(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第6201478号
(P6201478)

(45)発行日　平成29年9月27日(2017.9.27)　(24)登録日　平成29年9月8日(2017.9.8)

(51)Int.Cl.			F I		
G06F	9/48	(2006.01)	G06F	9/46	452Z
G06F	9/52	(2006.01)	G06F	9/46	472Z
			G06F	9/46	457

請求項の数 7　(全 19 頁)

(21)出願番号　特願2013-151711(P2013-151711)
(22)出願日　平成25年7月22日(2013.7.22)
(65)公開番号　特開2015-22626(P2015-22626A)
(43)公開日　平成27年2月2日(2015.2.2)
審査請求日　平成28年4月5日(2016.4.5)

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100129425
弁理士　小川　護晃
(74)代理人　100099623
弁理士　奥山　尚一
(74)代理人　100078330
弁理士　笹島　富二雄
(72)発明者　川崎　智弘
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

審査官　圓道　浩史

最終頁に続く

(54)【発明の名称】処理順序制御装置、処理順序制御方法及び処理順序制御プログラム

(57)【特許請求の範囲】
【請求項１】
１又は複数のコンピュータで動作する複数のプロセスが夫々実行する処理の実行順序を制御する制御プロセスが、当該制御プロセス及び前記プロセスによりアクセス可能な記憶手段に含まれる所定の記憶領域の複数の箇所に対して、ロックを実行し、
前記プロセスが、前記制御プロセスによりロックが実行された前記記憶領域のうち、各プロセスの自処理の実行順序に対応する箇所に対してロックを要求し、
前記制御プロセスが、前記プロセスからの前記ロックの要求を前記実行順序に対応する前記箇所毎に前記ロックが解放されるまで待機状態とし、
前記制御プロセスが、前記プロセスによりロックが要求された箇所のロックを、前記処理の前記実行順序と対応させて順次解放する
処理を含んだ処理順序制御方法。
【請求項２】
前記ロックを順次解放する処理は、前記制御プロセスが、前記プロセスにより実行を終了した処理に対応する箇所に対するロックが解放されると、当該箇所に対するロックを再実行し、当該箇所に対応する実行順序の次の実行順序に対応する箇所のロックを解放する、請求項１記載の処理順序制御方法。
【請求項３】
前記処理が複数の異なるコンピュータで実行され、前記制御プロセスが前記複数の異なるコンピュータの夫々で動作し、

前記制御プロセスの夫々と通信可能な総合制御プロセスが、前記制御プロセスの夫々に対し、前記複数の異なるコンピュータの夫々に設けられた前記記憶領域のうち前記制御プロセスの夫々が実行順序を制御する処理の実行順序に対応する箇所のロックを順次解放する指示を行い、

前記ロックを順次解放する処理は、前記制御プロセスの夫々が、前記総合制御プロセスによる指示に応じて、少なくとも前記記憶領域のうち自制御プロセスが実行順序を制御する処理の実行順序に対応する箇所のロックを順次解放する、請求項１又は２に記載の処理順序制御方法。

【請求項４】

前記ロックを順次解放する指示を行う処理は、総合制御プロセスが、前記制御プロセスの夫々に対し、同時に同一の実行順序に対応する箇所のロックを解放する指示を行い、

前記ロックを順次解放する処理は、前記制御プロセスの夫々が、前記総合制御プロセスによる指示に応じて、他の制御プロセスと同一の実行順序に対応する箇所のロックを順次解放する、請求項３記載の処理順序制御方法。

【請求項５】

前記制御プロセスは、前記実行順序の夫々に対応する箇所と当該実行順序を規定する通し番号のいずれかの値とを対応付け、当該通し番号のうちでより小さい前記値が対応付けられている処理から前記ロックを解放するようにし、

前記実行順序の夫々に対応する箇所には、新たな処理を実行するプロセスが空いている領域の箇所にロックをかけられるように、少なくとも１つの処理の実行順序に対応する箇所分の間隔を空けて前記値が夫々対応付けられている、請求項１又は２に記載の処理順序制御方法。

【請求項６】

コンピュータで動作する複数のプロセスが夫々実行する処理の実行順序を制御する制御プロセスが、当該制御プロセス及び前記プロセスによりアクセス可能な記憶手段に含まれる所定の記憶領域の複数の箇所に対して、ロックを実行し、

前記プロセスが、前記制御プロセスによりロックが実行された前記記憶領域のうち、各プロセスの自処理の実行順序に対応する箇所に対してロックを要求し、

前記制御プロセスが、前記プロセスからの前記ロックの要求を前記実行順序に対応する前記箇所毎に前記ロックが解放されるまで待機状態とし、

前記制御プロセスが、前記プロセスによりロックが要求された箇所のロックを、前記処理の前記実行順序と対応させて順次解放する

処理をコンピュータに実行させる処理順序制御プログラム。

【請求項７】

複数のプロセスが夫々処理を実行する処理実行部と、

前記処理の実行順序を制御する制御プロセスを実行する制御プロセス実行部と

を含み、

前記制御プロセスが、当該制御プロセス及び前記プロセスによりアクセス可能な記憶手段に含まれる所定の記憶領域の複数の箇所に対して、ロックを実行し、

前記プロセスが、前記制御プロセスによりロックが実行された前記記憶領域のうち、各プロセスの自処理の実行順序に対応する箇所に対してロックを要求し、

前記制御プロセスが、前記プロセスからの前記ロックの要求を前記実行順序に対応する前記箇所毎に前記ロックが解放されるまで待機状態とし、

前記制御プロセスが、前記プロセスによりロックが要求された箇所のロックを、前記処理の前記実行順序と対応させて順次解放する

処理順序制御装置。

【発明の詳細な説明】

【技術分野】

【０００１】

本発明は、コンピュータにおいて、一連の複数処理を順次実行する技術に関する。

【背景技術】
【０００２】
コンピュータにおいて複数処理を実行する技術がある。その一例として、プログラムの１つのメインスレッドに、そのスレッドが現在その資源の使用を必要とするか否かに関わらず、その資源に関連するロックの所有権を与えることにより、他のスレッドにまさる優先権を与える技術が提案されている。当該技術では、メインスレッドは、資源を使用し終え、別のスレッドが資源を使用するために待っていることを検知すると、その資源についての資源ロックを解放する。
【０００３】
ここで、コンピュータにおいて複数処理を順次実行する方法の一例として、処理を実行するプロセス同士で連携し、ある処理が終了すると、当該処理を実行するプロセスが次の実行順序の処理を実行するプロセスに対し処理の開始指示を行う方法がある。また、他の一例として、複数の処理の実行順序を制御する制御プロセスが、各処理を実行するプロセスに対して順次処理の開始指示を行う方法がある。
【先行技術文献】
【特許文献】
【０００４】
【特許文献１】特開平９－１９８２６５号公報
【発明の概要】
【発明が解決しようとする課題】
【０００５】
しかし、上記のような方法では、例えばシステムの更新等に伴って処理の実行順序を変更する際に、少なくとも、各処理を実行するプロセスや制御プロセスが行う開始指示の相手先（次に実行する処理）を変更する作業が発生する。また、例えば、各処理を実行するプロセス構成やコンピュータを変更する際等にも、必要に応じて各処理を実行するプロセスや制御プロセスの開始指示の相手先の詳細設定等を変更する作業が発生する。このため、システムの更新を柔軟に行うことが難しい。
【０００６】
そこで、本技術は、１つの側面において、複数の処理を順次実行するシステムにおけるシステムの更新を容易にすることを目的とする。
【課題を解決するための手段】
【０００７】
本技術における１つの側面では、１又は複数のコンピュータで動作する複数のプロセスが夫々実行する処理の実行順序を制御する制御プロセスが、当該制御プロセス及び前記プロセスによりアクセス可能な記憶手段に含まれる所定の記憶領域の複数の箇所に対して、ロックを実行する。そして、プロセスが、制御プロセスによりロックが実行された記憶領域のうち、各プロセスの自処理の実行順序に対応する箇所に対してロックを要求する。さらに、制御プロセスが、プロセスからのロックの要求を実行順序に対応する箇所毎にロックが解放されるまで待機状態とする。制御プロセスが、プロセスによりロックが要求された箇所のロックを、上記処理の実行順序と対応させて順次解放する。
【発明の効果】
【０００８】
本技術における１つの側面によれば、複数の処理を順次実行するシステムにおけるシステムの更新を容易にすることができる。
【図面の簡単な説明】
【０００９】
【図１】コンピュータにおいて複数の処理を実行するシステム構成の一例を示す図である。
【図２】処理の実行順序の制御の概要の説明図である。
【図３】処理の実行順序の制御の一例の説明図である。

【図４】処理の実行順序の制御の一例の説明図である。
【図５】制御プロセス及び各処理が実行する、処理の実行順序の制御処理の一例のフローチャートである。
【図６】ファイルの領域における各処理に対応する箇所の配置の一例の説明図である。
【図７】コンピュータにおいて複数の処理を実行するシステム構成の一例を示す図である。
【図８】コンピュータにおいて複数の処理を実行するシステム構成の一例を示す図である。
【図９】処理の実行順序の制御の一例の説明図である。
【図１０】処理の実行順序の制御の一例の説明図である。
【図１１】処理の実行順序の制御の一例の説明図である。
【図１２】処理の実行順序の制御の一例の説明図である。
【図１３】処理の実行順序の制御の一例の説明図である。
【図１４】制御プロセス及び各処理が実行する、処理の実行順序の制御処理の一例のフローチャートである。
【図１５】ハードウェア構成の一例の説明図である。
【図１６】複数処理の実行順序の制御を行う技術の一例の説明図である。
【図１７】複数処理の実行順序の制御を行う技術の一例の説明図である。
【発明を実施するための形態】
【００１０】
本明細書で説明する技術は、複数処理を順次実行するシステムにおける処理の実行順序の制御を、記憶手段の任意の領域に対してロックをかけて排他制御を行うことができるファイルロックの技術を用いて行うことで、処理の実行順序の変更等を容易にする。なお、本明細書において「システム」とは、コンピュータを用いて処理を実行する構成全般を示し、処理を実行するコンピュータは１台であっても複数であってもよい。また、本明細書で説明するプロセスと処理との関係は全て一例に過ぎず、各処理を実行するプロセス構成は任意である。
【００１１】
［各種処理方法の問題点等］
まず、コンピュータによって複数処理を順次実行する複数の方法の例についていくつか列挙し、その問題点について説明する。ここでは、プロセスＡの処理Ａ、プロセスＢの処理Ｂ、プロセスＣの処理Ｃの順で処理を実行する場合における種々の方法について説明する。
【００１２】
例えば、図１６に示すように、処理を実行するプロセス同士が連携する方法１がある。方法１では、プロセスＡ９０１は、処理Ａの実行が完了すると、処理Ｂを実行するプロセスＢ９０２に対して通信を行い、処理Ｂの開始指示を通知する（９１１）。プロセスＢ９０２は、当該開始指示を受けて処理Ｂを実行し、処理Ｂの実行が完了すると、処理Ｃを実行するプロセスＣ９０３に対して通信を行い、処理Ｃの開始指示を通知する（９１２）。
【００１３】
しかし、この方法では、処理の実行順序を変更したい場合に、順序の変更を柔軟に行うことが難しい。例えば、処理Ａと処理Ｂの間に、新たにプロセスＤが実行する処理Ｄを追加するような場合、原則としてプロセスＡ９０１の通信論理の変更が必要となる。なお、「通信論理の変更」には、例えば、次に開始指示をする処理の変更をはじめ、変更後において次に実行される処理が変更前と異なるコンピュータで実行される場合には、通信先のコンピュータの変更を含め、必要に応じた通信プロトコル等の変更が含まれる。
【００１４】
さらに他の方法として、同じく図１６に示すように、制御プロセス９０４が、夫々の処理の実行順序や処理を実行するプロセスの情報を把握し、各プロセスと通信を行う方法２がある。方法２では、制御プロセス９０４は、プロセスＡ９０１に対して通信を行い、処

理Ａの開始指示を通知する（９２１）。プロセスＡ９０１は、処理Ａが完了すると、制御プロセス９０４に対して処理Ａの完了を通知する（９２２）。すると、制御プロセス９０４は、プロセスＢ９０２に対して処理Ｂの開始指示を通知する（９２３）。以下同様にして、図１６に示す９２４～９２６を経て、処理Ｂの次に処理Ｃが実行される。

【００１５】

しかし、この方法２でも、処理の実行順序を変更したい場合に、順序の変更を柔軟に行うことが難しい。処理の実行順序を変更するには、原則として少なくとも制御プロセスの通信論理の変更が必要となるからである。

【００１６】

また、プロセス間における通信を不要とする方法として、図１７に示すように、管理テーブル９０６において各処理の実行順序を一元管理し、各プロセスが、夫々管理テーブル９０６を参照しながら順次処理の実行を開始する方法３がある。方法３では、プロセスＡ９０１～プロセスＣ９０３の夫々が管理テーブルにアクセスし、自プロセスが実行する処理の実行可否フラグが「true」であることを検出すると、実行を開始する。例えば、処理Ａの実行可否フラグが「true」の場合、プロセスＡ９０１が、処理Ａの実行を開始する（９３１）。そして、プロセスＡ９０１は、処理Ａの完了後に管理テーブルにおける処理Ａに対応する実行可否フラグを「false」にし、次の実行順序の処理Ｂに対応する実行可否フラグを「true」にする（９３２）。するとプロセスＢ９０２が、処理Ｂの実行可否フラグが「true」になったことを検出して実行を開始する（９３３）。以下同様にして、図１７に示す９３４～９３６を経て、処理Ｂの次に処理Ｃが実行される。

【００１７】

この方法Ｃでは、プロセス同士が直接通信を行わないため、処理の実行順序を変更する場合に、プロセスの通信論理の変更が原則として不要である。しかし、方法Ｃでは、管理テーブル９０４に情報が集中するため、ファイル破壊等により管理テーブル９０４のデータに不具合が生じると、各処理の実行順序の制御が困難になる。また、複数のプロセスが管理テーブル９０４に同時にアクセスするため、データの不整合が発生し易い。さらには、処理実行中に管理テーブル９０４のデータを変更して処理の実行順序を変更する場合、実行中の処理と管理テーブル９０４のデータとの不整合も発生し易く、処理に誤動作が生じることもある。

以下に説明する技術は、上記のような問題点を解決するものである。以下、実施例を示して説明する。

【００１８】

［第１実施例］

＜システム構成＞

図１は、第１実施例におけるシステム構成を示す。本実施例では、システムが、サーバ１を備える。サーバ１では、プログラムがロードされて実行されることによって実現される、制御プロセス実行部１０が実行する制御プロセス１１が動作するとともに、処理実行部２０が実行するプロセスＡ２１、プロセスＢ２２及びプロセスＣ２３が動作している。プロセスＡ２１は処理Ａを、プロセスＢ２２は処理Ｂを、プロセスＣ２３は処理Ｃを夫々実行する。

【００１９】

また、サーバ１が備える記憶手段３０が、ファイル３１を備える。記憶手段３０には、制御プロセス１１及びプロセスＡ２１～プロセスＣ２３がアクセスすることができ、ファイル３１には、制御プロセス１１及びプロセスＡ２１～プロセスＣ２３が、ファイル３１に含まれる所定の領域に対してロックを実行することができる。当該所定の領域は、ユーザ等が任意に設定することができる。

【００２０】

＜処理の実行順序の制御の概要＞

ここで、第１実施例で実現する処理の実行順序の制御の概要について、図２～図４を参照して説明する。本説明では、プロセスＡの処理Ａ、プロセスＢの処理Ｂ、プロセスＣの

処理Ｃの順に実行する例を示して説明する。また、当該制御において、処理Ａ、処理Ｂ、処理Ｃは、夫々、ファイル３１の１バイト目、２バイト目、３バイト目を、実行順序の制御の対象領域として使用するものとする。換言すれば、ファイル３１のうち、処理Ａ、処理Ｂ、処理Ｃに対応する箇所が夫々１バイト目、２バイト目、３バイト目であり、これらの箇所は処理Ａ、処理Ｂ、処理Ｃの実行順序に対応させて配置されている。図２～図４に示す１０１～１０９では、ファイル３１に対してロックをする処理及びロックをしている状態を実線矢印、ロック要求がＷａｉｔ（待機）状態となる処理及びＷａｉｔ状態を破線矢印、アンロックをする処理をブロック矢印（白色矢印）で示す。また、以下、本明細書において、「処理」が行うこととして記載している内容は、「処理を実行するプロセス」が行うことと同義とする。

【００２１】

（１０１）制御プロセス１１は、ファイル３１の１バイト目～３バイト目をロックする。

（１０２）処理Ａ～処理Ｃが、夫々ファイル３１の１バイト目～３バイト目に対してロックを要求する。しかし、すでに（１０１）で制御プロセス１１によりロックされているため、処理Ａ～処理Ｃによるロック要求はＷａｉｔ状態となる。

【００２２】

（１０３）制御プロセス１１が、ファイル３１の１バイト目をアンロックする。すると、処理Ａによる１バイト目のロック要求のＷａｉｔ状態が解除される。

（１０４）処理Ａは、（１０３）でファイル３１の１バイト目のロック要求のＷａｉｔ状態が解除されると即時に１バイト目をロックする。そして、処理Ａは、処理Ａ本体の実行を開始する。

【００２３】

（１０５）制御プロセス１１は、ファイル３１の１バイト目に対して再度ロックを要求する。しかし、１バイト目は（１０４）ですでに処理Ａによりロックされているため、制御プロセス１１によるロック要求はＷａｉｔ状態となる。

（１０６）処理Ａは、処理Ａ本体の実行が終了すると、１バイト目をアンロックする。すると、制御プロセス１１による１バイト目のロック要求のＷａｉｔ状態が解除される。

【００２４】

（１０７）制御プロセス１１は、（１０６）でファイル３１の１バイト目のロック要求のＷａｉｔ状態が解除されると、即時に１バイト目をロックする。

（１０８）制御プロセス１１は、次に２バイト目をアンロックする。すると、処理Ｂによる２バイト目のロック要求のＷａｉｔ状態が解除される。

（１０９）処理Ｂは、（１０８）で２バイト目のロック要求のＷａｉｔ状態が解除されると、即時に２バイト目をロックする。そして、処理Ｂは、処理Ｂ本体の実行を開始する。

以降、同様にして、処理Ｂに続いて処理Ｃが実行される。

【００２５】

＜処理フローチャートの説明＞

図５は、制御プロセス１１及び各処理のうちの１つの処理である処理ｎが実行する、処理の実行順序の制御処理の一例につき、フローチャート及びファイル３１の領域例を用いて説明した図である。ステップＳ１～ステップＳ７が制御プロセス１１による処理を示し、ステップＳ１１～ステップＳ１４が処理ｎによる処理を示す。当該説明では、処理ｎは、ファイル３１のうちｎバイト目に対してロックをかけるものとして説明する。なお、図５では、処理ｎを実行するプロセスの図示は省略する。

【００２６】

ステップＳ１で、制御プロセス１１が、処理の実行順序の制御に用いるファイル３１の対象領域の全バイトをロックする。

一方、ステップＳ１１で、処理ｎが、自処理の実行順序に対応するｎバイト目に対してロックを要求する。しかし、制御プロセス１１がすでにｎバイト目を含めた対象領域の全バイトをロックしているため、処理ｎのロック要求はＷａｉｔ状態となる。

【００２７】

ステップＳ２で、制御プロセス１１が、変数ｉに１をセットする。なお、ステップＳ２とステップＳ１１とはどちらが先でもよい。
ステップＳ３で、制御プロセス１１が、ｉバイト目をアンロックする。
【００２８】
ここで、ｉがｎに等しいとき、処理ｎは、ｎバイト目に対するロック要求のＷａｉｔ状態が解除され、ステップＳ１２で、ｎバイト目のロックに成功する。そして、処理ｎは、ステップＳ１３で、処理ｎ本体の処理を実行する。
【００２９】
一方、ステップＳ４で、制御プロセス１１が、ｉバイト目（＝ｎバイト目）に対してロックを要求する。しかし、処理ｎがすでにｎバイト目をロックしているため、制御プロセス１１によるｉバイト目に対するロック要求はＷａｉｔ状態となる。なお、ステップＳ１３とステップＳ４とはどちらが先でもよい。
【００３０】
処理ｎは、処理ｎ本体の処理の実行が終了すると、ステップＳ１４で、領域のｎバイト目をアンロックする。
すると、制御プロセス１１は、ｉバイト目に対するロックのＷａｉｔ状態が解除され、ステップＳ５で、ｉバイト目のロックに成功する。
【００３１】
ステップＳ６で、制御プロセス１１は、変数ｉが最大値（ｍａｘ）であるか否かを判定する。この最大値は、実行順序の制御対象となる処理の数の最大数として見込まれる値であり、例えばユーザが予め指定しておくことができる。変数ｉが最大値であれば（Ｙｅｓ）、処理を終了し、変数ｉが最大値でなければ（Ｎｏ）、ステップＳ７に進む。
【００３２】
ステップＳ７で、制御プロセス１１は、変数ｉに１を加算してステップＳ３に戻り、ファイル３１の次の箇所の処理を開始する。
なお、処理ｎがｉバイト目に対してロックをかけない場合、制御プロセス１１はステップＳ４でｉバイト目にロック要求をかけ、すぐにステップＳ５でロックに成功することができる。この場合、制御プロセスは、ステップＳ６～ステップＳ７を経て、ファイルの次の箇所の処理を開始する。
【００３３】
＜本実施例による効果、変形例等＞
本技術では、このように複数の処理を順次実行するときに、ファイルロックの機能を用いて処理の実行順序を制御することにより、次のような効果を奏する。すなわち、本技術では、各処理を実行するプロセスは、記憶手段３０のファイル３１の対象領域のうち、自プロセスが実行する処理の実行順序に対応する箇所に対してロック又はアンロックを行うだけであり、他の処理やプロセスとの通信を行わない。一方、制御プロセス１１も同様であり、ファイル３１の対象領域に各処理の実行順序に対応させて配置された箇所に対して、実行順序と対応させて順次ロック又はアンロックを行うだけである。このように、各プロセスが疎に連携しているため、システムの更新によりプロセスの構成や処理の実行順序等を変更する場合に、変更作業が容易となる。具体的には、例えば、プロセスの構成を変更する場合であっても、実行順序の制御に伴うプロセスの通信論理を変更する必要がない。また、例えば、処理の実行順序を変更する場合には、各処理がロックをする領域を必要に応じて変更するだけでよい。
【００３４】
ここで、各処理に対応する箇所は、予め少なくとも１つの処理の実行順序に対応する箇所分の間隔を空けておいてもよい。そして、新たな処理を途中に追加する場合に、新たな処理がその空いている領域のうちの１箇所に対応するようにすれば（すなわち、新たな処理を実行するプロセスがその空いている領域のうちの１箇所にロックをかけるようにすれば）、他のプロセスや処理に対する変更が一切必要ない。その具体例を図６に示す。この例では、処理の実行順序の制御に用いる対象領域として、ファイル３１の１バイト目～１

０バイト目を確保しておき、処理Ａが１バイト目、処理Ｂが５バイト目、処理Ｃが１０バイト目にロックをかけるように、各処理に対応する箇所を配置する。この状態において、処理Ｂと処理Ｃとの間に新たにプロセスＤ２４の処理Ｄを追加する場合、例えば、処理Ｄが７バイト目をロックするようにする。このようにすれば、制御プロセス１１やプロセスＡ２１～プロセスＣ２３の処理Ａ～処理Ｃに一切変更を加えることなく、処理Ａ、処理Ｂ、処理Ｄ、処理Ｃの順に処理が実行されることとなる。

【００３５】

また、各処理がロックをする箇所を必要に応じて変更するだけで処理の実行順序を変更できるため、処理の実行中に他の処理の実行順序を変更しても、不整合が発生しづらい。

さらに、当該処理の実行順序の制御において使用する記憶領域は、単にロック又はアンロックされるだけであって、処理の実行順序を制御するための情報を持つ必要がないため、データの内容が破壊されることによる不整合も発生しづらい。

【００３６】

このため、本技術によれば、処理の実行順序の変更が容易になり、システムの更新にかかる工数を大幅に削減することができる。

なお、各処理の実行順序の制御に用いる記憶手段の対象領域は、ファイル形式に限らず、いかなる態様であってもよい。

【００３７】

ここで、上記説明では、１つのサーバ１内で制御プロセス１１並びにプロセスＡ２１の処理Ａ、プロセスＢ２２の処理Ｂ及びプロセスＣ２３の処理Ｃが動作し、かつ、ファイル３１も同一サーバにある例を用いて説明した。しかし、システム構成の変形例として、複数のサーバから参照可能な共有ファイルを設けることにより、各プロセスや処理が複数の異なるサーバで動作していても、同様の処理を実現することができる。そのようなシステムの一例を、図７に示す。図７に示すシステムでは、制御プロセス１１が動作するサーバ１－１と、プロセスＡ２１の処理Ａ、プロセスＢ２２の処理Ｂが動作するサーバ１－２と、プロセスＣ２３の処理Ｃが動作するサーバ１－３とを備える。そして、サーバ１－１～サーバ１－３と夫々ネットワーク接続され、制御プロセス１１及び各プロセスが夫々アクセス可能な記憶手段３２が、制御プロセス１１及び各プロセスによる共有ファイル３３を備える。このようなシステムにおいても、共有ファイル３３を上記説明のファイル３１と同様に用いることにより、上記と同様の処理を実現することができる。また、図示を省略するが、共有ファイルが、制御プロセス１１又はいずれかの処理が実行されるサーバ内に存在しても、上記と同様の処理を実現することができる。

【００３８】

［第２実施例］

第２実施例は、第１実施例と異なり、実行順序の制御対象となる処理を実行する全てのプロセスが共通してアクセスする記憶手段を設けることができない場合であっても、処理の実行順序の制御を実現するものである。

【００３９】

図８は、第２実施例におけるシステム構成を示す。本実施例では、システムが、マスターサーバ２、サーバ３－１及びサーバ３－２を備える。

マスターサーバ２では、サーバ３－１及びサーバ３－２で夫々動作する制御プロセス１５と通信可能であり制御プロセス１５の夫々に対して動作指示を行う、総合制御プロセス実行部１２が実行する総合制御プロセス１３が動作する。

【００４０】

サーバ３－１では、制御プロセス実行部１４－１が実行する制御プロセス１５－１が動作するとともに、処理実行部２０－１が実行するプロセスＡ２１及びプロセスＣ２３が動作している。プロセスＡは処理Ａを、プロセスＣは処理Ｃを夫々実行する。さらに、サーバ３－１が備える記憶手段３０－１が、ファイル３１－１を備える。制御プロセス１５－１、プロセスＡ２１で動作する処理Ａ及びプロセスＣ２３で動作する処理Ｃが、夫々ファイル３１－１にアクセスすることができ、ファイル３１－１に含まれる特定の箇所に対し

てロックをすることができる。
【００４１】
サーバ３－２では、同様に、制御プロセス実行部１４－２が実行する制御プロセス１５－２が動作するとともに、処理実行部２０－２が実行するプロセスＢ２２が動作している。プロセスＢ２２は処理Ｂを実行する。さらに、サーバ３－２が備える記憶手段３０－２が、ファイル３１－２を備える。制御プロセス１５－２、プロセスＢ２２で動作する処理Ｂが、ファイル３１－２にアクセスすることができ、ファイル３１－２に含まれる特定の箇所に対してロックをすることができる。
【００４２】
ここで、第２実施例で実現する処理の概要について、図９～図１３を参照して説明する。本説明では、処理Ａ、処理Ｂ、処理Ｃの順に実行する例を示して説明する。また、当該制御において、処理Ａ及び処理Ｃは夫々ファイル３１－１の１バイト目及び３バイト目を、処理Ｂは、ファイル３１－２の２バイト目を、実行順序の制御に使用するものとする。換言すれば、処理Ａ及び処理Ｃに対応する箇所が、ファイル３１－１のうち、夫々１バイト目及び３バイト目であり、処理Ｂに対応する箇所が、ファイル３１－２のうち、２バイト目である。これらの箇所は処理Ａ、処理Ｂ、処理Ｃの実行順序に対応させて配置されている。図９～図１３に示す１１１～１２３では、ファイル３１－１又はファイル３１－２に対してロックをする処理及びロックをしている状態を実線矢印、ロックがＷａｉｔ（待機）状態となる処理及びＷａｉｔ状態を破線矢印、アンロックをする処理をブロック矢印（白色矢印）で示す。さらに、マスターサーバ２の総合制御プロセス１３から各サーバ３の制御プロセス１５への指示を、一点鎖線矢印で示す。
【００４３】
（１１１）サーバ３－１の制御プロセス１５－１は、ファイル３１－１の１バイト目～３バイト目をロックする。また、サーバ３－２の制御プロセス１５－２も同様に、ファイル３１－２の１バイト目～３バイト目をロックする。
（１１２）サーバ３－１で動作する処理Ａ及び処理Ｃが、夫々ファイル３１－１の１バイト目及び３バイト目に対してロックを要求する。しかし、すでに（１１１）で制御プロセス１５－１によりロックされているため、処理Ａ及び処理Ｃによるロック要求はＷａｉｔ状態となる。一方、サーバ３－２で動作する処理Ｂも、ファイル３１－２の２バイト目に対してロックを要求する。しかし、同様に、すでに（１１１）で制御プロセス１５－２によりロックされているため、処理Ｂによるロック要求はＷａｉｔ状態となる。
【００４４】
（１１３）ここで、マスターサーバ２の総合制御プロセス１３が、全サーバの制御プロセス１５に対し、各サーバのファイル３１の１バイト目をアンロックするように指示を送信する。その後、総合制御プロセス１３は、全サーバからファイル３１の１バイト目の再ロックが成功した旨の通知を受信するまで待機する。
（１１４）サーバ３－１の制御プロセス１５－１が、ファイル３１－１の１バイト目をアンロックする。すると、処理Ａによる１バイト目のロック要求のＷａｉｔ状態が解除される。一方、サーバ３－２の制御プロセス１５－２も、ファイル３１－２の１バイト目をアンロックする。
【００４５】
（１１５）サーバ３－１の処理Ａは、（１１４）で１バイト目のロック要求のＷａｉｔ状態が解除されると即時にファイル３１－１の１バイト目をロックする。そして、処理Ａは、処理Ａ本体の実行を開始する。
（１１６）サーバ３－１の制御プロセス１５－１が、ファイル３１－１の１バイト目に対して再度ロックを要求する。しかし、１バイト目は（１１５）ですでに処理Ａによりロックされているため、制御プロセス１５－１によるロック要求はＷａｉｔ状態となる。一方、サーバ３－２の制御プロセス１５－２が、ファイル３１－２の１バイト目に対して再度ロックを要求する。ファイル３１－２の１バイト目は他の処理によってロックされていないため、制御プロセス１５－２は、再ロックに成功する。

【００４６】
（１１７）サーバ３－２の制御プロセス１５－２は、ファイル３１－２の１バイト目の再ロックに成功したことを、マスターサーバ２の総合制御プロセス１３に通知する。
（１１８）サーバ３－１の処理Ａは、処理Ａ本体の実行が終了すると、ファイル３１－１の１バイト目をアンロックする。すると、制御プロセス１５－１によるファイル３１－１の１バイト目のロック要求のＷａｉｔ状態が解除される。
【００４７】
（１１９）サーバ３－１の制御プロセス１５－１は、（１１８）でファイル３１－１の１バイト目のロック要求のＷａｉｔ状態が解除されると、即時にファイル３１－１の１バイト目をロックする。
（１２０）サーバ３－１の制御プロセス１５－１は、ファイル３１－１の１バイト目の再ロックに成功したことを、マスターサーバ２の総合制御プロセス１３に通知する。
【００４８】
（１２１）マスターサーバ２の総合制御プロセス１３が、全サーバの制御プロセスに対し、各サーバのファイル３１の２バイト目をアンロックするように指示を送信する。その後、総合制御プロセス１３は、全サーバからファイル３１の２バイト目の再ロックが成功したことの通知を受信するまで待機する。
（１２２）サーバ３－１の制御プロセス１５－１が、ファイル３１－１の２バイト目をアンロックする。一方、サーバ３－２の制御プロセス１５－２も、ファイル３１－２の２バイト目をアンロックする。すると、処理Ｂによるファイル３１－２の２バイト目のロック要求のＷａｉｔ状態が解除される。
【００４９】
（１２３）サーバ３－２の処理Ｂは、（１２２）でファイル３１－２の２バイト目のロック要求のＷａｉｔ状態が解除されると即時にファイル３１－２の２バイト目をロックする。そして、処理Ｂは、処理Ｂ本体の実行を開始する。
【００５０】
以降、同様にして、サーバ３－２の処理Ｂに続いて、サーバ３－１の処理Ｃが実行される。
なお、制御プロセスは、上記（１１１）の処理も、総合制御プロセス１３の指示によって行ってもよい。
【００５１】
＜処理フローチャートの説明＞
図１４は、マスターサーバ２の総合制御プロセス１３、各サーバ３の制御プロセス１５及び各処理のうちの１つの処理である処理ｎが実行する処理の一例につき、フローチャート及びファイルの領域例を用いて説明した図である。ステップＳ３１～ステップＳ３５が総合制御プロセス１３による処理を示す。また、ステップＳ４１～ステップＳ４６が制御プロセス１５による処理を示し、ステップＳ５１～ステップＳ５４が処理ｎによる処理を示す。当該説明では、処理ｎは、ファイルのうちｎバイト目に対してロックをかけるものとして説明する。なお、図１３及び図１３を参照した説明では、処理ｎが含まれるプロセスについての記載は省略する。
【００５２】
ステップＳ４１で、制御プロセス１５が、処理の実行順序の制御に用いるファイル３１の対象領域の全バイトをロックする。
一方、ステップＳ５１で、処理ｎが、処理ｎの実行順序に対応するｎバイト目に対してロックを要求する。しかし、制御プロセス１５がすでにｎバイト目を含めた対象領域の全バイトをロックしているため、処理ｎのロック要求はＷａｉｔ状態となる。
【００５３】
ステップＳ３１で、総合制御プロセス１３が、変数ｉに１をセットする。なお、ステップＳ４１及びステップ５１と、ステップＳ３１とはどちらが先になってもよい。
ステップＳ３２で、マスターサーバ２の総合制御プロセス１３が、全サーバの制御プロ

セス１５に対し、各サーバのファイル３１のｉバイト目をアンロックするように指示を送信する。その後、総合制御プロセス１３は、全サーバからファイル３１のｉバイト目の再ロックが成功したことの通知を受信するまで待機する。

【００５４】

一方、ステップＳ４２で、制御プロセス１５が、マスターサーバ２の総合制御プロセス１３からのステップＳ３２の指示を受信する。当該指示を受け、ステップＳ４３で、制御プロセス１５が、ｉバイト目をアンロックする。

【００５５】

ここで、ｉがｎに等しいとき、処理ｎは、ｎバイト目に対するロック要求のＷａｉｔ状態が解除され、ステップＳ５２で、ｎバイト目のロックに成功する。そして、処理ｎは、ステップＳ５３で、処理ｎ本体の処理を実行する。

【００５６】

一方、ステップＳ４４で、制御プロセス１５が、ｉバイト目（＝ｎバイト目）に対してロックを要求する。しかし、処理ｎがすでにｎバイト目をロックしているため、制御プロセス１５によるｉバイト目に対するロック要求はＷａｉｔ状態となる。なお、ステップＳ５３とステップＳ４４とはどちらが先になってもよい。

【００５７】

処理ｎは、処理ｎ本体の処理の実行が終了すると、ステップＳ５４で、領域のｎバイト目をアンロックする。

すると、制御プロセス１５は、ｉバイト目に対するロック要求のＷａｉｔ状態が解除され、ステップＳ４５で、ｉバイト目のロックに成功する。そして、ステップＳ４６で、総合制御プロセス１３に対し、ｉバイト目のロックが成功したことを通知する。

【００５８】

ステップＳ３４で、総合制御プロセス１３は、変数ｉが最大値（ｍａｘ）であるか否かを判定する。変数ｉが最大値であれば（Ｙｅｓ）、処理を終了し、変数ｉが最大値でなければ（Ｎｏ）、ステップＳ３５に進む。

ステップＳ３５で、総合制御プロセス１３は、変数ｉに１を加算してステップＳ３２に戻り、ファイル３１の次の箇所の処理を開始する。

【００５９】

＜本実施例による効果等＞

第２実施例によれば、第１実施例で説明した効果に加えて、次のような効果を奏する。すなわち、複数の異なるサーバで処理が実行されている場合において、各処理からアクセス可能な記憶手段や共有ファイルが存在しないシステム構成であっても、ファイルロックの機能を用いた処理の実行順序の制御が可能となる。

【００６０】

＜本実施例による効果、変形例等＞

なお、上記説明では、総合制御プロセス１３が、各処理を実行するサーバ３とは別のマスターサーバ２で動作していたが、各処理を実行する全てのサーバ３と通信可能であれば、総合制御プロセス１３は、例えばサーバ３のうちのいずれかで動作していてもよい。

その他の変形例等については、第１実施例と同様である。

【００６１】

［ハードウェア構成］

図１５は、前述の各実施例のサーバやマスターサーバとして機能する情報処理装置のハードウェア構成の一例を示す。本情報処理装置は、プロセッサ１０１、メモリ１０２、ストレージ１０３、可搬記憶媒体駆動装置１０４、入出力装置１０５及び通信インタフェース１０６を備える。

【００６２】

プロセッサ１０１は、制御ユニット、演算ユニット及び命令デコーダ等を含み、実行ユニットが、命令デコーダで解読されたプログラムの命令に従い、制御ユニットより出力される制御信号に応じ、演算ユニットを用いて算術・論理演算を実行する。かかるプロセッ

サ１０１は、制御に用いる各種情報が格納される制御レジスタ、既にアクセスしたメモリ２等の内容を一時的に格納可能なキャッシュ、及び、仮想記憶のページテーブルのキャッシュとしての機能を果たすＴＬＢを備える。なお、プロセッサ１０１は、ＣＰＵ（Central Processing Unit）コアが複数設けられている構成でもよい。

【００６３】

メモリ１０２は、例えばＲＡＭ（Random Access Memory）等の記憶装置であり、プロセッサ１０１で実行されるプログラムがロードされるとともに、プロセッサ１０１の処理に用いるデータが格納されるメインメモリである。また、ストレージ１０３は、例えばＨＤＤ（Hard Disk Drive）やフラッシュメモリ等の記憶装置であり、プログラムや各種データが格納される。可搬記憶媒体駆動装置１０４は、可搬記憶媒体１０７に記憶されたデータやプログラムを読み出す装置である。可搬記憶媒体１０７は、例えば磁気ディスク、光ディスク、光磁気ディスク又はフラッシュメモリ等である。プロセッサ１０１は、メモリ１０２やストレージ１０３と協働しつつ、ストレージ１０３や可搬記憶媒体１０７に格納されたプログラムを実行する。なお、プロセッサ１０１が実行するプログラムや、アクセス対象となるデータは、当該情報処理装置と通信可能な他の装置に格納されていてもよい。なお、上記各実施例で記載した記憶手段３０とは、メモリ１０２、ストレージ１０３及び可搬記憶媒体１０７若しくは当該情報処理装置と通信可能な他の装置の少なくともいずれかを示す。

【００６４】

入出力装置１０５は例えばキーボード等やディスプレイ等であり、ユーザ操作等による動作命令を受け付ける一方、情報処理装置による処理結果を出力する。通信インタフェース１０６は例えばＬＡＮ（Local Area Network）カード等であり、外部とのデータ通信を可能にする。前述した情報処理装置の各構成要素は、バス１０８で接続されている。

【００６５】

［その他］

本明細書で説明した情報処理装置の機能的構成及び物理的構成は、上述の態様に限るものではなく、例えば、各機能や物理資源を統合して実装したり、逆に、さらに分散して実装したりすることも可能である。

【００６６】

以上の実施形態に関し、更に以下の付記を開示する。

（付記１）

１又は複数のコンピュータで動作する１又は複数のプロセスが実行する複数の処理の実行順序を制御する制御プロセスが、当該制御プロセス及び前記プロセスがアクセス可能な記憶手段に含まれる所定の記憶領域に対してロックを実行し、

前記プロセスが、前記制御プロセスによりロックが実行された前記記憶領域のうち、前記複数の処理の実行順序に対応した箇所に対してロックを要求し、

前記制御プロセスが、前記プロセスによりロックが要求された箇所のロックを、前記複数の処理の実行順序と対応させて順次解放し、

前記プロセスが、要求していたロックが実行できると、当該箇所に対応する処理を実行する

処理を含んだ処理順序制御方法。

【００６７】

（付記２）

前記プロセスが、処理の実行を終了したときに、当該処理に対応する箇所に対するロックを解放し、

前記ロックを順次解放する処理は、前記制御プロセスが、前記プロセスにより実行を終了した処理に対応する箇所に対するロックが解放されると、当該箇所に対するロックを再実行し、当該箇所に対応する実行順序の次の実行順序に対応する箇所のロックを解放する、付記１記載の処理順序制御方法。

【００６８】

（付記３）
前記複数の処理が複数のコンピュータで実行され、前記制御プロセスが前記複数のコンピュータの夫々で動作し、
前記制御プロセスの夫々と通信可能な総合制御プロセスが、前記制御プロセスの夫々に対し、前記記憶領域のうち前記制御プロセスの夫々が実行順序を制御する処理の実行順序に対応する箇所のロックを順次解放する指示を行い、
前記ロックを順次解放する処理は、前記制御プロセスの夫々が、前記総合制御プロセスによる指示に応じて、少なくとも前記記憶領域のうち自制御プロセスが実行順序を制御する処理の実行順序に対応する箇所のロックを順次解放する、付記１又は２に記載の処理順序制御方法。

【００６９】
（付記４）
前記ロックを順次解放する指示を行う処理は、総合制御プロセスが、前記制御プロセスの夫々に対し、同時に同一の実行順序に対応する箇所のロックを解放する指示を行い、
前記ロックを順次解放する処理は、前記制御プロセスの夫々が、前記総合制御プロセスによる指示に応じて、他の制御プロセスと同一の実行順序に対応する箇所のロックを順次解放する、付記３記載の処理順序制御方法。

【００７０】
（付記５）
前記記憶領域のうち前記複数の処理の実行順序の夫々に対応する箇所には、少なくとも１つの処理の実行順序に対応する箇所分の間隔を空けて前記複数の処理が夫々対応付けられている、付記１又は２に記載の処理順序制御方法。

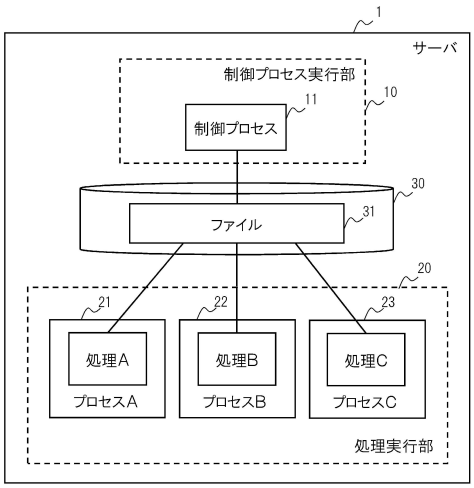
【００７１】
（付記６）
１又は複数のコンピュータで動作する１又は複数のプロセスが実行する複数の処理の実行順序を制御する制御プロセスが、当該制御プロセス及び前記プロセスがアクセス可能な記憶手段に含まれる所定の記憶領域に対してロックを実行し、
前記プロセスが、前記制御プロセスによりロックが実行された前記記憶領域のうち、前記複数の処理の実行順序に対応した箇所に対してロックを要求し、
前記制御プロセスが、前記プロセスによりロックが要求された箇所のロックを、前記複数の処理の実行順序と対応させて順次解放し、
前記プロセスが、要求していたロックが実行できると、当該箇所に対応する処理を実行する
処理をコンピュータに実行させる処理順序制御プログラム。

【００７２】
（付記７）
複数の処理を実行する１又は複数のプロセスを実行する処理実行部と、
前記複数の処理の実行順序を制御する制御プロセスを実行する制御プロセス実行部と
を含み、
前記制御プロセスが、当該制御プロセス及び前記プロセスがアクセス可能な記憶手段に含まれる所定の記憶領域に対してロックを実行し、
前記プロセスが、前記制御プロセスによりロックが実行された前記記憶領域のうち、前記複数の処理の実行順序に対応した箇所に対してロックを要求し、
前記制御プロセスが、前記プロセスによりロックが要求された箇所のロックを、前記複数の処理の実行順序と対応させて順次解放し、
前記プロセスが、要求していたロックが実行できると、当該箇所に対応する処理を実行する
処理順序制御装置。

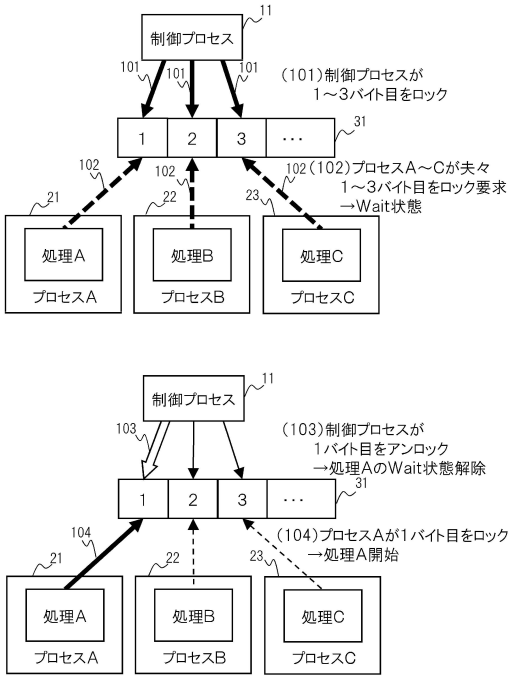
【符号の説明】
【００７３】

１、３…サーバ、２…マスターサーバ、１０…制御プロセス実行部、１１…制御プロセス、２０…処理実行部、２１…プロセス、３０、３２…記憶手段、３１、３３…ファイル

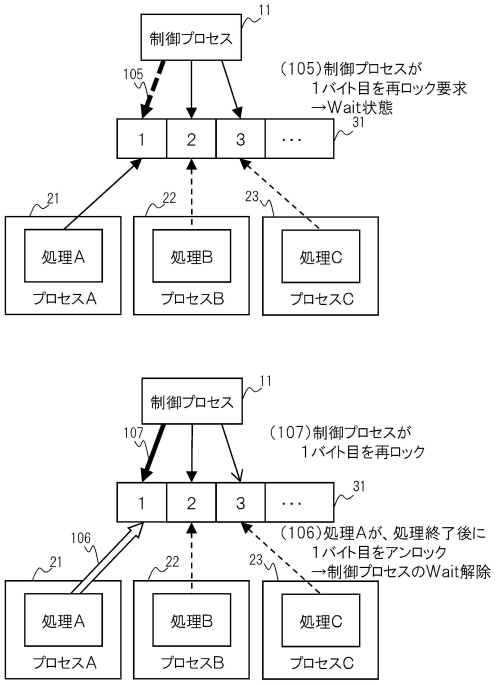
【図１】



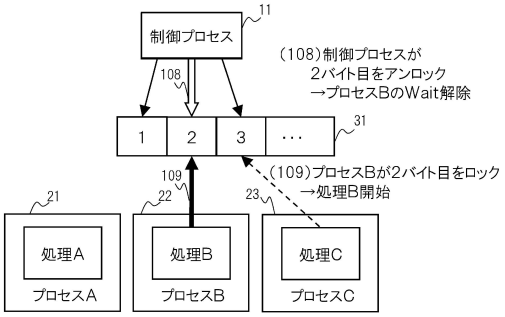
【図２】



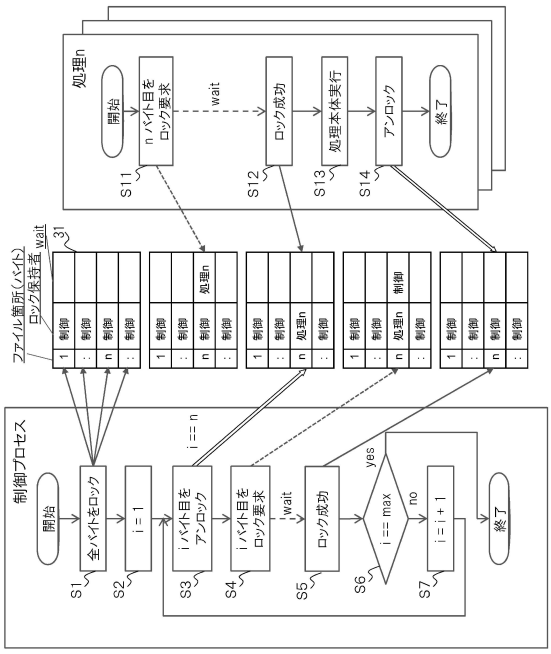
【図３】


制御プロセス
11
(105)制御プロセスが
1バイト目を再ロック要求
→Wait状態
105
1
2
3
…
31
21
22
23
処理A
処理B
処理C
プロセスA
プロセスB
プロセスC
制御プロセス
11
(107)制御プロセスが
1バイト目を再ロック
107
1
2
3
…
31
(106)処理Aが、処理終了後に
1バイト目をアンロック
→制御プロセスのWait解除
106
21
22
23
処理A
処理B
処理C
プロセスA
プロセスB
プロセスC

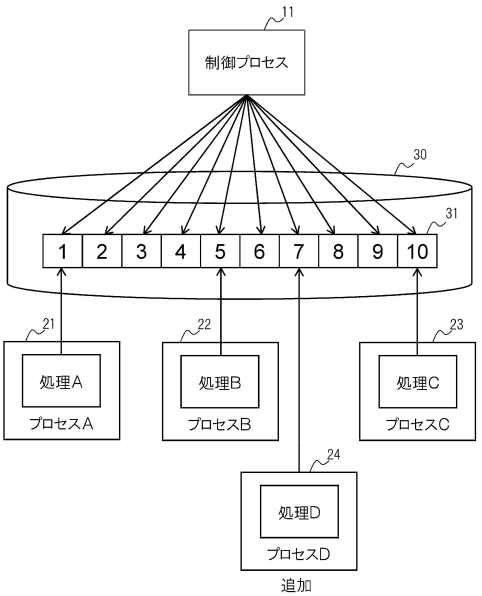
【図４】


制御プロセス
11
(108)制御プロセスが
2バイト目をアンロック
→プロセスBのWait解除
108
1
2
3
…
31
(109)プロセスBが2バイト目をロック
→処理B開始
109
21
22
23
処理A
処理B
処理C
プロセスA
プロセスB
プロセスC

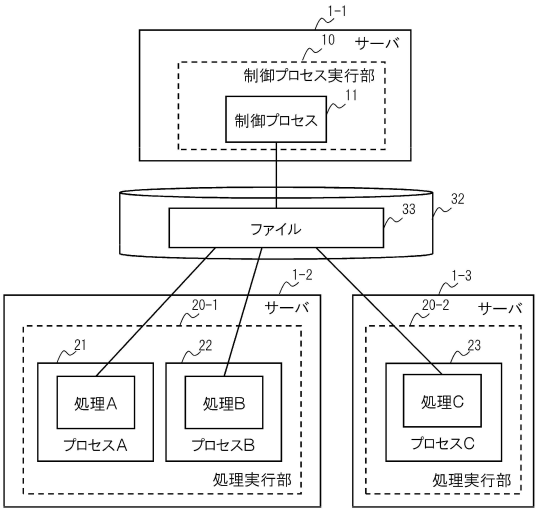
【図５】


処理n
開始
n バイト目を
ロック要求
wait
ロック成功
処理本体実行
アンロック
終了
S11
S12
S13
S14
31
ファイル箇所(バイト)
ロック保持者 wait
制御プロセス
開始
全バイトをロック
i = 1
i バイト目を
アンロック
i バイト目を
ロック要求
wait
ロック成功
i == max
yes
no
i = i + 1
終了
i == n
S1
S2
S3
S4
S5
S6
S7

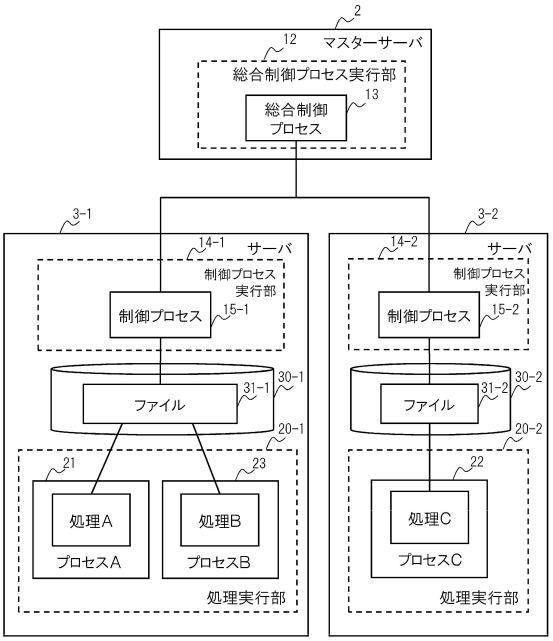
【図６】


制御プロセス
11
30
31
1
2
3
4
5
6
7
8
9
10
21
22
23
処理A
処理B
処理C
プロセスA
プロセスB
プロセスC
24
処理D
プロセスD
追加

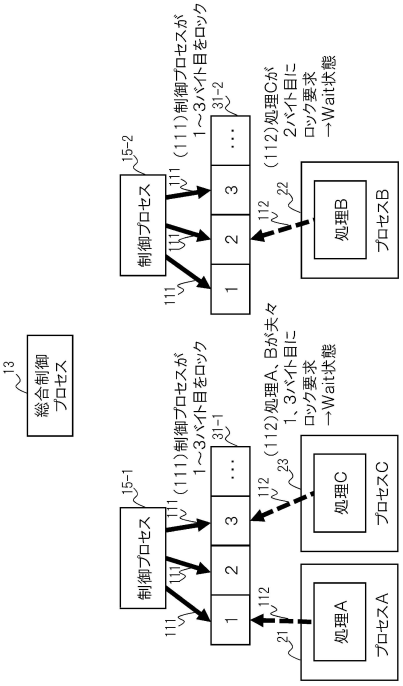
【図７】


1-1
10
サーバ
制御プロセス実行部
11
制御プロセス
32
33
ファイル
1-2
20-1
サーバ
21
22
処理A
処理B
プロセスA
プロセスB
処理実行部
1-3
20-2
サーバ
23
処理C
プロセスC
処理実行部

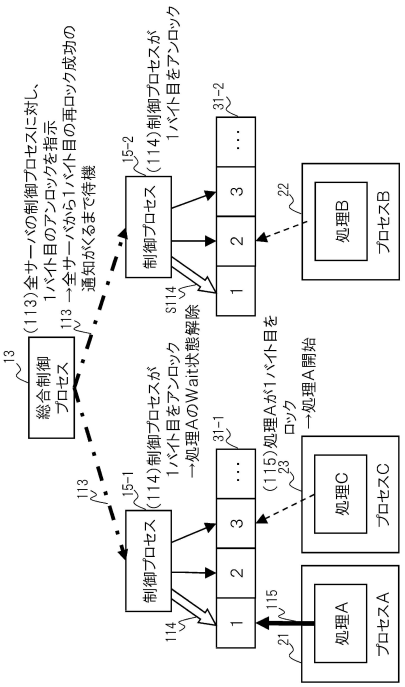
【図８】


2
12
マスターサーバ
総合制御プロセス実行部
13
総合制御プロセス
3-1
14-1
サーバ
制御プロセス実行部
15-1
制御プロセス
30-1
31-1
ファイル
20-1
21
23
処理A
処理B
プロセスA
プロセスB
処理実行部
3-2
14-2
サーバ
制御プロセス実行部
15-2
制御プロセス
30-2
31-2
ファイル
20-2
22
処理C
プロセスC
処理実行部

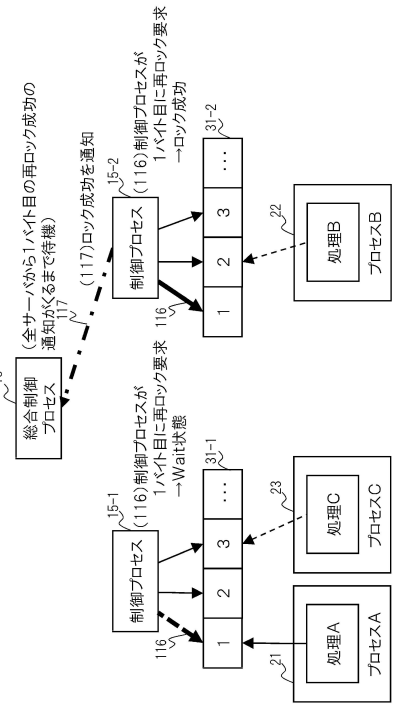
【図９】


13
総合制御プロセス
15-2
制御プロセス
(111)制御プロセスが
1～3バイト目をロック
111
31-2
1
2
3
…
(112)処理Cが
2バイト目に
ロック要求
→Wait状態
112
22
処理B
プロセスB
15-1
制御プロセス
(111)制御プロセスが
1～3バイト目をロック
111
31-1
1
2
3
…
(112)処理A、Bが夫々
1、3バイト目に
ロック要求
→Wait状態
112
23
処理C
プロセスC
112
21
処理A
プロセスA

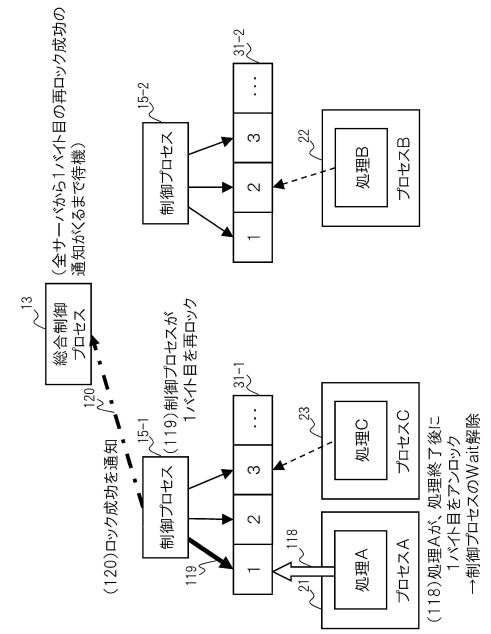
【図１０】


(113)全サーバの制御プロセスに対し、
1バイト目のアンロックを指示
113
→全サーバから1バイト目の再ロック成功の
通知がくるまで待機
13
総合制御プロセス
15-2
制御プロセス
(114)制御プロセスが
1バイト目をアンロック
31-2
1
2
3
…
S114
22
処理B
プロセスB
113
15-1
制御プロセス
(114)制御プロセスが
1バイト目をアンロック
→処理AのWait状態解除
31-1
1
2
3
…
(115)処理Aが1バイト目を
ロック
→処理A開始
23
処理C
プロセスC
114
115
21
処理A
プロセスA

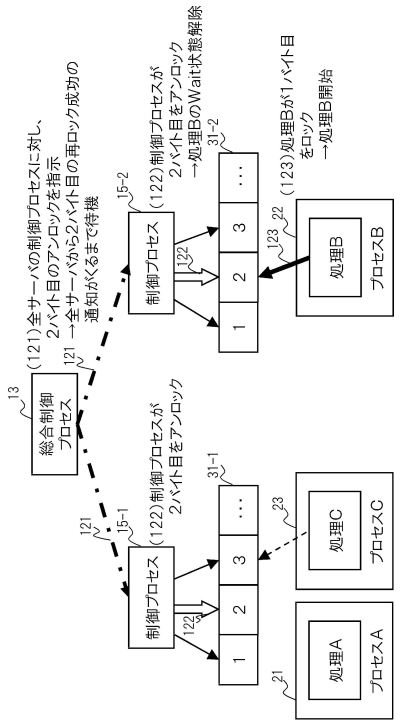
【図１１】



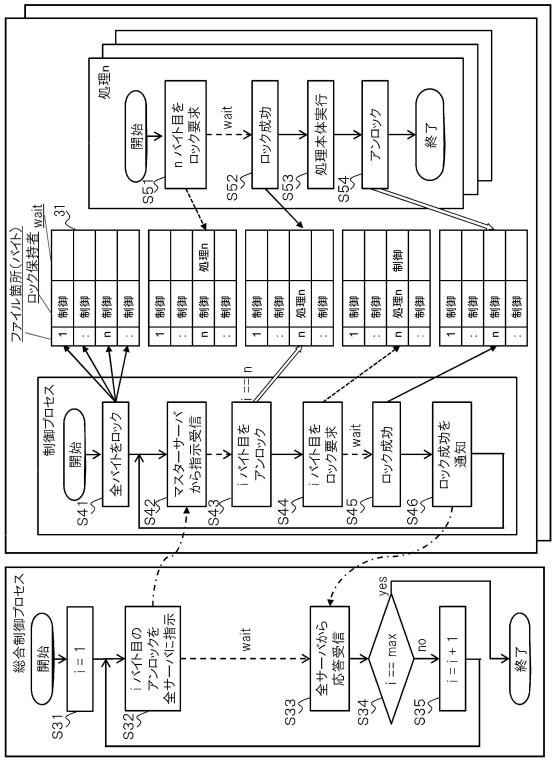
【図１２】



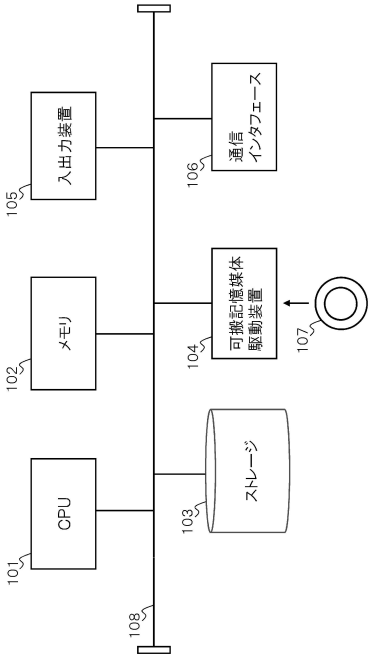
【図１３】



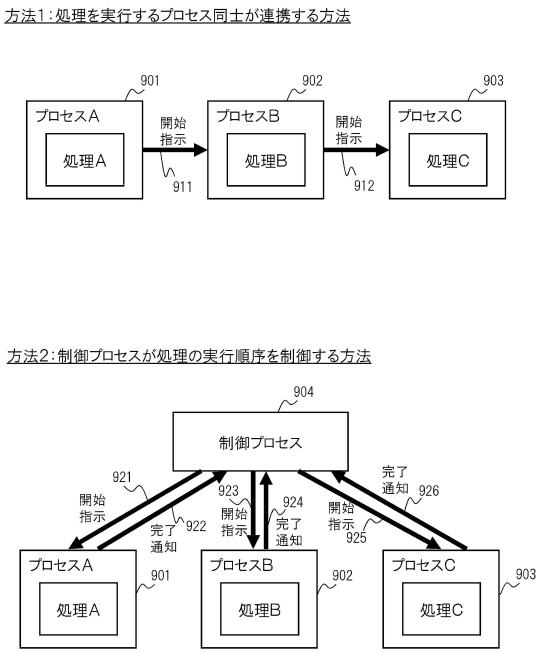
【図１４】



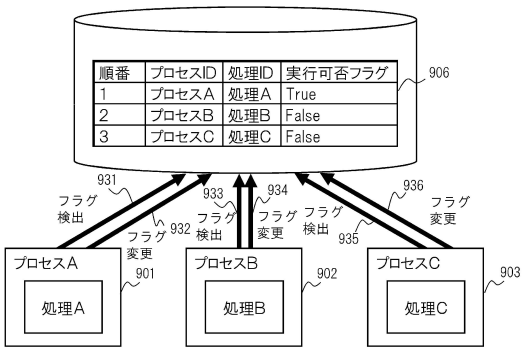
【図１５】


入出力装置
105
通信インタフェース
106
メモリ
102
可搬記憶媒体駆動装置
104
107
CPU
101
ストレージ
103
108

【図１６】


方法1：処理を実行するプロセス同士が連携する方法
901
プロセスA
処理A
開始指示
911
902
プロセスB
処理B
開始指示
912
903
プロセスC
処理C
方法2：制御プロセスが処理の実行順序を制御する方法
904
制御プロセス
921
開始指示
922
完了通知
923
開始指示
924
完了通知
925
開始指示
926
完了通知
プロセスA
901
処理A
プロセスB
902
処理B
プロセスC
903
処理C

【図１７】


方法3：テーブルを用いて処理の実行順序を制御する方法
順番 プロセスID 処理ID 実行可否フラグ
1 プロセスA 処理A True
2 プロセスB 処理B False
3 プロセスC 処理C False
906
931
フラグ検出
932
フラグ変更
933
フラグ検出
934
フラグ変更
935
フラグ検出
936
フラグ変更
プロセスA
901
処理A
プロセスB
902
処理B
プロセスC
903
処理C

フロントページの続き

(56)参考文献 特開平０４－１２７３２２（ＪＰ，Ａ）
特開平０９－１９８２６５（ＪＰ，Ａ）
特開平１１－０６５８６３（ＪＰ，Ａ）
米国特許第０６７７９０９０（ＵＳ，Ｂ１）
特開２０１１－１１８５８７（ＪＰ，Ａ）
特開２０１１－１２９０２４（ＪＰ，Ａ）
国際公開第２０１３／０６３０３１（WO，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／４６
９／４８
９／５０－ ９／５２
９／５４
１５／１６－１５／１７７