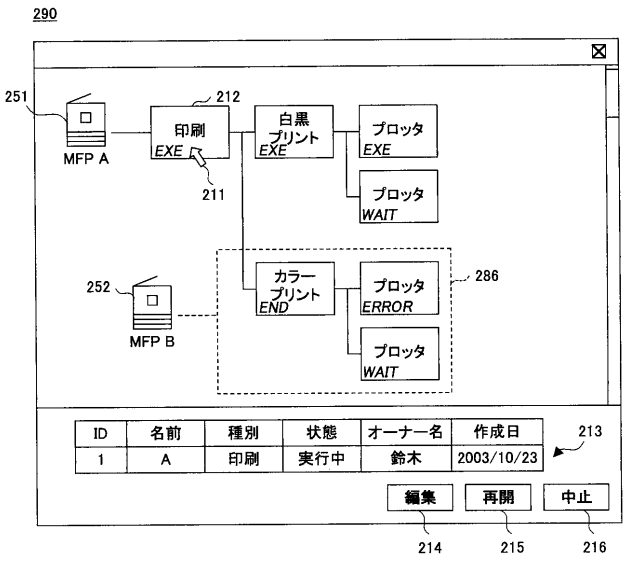
【図７０】

2台のMFPとジョブ管理サーバからなる構成を示す図



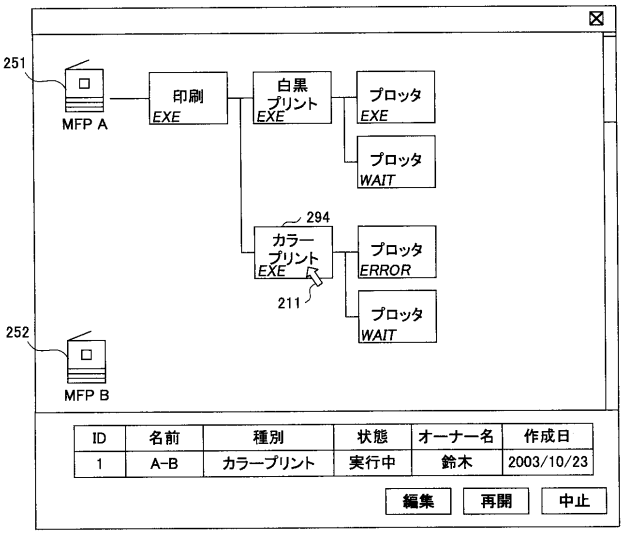
【図７１】

ジョブ管理サーバに表示される管理画面を示す図



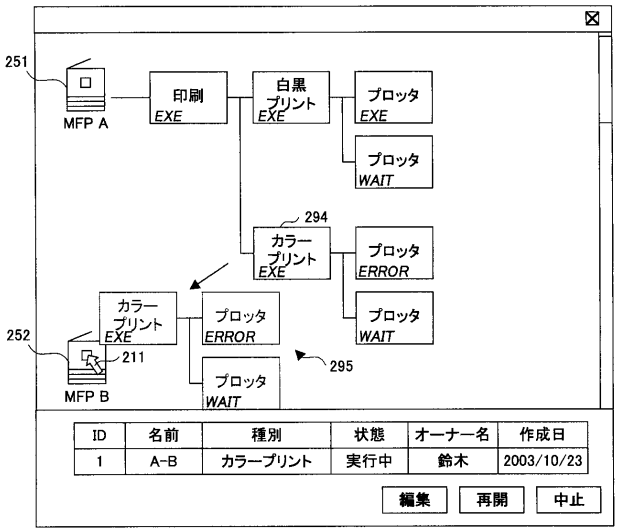
【図７２】

MFP Aのジョブを示す図



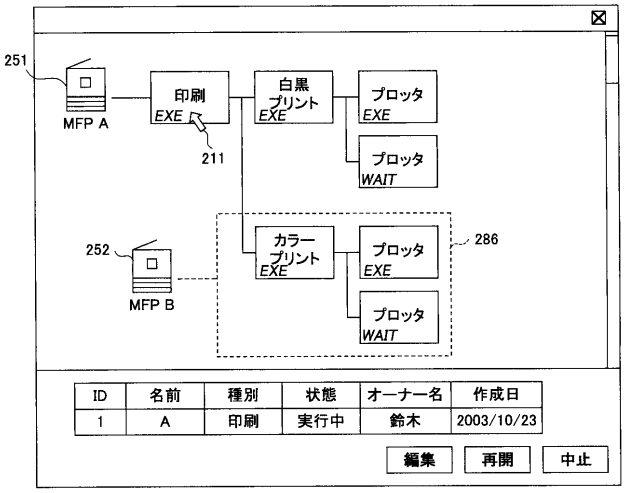
【図７３】

ジョブのアイコンをドラッグする様子を示す図



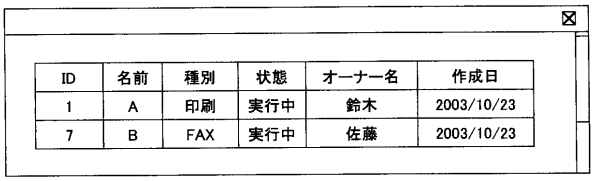
【図７４】

ドラッグされたジョブが他のMFPのジョブとなった様子を示す図



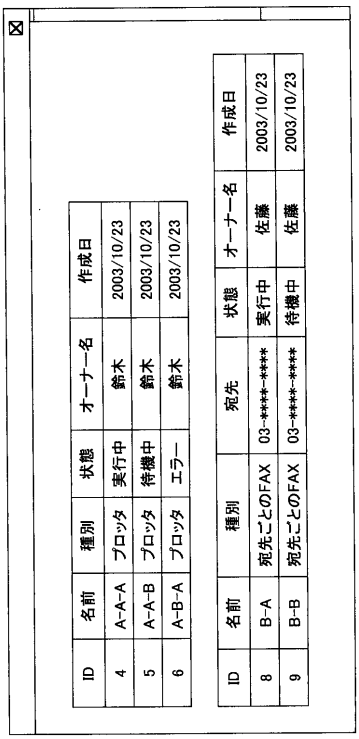
【図７５】

根幹ジョブのみの表示を示す図



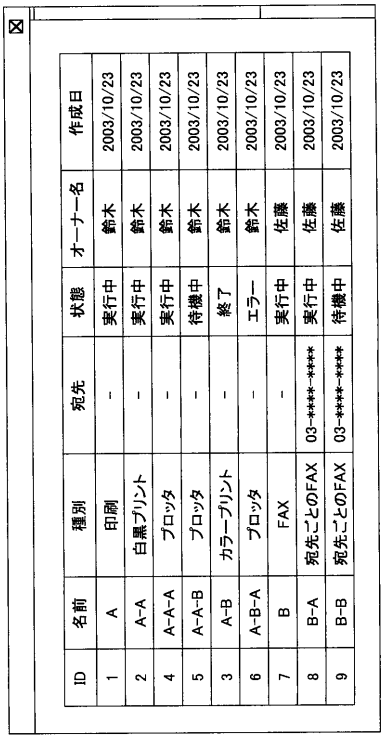
【図７６】

末端ジョブのみの表示を示す図



【図７７】

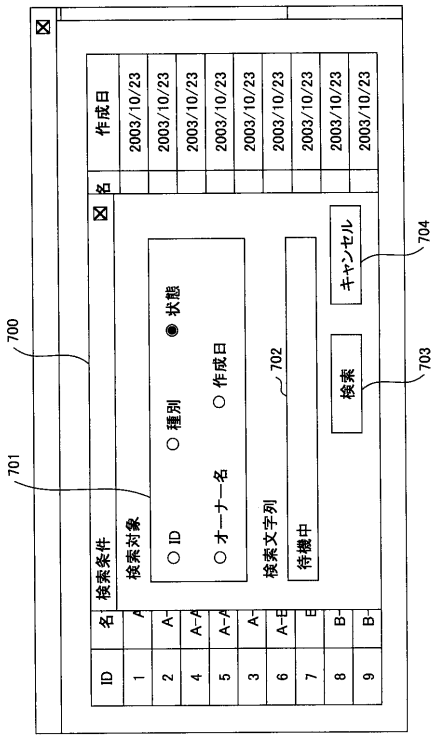
全てのジョブの一覧表示を示す図



ID	名前	種別	宛先	状態	オーナー名	作成日
1	A	印刷	-	実行中	鈴木	2003/10/23
2	A-A	白黒プリント	-	実行中	鈴木	2003/10/23
4	A-A-A	プロッタ	-	実行中	鈴木	2003/10/23
5	A-A-B	プロッタ	-	待機中	鈴木	2003/10/23
3	A-B	カラープリント	-	終了	鈴木	2003/10/23
6	A-B-A	プロッタ	-	エラー	鈴木	2003/10/23
7	B	FAX	-	実行中	佐藤	2003/10/23
8	B-A	宛先ごとのFAX	03-****-****	実行中	佐藤	2003/10/23
9	B-B	宛先ごとのFAX	03-****-****	待機中	佐藤	2003/10/23

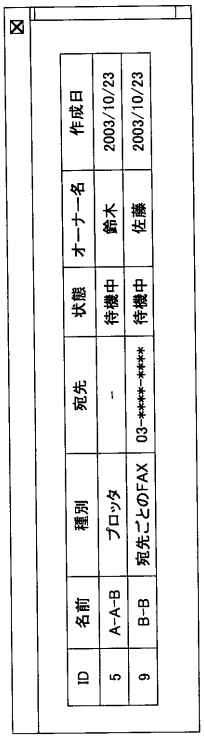
【図７８】

検索条件画面を示す図



【図７９】

検索結果を示す図



ID	名前	種別	宛先	状態	オーナー名	作成日
5	A-A-B	プロッタ	-	待機中	鈴木	2003/10/23
9	B-B	宛先ごとのFAX	03-****-****	待機中	佐藤	2003/10/23

フロントページの続き

(51)Int.Cl.[7]	FI		テーマコード（参考）
Ｇ０６Ｆ 3/12	Ｇ０６Ｆ 3/12	Ｃ	５Ｅ５０１
// Ｇ０６Ｆ 11/32	Ｇ０６Ｆ 11/32	Ｂ	

Ｆターム(参考) 5C062 AA05 AB11 AB23 AB42 AC22 AE15 AF14
5E501 AC24 BA05 CA02 DA09 FA46