(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7257329号
(P7257329)

(45)発行日　令和5年4月13日(2023.4.13)

(24)登録日　令和5年4月5日(2023.4.5)

(51)国際特許分類
Ｇ０５Ｂ　19/042 (2006.01)

ＦＩ
Ｇ０５Ｂ　19/042

請求項の数　37　（全39頁）

(21)出願番号　特願2019-560214(P2019-560214)
(86)(22)出願日　平成30年4月30日(2018.4.30)
(65)公表番号　特表2020-518921(P2020-518921A)
(43)公表日　令和2年6月25日(2020.6.25)
(86)国際出願番号　PCT/US2018/030110
(87)国際公開番号　WO2018/204225
(87)国際公開日　平成30年11月8日(2018.11.8)
審査請求日　令和3年4月12日(2021.4.12)
(31)優先権主張番号　62/492,895
(32)優先日　平成29年5月1日(2017.5.1)
(33)優先権主張国・地域又は機関
米国(US)
(31)優先権主張番号　62/500,198
(32)優先日　平成29年5月2日(2017.5.2)
最終頁に続く

(73)特許権者　512132022
フィッシャー－ローズマウント システムズ，インコーポレイテッド
アメリカ合衆国 テキサス ７８６８１－７４３０ ラウンド ロック ウェスト ルイス ヘナ ブルバード １１００ ビルディング １ エマーソン プロセス マネージメント
(74)代理人　100096091
弁理士　井上 誠一
(72)発明者　マーク・ジェイ・ニクソン
アメリカ合衆国 テキサス州 ７８６８１ ラウンド ロック ブラックジャック ドライブ １５０３
(72)発明者　ファン・カルロス・ブラボー
最終頁に続く

(54)【発明の名称】　オープンアーキテクチャ産業制御システム

(57)【特許請求の範囲】
【請求項１】
産業環境内で物理的機能を実施する複数のフィールドデバイスを有する産業制御システムであって、
前記産業環境内の前記複数のフィールドデバイスに連結された物理的なプロセッサを含む入力／出力ノードと、
１つ以上の仮想コントローラノードであって、前記１つ以上の仮想コントローラノードの各々が、
１つ以上のプロセッサ、
メモリ、
前記メモリに記憶された複数の仮想コントローラであって、前記複数の仮想コントローラの各々は、前記産業環境内で前記フィールドデバイスの制御を実施するように前記１つ以上のプロセッサ上で実行可能であるもの、および、
前記メモリに記憶され、前記１つ以上の仮想コントローラノードの複数の位置にわたって前記複数の仮想コントローラを配置し、移動させることによって、前記複数の仮想コントローラの動作を管理するように前記１つ以上のプロセッサ上で実行可能なスーパーバイザ、を含む、前記１つ以上の仮想コントローラノードと、
前記入力／出力ノードを前記複数の仮想コントローラノードの各々に接続する通信ネットワークと、を備え、
前記物理的なプロセッサが、

（ｉ）前記産業環境内の前記フィールドデバイスのうちの１つ以上からデバイス信号を受信し、前記デバイス信号を処理し、前記処理されたデバイス信号を、前記通信ネットワーク上に配置し、配置されたデバイス信号は、前記複数の仮想コントローラの特定の１つ以上の、それぞれのハードウェア位置を指定することなく、また、前記複数の仮想コントローラの特定の１つ以上のそれぞれのネットワーク位置を指定することなく、それぞれ、前記複数の仮想コントローラの特定の１つ以上に届けるため、前記入力／出力ノードによってアドレス指定されており、
（ｉｉ）前記通信ネットワークを介して、前記複数の仮想コントローラのうちの１つ以上から制御信号を受信し、前記受信された制御信号を処理し、前記処理された制御信号を前記産業環境内の前記フィールドデバイスのうちの１つ以上に送信する、産業制御システム。
【請求項２】
前記１つ以上の仮想コントローラノードが、複数の仮想コントローラノードを含み、前記複数の仮想コントローラノードのうちの１つの前記複数の仮想コントローラのうち１つ以上が、前記仮想コントローラ通信を再構成することなく、前記複数の仮想コントローラノードのうちの前記１つから、前記複数の仮想コントローラノードのうちの別の１つへと移動可能である、請求項１に記載の産業制御システム。
【請求項３】
前記複数の仮想コントローラのうち前記１つ以上が、前記仮想コントローラ通信を再構成することなく、仮想コントローラノードの前記メモリの１つの場所から、前記仮想コントローラノードの前記メモリの別の場所へと移動可能である、請求項１に記載の産業制御システム。
【請求項４】
前記仮想コントローラノードのうちの少なくとも１つが、複数の異なるメモリデバイスを含み、前記複数の仮想コントローラのうち前記１つ以上が、前記仮想コントローラ通信を再構成することなく、前記仮想コントローラノードのうちの前記少なくとも１つの前記複数のメモリデバイスのうちの１つから、前記仮想コントローラノードのうちの前記少なくとも１つの前記複数のメモリデバイスのうちの別の１つへと移動可能である、請求項１に記載の産業制御システム。
【請求項５】
前記入力／出力ノードが、前記産業環境内の物理的入力／出力デバイスを介してフィールドデバイスからデバイス信号を受信するように、前記産業環境内の前記物理的入力／出力デバイスに連結された仮想入力／出力ルーチンを含む、請求項１に記載の産業制御システム。
【請求項６】
前記入力／出力ノードが、前記産業環境内のフィールドデバイスから直接デバイス信号を受信するように連結された仮想入力／出力ルーチンを含み、前記仮想入力／出力ルーチンが、前記フィールドデバイスからのデバイス信号、および前記フィールドデバイスに伝達された制御信号に対して入力／出力信号処理を実施するように、前記入力／出力ノード内の汎用プロセッサで実行される、請求項１に記載の産業制御システム。
【請求項７】
前記入力／出力ノードが、自己記述型通信スキームを使用して、デバイス信号を前記仮想コントローラノード内の前記複数の仮想コントローラの前記１つ以上に通信する仮想入力／出力ルーチンを含む、請求項１に記載の産業制御システム。
【請求項８】
前記入力／出力ノードが、複数の異なる通信プロトコルを使用して、前記通信ネットワークを介してデバイス信号を前記複数の仮想コントローラの前記１つ以上に通信する仮想入力／出力ルーチンを含む、請求項１に記載の産業制御システム。
【請求項９】
前記入力／出力ノードが、前記入力／出力ノードの前記メモリに記憶され、前記デバイス信号のうちの１つ以上を使用して、前記産業環境内の前記フィールドデバイスのうちの

１つを制御するための１つ以上の制御信号を生成するように、前記入力／出力ノードの前記プロセッサで実行される、制御ルーチンをさらに含む、請求項１に記載の産業制御システム。

【請求項１０】

前記１つ以上の仮想コントローラノード内の前記複数の仮想コントローラの前記１つ以上が、第１のレート以下で制御ルーチン実行を実施し、前記入力／出力ノード内の制御ルーチンが、前記第１のレートよりも大きい第２のレートで制御ルーチン実行を実施する、請求項９に記載の産業制御システム。

【請求項１１】

前記第２のレートが、前記第１のレートの５倍以上である、請求項１０に記載の産業制御システム。

【請求項１２】

前記入力／出力ノードが、デバイス信号を前記通信ネットワーク上で多重化する仮想入力／出力通信ルーチンを含む、請求項１に記載の産業制御システム。

【請求項１３】

前記入力／出力ノードが、複数の異なる通信プロトコルを介して複数のフィールドデバイスからデバイス信号を受信し、自己記述型通信スキームを使用して、前記デバイス信号を前記通信ネットワーク上で多重化する仮想入力／出力通信ルーチンを含む、請求項１に記載の産業制御システム。

【請求項１４】

前記入力／出力ノードが、複数の異なる通信プロトコルを介して複数のフィールドデバイスからデバイス信号を受信し、前記複数の異なる通信プロトコル内で、自己記述型通信スキームを使用して、前記デバイス信号を前記通信ネットワーク上で多重化する仮想入力／出力通信ルーチンを含む、請求項１に記載の産業制御システム。

【請求項１５】

前記自己記述型通信スキームが、第１の通信プロトコルにおける前記デバイス信号と、前記第１の通信プロトコルを記述するプロトコル記述信号と、を含むデバイス信号データを含む、請求項１４に記載の産業制御システム。

【請求項１６】

前記１つ以上の仮想コントローラノードの前記スーパーバイザが、前記１つ以上の仮想コントローラノード内の前記プロセッサの負荷バランシングを実施する、請求項１に記載の産業制御システム。

【請求項１７】

前記１つ以上の仮想コントローラノードのうちの少なくとも１つの前記スーパーバイザが、仮想コントローラを、前記仮想コントローラの前記仮想コントローラ通信を再構成することなく、前記仮想コントローラノードにおいて１つのプロセッサから別のプロセッサへと移動させるように構成されている、請求項１に記載の産業制御システム。

【請求項１８】

産業環境内で物理的機能を実施する複数のフィールドデバイスを有する産業制御システムであって、

１つ以上の通信チャネルを介して前記産業環境内の前記複数のフィールドデバイスに連結された物理的なプロセッサを含む入力／出力ノードと、

複数のコントローラノードであって、前記コントローラノードの各々が、

１つ以上のプロセッサ、

メモリ、

前記メモリに記憶された複数の仮想コントローラであって、前記複数の仮想コントローラの各々は、前記産業環境内で前記フィールドデバイスの制御を実施するように前記１つ以上のプロセッサ上で実行可能であるもの、および、

前記メモリに記憶され、前記複数の仮想コントローラノードの複数の位置にわたって前記複数の仮想コントローラを配置し、移動させることによって、前記コントローラノ

ードの前記複数の仮想コントローラの動作を管理するように前記１つ以上のプロセッサ上で実行可能なスーパーバイザ、を含む、前記複数のコントローラノードと、
オープン通信プロトコルを使用して、前記入力／出力ノードを前記１つ以上のコントローラノードの各々に接続するオープンプロトコル通信ネットワークと、を備え、
前記物理的なプロセッサが、
（ｉ）前記産業環境内の前記フィールドデバイスのうちの１つ以上からデバイス信号を受信し、オープン通信プロトコルを使用して、前記デバイス信号を前記通信ネットワーク上に配置し、配置されたデバイス信号は、前記複数の仮想コントローラの特定の１つ以上の、それぞれのハードウェア位置を指定することなく、また、前記複数の仮想コントローラの特定の１つ以上のそれぞれのネットワーク位置を指定することなく、それぞれ、前記複数の仮想コントローラの特定の１つ以上に届けるため、前記入力／出力プロセッサによってアドレス指定されており、
（ｉｉ）前記通信ネットワークを介して、前記コントローラのうちの１つ以上から制御信号を受信し、前記制御信号を前記産業環境内の前記フィールドデバイスのうちの１つ以上に送信する、産業制御システム。
【請求項１９】
前記オープンプロトコル通信ネットワークが、Ｅｔｈｅｒｎｅｔ通信ネットワークである、請求項１８に記載の産業制御システム。
【請求項２０】
前記オープンプロトコル通信ネットワークが、ＴＣＰ／ＩＰパケットベースのネットワーク通信スキームを使用する、請求項１８に記載の産業制御システム。
【請求項２１】
前記複数の仮想コントローラが、前記複数のコントローラノードのうちの前記１つにおいて前記プロセッサの第１のものから前記プロセッサの第２のものへと移動可能である、請求項１８に記載の産業制御システム。
【請求項２２】
前記入力／出力ノードが、自己記述型通信スキームを使用して、デバイス信号を前記コントローラノード内の前記仮想コントローラに通信する、請求項１８に記載の産業制御システム。
【請求項２３】
前記入力／出力ノードが、前記オープン通信プロトコルに従って構成されたパケットを介して複数の異なる通信プロトコルでデータを送信することによって、デバイス信号を前記コントローラノードの前記仮想コントローラに通信する、請求項１８に記載の産業制御システム。
【請求項２４】
前記入力／出力ノードが、異なる通信プロトコルスキームで構成されたデバイス信号を前記オープンプロトコル通信ネットワーク上で多重化する入力／出力通信ルーチンを含む、請求項１８に記載の産業制御システム。
【請求項２５】
前記入力／出力ノードが、複数の異なる通信プロトコルを介して前記複数のフィールドデバイスからデバイス信号を受信し、自己記述型通信スキームを使用して、前記デバイス信号を前記通信ネットワーク上で多重化する入力／出力通信ルーチンを含む、請求項１８に記載の産業制御システム。
【請求項２６】
前記自己記述型通信スキームが、第１の通信プロトコルにおける前記デバイス信号と、前記第１の通信プロトコルを記述するプロトコル記述信号と、を含むデバイス信号データを含む、請求項２５に記載の産業制御システム。
【請求項２７】
産業環境内で物理的機能を実施する複数のフィールドデバイスを有する産業制御システムであって、

１つ以上の通信チャネルを介して前記産業環境内の前記複数のフィールドデバイスに連結された物理的なプロセッサを含む入力／出力ノードと、
複数のコントローラノードであって、前記コントローラノードの各々が、
１つ以上のプロセッサ、
１つ以上のメモリ、および、
前記１つ以上のメモリに記憶された複数の仮想コントローラであって、前記仮想コントローラの各々は、前記１つ以上のプロセッサ上で、前記産業環境内で前記フィールドデバイスの制御を実施することを実行可能であるもの、を含む、前記複数のコントローラノードと、
前記入力／出力ノードを前記複数のコントローラノードの各々に接続する通信ネットワークと、を備え、
前記物理的なプロセッサが、前記産業環境内の前記フィールドデバイスのうちの１つ以上からデバイス信号を受信し、ハードウェアの場所に依存しない通信アドレス指定スキームを使用して、前記複数の仮想コントローラの特定の１つ以上に届けるため、前記デバイス信号を前記通信ネットワーク上に配置して、前記仮想コントローラの特定の１つ以上の、それぞれのハードウェア位置を指定することなく、また、前記複数の仮想コントローラの特定の１以上の、それぞれのネットワーク位置を指定することなく、１つ以上の仮想コントローラを特定する、産業制御システム。
【請求項２８】
前記ハードウェアの場所に依存しない通信アドレス指定スキームが、自己記述型通信スキームを含む、請求項２７に記載の産業制御システム。
【請求項２９】
前記自己記述型通信スキームが、第１の通信プロトコルにおける前記デバイス信号と、前記第１の通信プロトコルを記述するプロトコル記述信号と、を含むデバイス信号データを含む、請求項２８に記載の産業制御システム。
【請求項３０】
前記自己記述型通信スキームが、第１の通信プロトコルにおける前記デバイス信号と、前記フィールドデバイスのうちの１つに関して前記第１の通信プロトコルを記述するプロトコル記述信号と、を含むデバイス信号データを含み、ならびに第２の通信プロトコルにおける前記デバイス信号と、前記フィールドデバイスのうちの第２のものに関して前記第２の通信プロトコルを記述するプロトコル記述信号と、を含むデバイス信号データを含む、請求項２８に記載の産業制御システム。
【請求項３１】
前記通信ネットワークが、オープンプロトコル通信ネットワークである、請求項２８に記載の産業制御システム。
【請求項３２】
前記オープンプロトコル通信ネットワークが、ＴＣＰ／ＩＰパケットベースのネットワーク通信スキームを使用する、請求項３１に記載の産業制御システム。
【請求項３３】
前記コントローラノードのうちの１つが、複数の仮想コントローラを含む仮想コントローラノードであり、前記複数の仮想コントローラが、前記コントローラノードのうちの前記１つにおいて前記プロセッサの第１のものから前記プロセッサの第２のものへと移動可能である、請求項２７に記載の産業制御システム。
【請求項３４】
前記入力／出力ノードが、自己記述型通信スキームに従って構成されたパケットを介して複数の異なる通信プロトコルでデータを送信することによって、デバイス信号を前記仮想コントローラに通信する、請求項３３に記載の産業制御システム。
【請求項３５】
前記入力／出力ノードが、前記ハードウェアの場所に依存しない通信スキームを使用して、異なる通信プロトコルスキームで構成されたデバイス信号を前記通信ネットワーク上

で多重化する入力／出力通信ルーチンを含む、請求項２７に記載の産業制御システム。
【請求項３６】
前記入力／出力ノードが、複数の異なる通信プロトコルを介して複数のフィールドデバイスからデバイス信号を受信し、自己記述型通信スキームを使用して、前記デバイス信号を前記通信ネットワーク上で多重化する入力／出力通信ルーチンを含む、請求項２７に記載の産業制御システム。
【請求項３７】
コントローラノードが、ハードウェアの場所に依存しない通信アドレス指定スキームを使用して、前記通信ネットワークを介して制御信号を送信する、請求項２７に記載の産業制御システム。
【発明の詳細な説明】
【技術分野】
【０００１】
［関連出願］
これは、２０１７年５月１日に出願された「Ｏｐｅｎ　Ａｒｃｈｉｔｅｃｔｕｒｅ　Ｉｎｄｕｓｔｒｉａｌ　Ｃｏｎｔｒｏｌ　Ｓｙｓｔｅｍ」と題する米国仮特許出願第６２／４９２，８９５号、および２０１７年５月２日に出願された「Ｏｐｅｎ　Ａｒｃｈｉｔｅｃｔｕｒｅ　Ｉｎｄｕｓｔｒｉａｌ　Ｃｏｎｔｒｏｌ　Ｓｙｓｔｅｍ」と題する米国仮特許出願第６２／５００，１９８号の優先権および出願日の利益を主張する正式に出願された出願であり、その全体の開示が、参照により本明細書に明示的に組み込まれる。
【０００２】
本特許出願は、概して、産業およびプロセス制御システムに関し、かつより具体的には、信頼性、回復力、応答性、および弾性を提供するオープンアーキテクチャ産業制御システムに関する。
【背景技術】
【０００３】
化学、石油、または他のプロセスプラントにおいて使用されるものなどのプロセスまたは産業制御システムは、典型的には、アナログバス、デジタルバス、もしくはアナログ／デジタル結合バスを介して、または無線通信リンクもしくはネットワークを介して、１つ以上のフィールドデバイスに通信可能に連結される、１つ以上のプロセスコントローラを含む。例えば、バルブ、バルブポジショナ、スイッチ、およびトランスミッタ（例えば、温度センサ、圧力センサ、レベルセンサ、およびフローレートセンサ）である場合があるフィールドデバイスは、プロセス環境内に位置し、概して、バルブの開放または閉鎖、プロセスパラメータの測定等の物理的制御機能またはプロセス制御機能を実施して、プロセスプラント内またはシステム内で実行中の１つ以上のプロセスを制御する。周知のＦＯＵＮＤＡＴＩＯＮ（登録商標）フィールドバスプロトコルに適合するフィールドデバイスなどのスマートフィールドデバイスはまた、制御計算、警告機能、およびコントローラ内で一般に実装される他の制御機能も実施し得る。中央に位置し得るが、分散方法でプラント環境内にも位置し得るプロセスコントローラは、フィールドデバイスによって行われるプロセス測定を指示する信号および／またはフィールドデバイスに関する他の情報を受信し、例えば、プロセス制御判断を行い、受信した情報に基づき制御信号を発生させ、ＨＡＲＴ（登録商標）、ＷｉｒｅｌｅｓｓＨＡＲＴ（登録商標）、およびＦＯＵＮＤＡＴＩＯＮ（登録商標）フィールドバスフィールドデバイスなどの、フィールドデバイスで実施される制御モジュールまたはブロックと連携する、異なる制御モジュールを作動させるコントローラアプリケーションを実行する。コントローラ内の制御モジュールは、通信ラインまたはリンクを経由して、フィールドデバイスに制御信号を送信し、それによって、プロセスプラントまたはシステムの少なくとも一部分の動作を制御する。
【０００４】
フィールドデバイスおよびコントローラからの情報は、制御室もしくはより厳しいプラント環境から離れた他の場所に典型的に配置される、オペレータワークステーション、パ

ーソナルコンピュータもしくはコンピューティングデバイス、データヒストリアン、レポートジェネレータ、集中データベース、または他の集中管理コンピューティングデバイスなどの１つ以上の他のハードウェアデバイスに対して、通常、データハイウェイを通じてコントローラから利用可能にされる。これらのハードウェアデバイスの各々は、典型的には、プロセスプラントにわたって、またはプロセスプラントの一部分にわたって集中化される。これらのハードウェアデバイスは、例えば、オペレータが、プロセス制御ルーチンの設定の変更、コントローラもしくはフィールドデバイス内の制御モジュールの動作の修正、プロセスの現在の状態の視認、フィールドデバイスおよびコントローラによって発生された警告の視認、担当者の訓練もしくはプロセス制御ソフトウェアの試験を目的としたプロセスの動作のシミュレーション、構成データベースの保守および更新などの、プロセスの制御および／またはプロセスプラントの動作に関する機能を実施することを可能にし得るアプリケーションを実行する。ハードウェアデバイスによりデータハイウェイを利用して、コントローラおよびフィールドデバイスは、有線通信パス、無線通信パス、または有線もしくは無線通信経路の組み合わせを含むことができる。

【０００５】

一例として、Ｅｍｅｒｓｏｎ　Ｐｒｏｃｅｓｓ　Ｍａｎａｇｅｍｅｎｔによって販売されている、ＤｅｌｔａＶ（商標）制御システムは、プロセスプラント内の多様な場所に配置された異なるデバイス内に記憶され、それらの異なるデバイスによって実行される複数のアプリケーションを含む。１つ以上のワークステーションまたはコンピューティングデバイス内に備わる構成アプリケーションは、ユーザによる、プロセス制御モジュールの作成または変更、およびデータハイウェイを経由した、これらのプロセス制御モジュールの、専用分散型コントローラへのダウンロードを可能にする。典型的には、これらの制御モジュールは、通信可能に相互接続された機能ブロックで構成され、これらの機能ブロックは、それに対する入力に基づき制御スキーム内で機能を実施し、出力を制御スキーム内の他の機能ブロックに提供するオブジェクト指向プログラミングプロトコル内のオブジェクトである。また、構成アプリケーションは、データをオペレータに対して表示するため、かつオペレータによるプロセス制御ルーチン内の設定点などの設定の変更を可能にするために視認アプリケーションが使用するオペレータインターフェースを、構成設計者が作成または変更することを可能にし得る。各専用コントローラ、およびいくつかの場合においては、１つ以上のフィールドデバイスは、実際のプロセス制御機能を実装するために、それらに割り当てられてダウンロードされた制御モジュールを作動させるそれぞれのコントローラアプリケーションを記憶および実行する。視認アプリケーションは、１つ以上のオペレータワークステーション（またはオペレータワークステーションおよびデータハイウェイと通信可能に接続された１つ以上のリモートコンピューティングデバイス）上で実行され得、この視認アプリケーションは、コントローラアプリケーションからデータハイウェイを経由してデータを受信し、ユーザインターフェースを使用してこのデータをプロセス制御システム設計者、オペレータ、またはユーザに表示して、オペレータのビュー、エンジニアのビュー、技師のビュー等のいくつかの異なるビューのうちのいずれかを提供し得る。データヒストリアンアプリケーションが、典型的には、データハイウェイにわたって提供されたデータの一部またはすべてを収集および記憶するデータヒストリアンデバイスに記憶され、それによって実行される一方で、構成データベースアプリケーションが、現在のプロセス制御ルーチン構成およびそれと関連付けられたデータを記憶するために、データハイウェイに取り付けられたなおさらに離れたコンピュータで作動され得る。代替的に、構成データベースは、構成アプリケーションと同じワークステーション内に位置し得る。

【発明の概要】

【発明が解決しようとする課題】

【０００６】

現在知られているプロセス制御プラントとプロセス制御システムのアーキテクチャは、限定されたコントローラおよびデバイスのメモリ、通信帯域幅、ならびにコントローラお

よびデバイスのプロセッサ能力の影響を強く受ける。例えば、現在知られているプロセス制御システムアーキテクチャでは、コントローラでの動的および静的不揮発性メモリの使用は、通常、最小限に抑えられるか、または少なくとも慎重に管理される。その結果、システム構成中（例えば、アプリオリ）に、ユーザは、典型的には、コントローラのどのデータをアーカイブまたは保存するか、保存する頻度、および圧縮を使用するかどうかを選択しなければならず、したがって、コントローラは、この限定されたデータルールセットで構成される。そのため、トラブルシューティングおよびプロセス分析に有用であり得るデータはアーカイブされないことが多く、収集された場合、データ圧縮により有用な情報が失われた可能性がある。

【０００７】

さらに、現在知られているプロセス制御システムでコントローラのメモリ使用量を最小限に抑えるために、（コントローラの構成によって指示されるときに）アーカイブまたは保存されるように選択されたデータが、適切なデータヒストリアンまたはデータサイロでのストレージ用にサイロワークステーションまたはコンピューティングデバイスにレポートされる。データをレポートするために使用される現在の技法は、通信リソースを十分に利用せず、過剰なコントローラの負荷を誘発する。さらに、ヒストリアンまたはサイロでの通信およびサンプリングの時間遅延により、データ収集およびタイムスタンプが実際のプロセスと同期していないことがよくある。

【０００８】

同様に、バッチプロセス制御システムでは、コントローラのメモリ使用量を最小限に抑えるために、コントローラ構成のバッチレシピおよびスナップショットは、典型的には、集中管理コンピューティングデバイスまたは（例えば、データサイロもしくはヒストリアン）の場所に記憶されたままであり、必要な場合にのみコントローラに転送される。かかる戦略は、コントローラに、およびワークステーションまたは集中管理コンピューティングデバイスとコントローラとの間の通信に、著しいバースト負荷をもたらす。

【０００９】

さらに、以前の高いコストのディスクストレージと組み合わせた、現在知られているプロセス制御システムのリレーショナルデータベースの能力および性能の限定は、特定のアプリケーションの目的を満たすために、データを非依存エンティティまたはサイロに構造化する際に大きな役割を果たす。例えば、ＤｅｌｔａＶ（商標）システム内では、プロセスモデル、継続的な履歴データ、ならびにバッチおよびイベントデータのアーカイブは、３つの異なるアプリケーションデータベースまたはデータのサイロに保存される。各サイロは、内部に記憶されたデータにアクセスするための異なるインターフェースを有する。

【００１０】

いずれの場合も、プロセス制御システムなどの産業制御システムの現在のアーキテクチャは、制御機能、入力／出力（Ｉ／Ｏ）機能、ユーザインターフェース機能等のようなさまざまな機能が、ハードウェア（例えば、ユーザワークステーション、プロセスコントローラ、専用Ｉ／Ｏデバイス、フィールドデバイス等）の特定の部分で実施され、それらに結び付けられており、常に特定のハードウェア内にとどまるという点で、主にハードウェアで駆動される。このアーキテクチャ設計は、これらのシステムが迅速に変更または再構成するのが難しく、プロプライエタリ制御ソフトウェアを実装するために使用されるプロプライエタリハードウェアに密接に結び付けられ、実装するのに費用がかかる場合があるデバイスごとのハードウェアの冗長性が必要であり、特定の方法で構成される必要がある追加のハードウェアを追加することなしでは容易にスケーリングすることができないため、制御システムの回復力、応答性、および弾性を限定する。

【課題を解決するための手段】

【００１１】

プロセスプラントで使用するためのプロセス制御などの産業制御システムは、システムの回復力、応答性、および弾性を高めることによってシステムのリアクティブ性を高めるハードウェア／ソフトウェアアーキテクチャを使用する。より具体的には、産業制御シス

テムは、大方の場合、制御システムで使用されるハードウェアを、そのハードウェアで機能するソフトウェアから連結解除し、システムのスケーリング、再構成、および変更を容易にする。概して、新しい産業制御システムまたはプロセス制御システムは、プラント内のフィールドデバイスに連結され、制御およびメッセージングの目的でフィールドデバイスへの直接または間接アクセスを提供する１つ以上の基本機能ノード（ＢＦＮ）入力／出力（Ｉ／Ｏ）デバイスを含む。本システムはまた、１つ以上の高機能ノード（ＡＦＮ）、およびＥｔｈｅｒｎｅｔバスまたはスイッチのようなオープンプロトコルネットワークなどのネットワーク接続を介してＢＦＮ　Ｉ／Ｏデバイスおよびａｆｎに連結された１つ以上のユーザまたは他の計算ノードを含む。

【００１２】

ＢＦＮ　Ｉ／Ｏデバイス（ＢＦＮコントローラとも称される）の各々は、専用Ｉ／Ｏハードウェアに接続された（マルチコアプロセッサであり得る）フロントエンドプロセッサを含むことができ、次に専用Ｉ／Ｏハードウェアは、フィールドデバイス用に現在存在するさまざまなフィールドデバイスプロトコルのいずれかを使用して、プラント内のさまざまなフィールドデバイスへの通信チャネルもしくはバス接続を提供または実装する。高機能ノードは、１つ以上のサーバなどの１つ以上の処理デバイスを含み得、当該処理デバイスは、制御処理、データ分析処理、警告および警報処理、データストレージ等のような高いレベルの処理活動を実施するために、特定のノードで一緒に連結されるか、または一緒に接続される。高機能ノードは、内部に配設され、実行される、仮想コントローラ、仮想Ｅｔｈｅｒｎｅｔデバイス、仮想プロトコルおよび制御中継デバイス、仮想データ分析デバイス等のようなさまざまな仮想デバイスを有し得る。しかしながら、（例えば、高機能ノード内の汎用処理ハードウェア上で作動するソフトウェアで実装され得る）仮想デバイスの各々またはさまざまなグループは、高機能ノード内およびＢＦＮ　Ｉ／Ｏデバイス内の他の仮想デバイスに依存せずに作動するスタンドアロンデバイスとして動作する。さらにまた、本システムは、ユーザワークステーション、データヒストリアン、構成データベース等を実装し得る１つ以上の他のコンピュータノードを含み得る。

【００１３】

ノード（例えば、基本機能ノード、高機能ノード、およびその他の計算ノード）の各々の中の仮想デバイスまたは要素を含むデバイスは、伝送デバイスまたは要素（アクタとも称される）がハードウェアデバイスの場所に依存しないメッセージフォーマット化およびアドレス指定スキームを使用して、他の仮想デバイスまたはアクタにデータメッセージを送信する、すなわち、通信アドレス指定スキームが、メッセージの意図された受信側をホストしているハードウェアデバイスまたはハードウェアデバイスの場所（例えば、サーバ、プロセッサ等）を特定しないか、またはそれらに依存しない、データメッセージングを使用して互いに通信する。例えば、データメッセージを、他の仮想デバイスまたは要素に関連付けられたアドレスに、それらの仮想デバイスのハードウェアの場所またはネットワークの場所を指定することなく、送信することができ、これは、仮想デバイスが、１つのハードウェアの場所から別のハードウェアの場所へ（例えば、特定の高機能ノードにおける異なる処理デバイス間、または異なる高機能ノードの異なるハードウェアデバイス間）、いかなる通信の中断もなく、かつかかる場所の変更を実装するために制御システムを再構成する必要なく、移動することを可能にする。

【００１４】

さらにまた、データメッセージングスキームは、単方向であり得、かつプロトコル不問またはプロトコル非依存であり得、これは、各仮想デバイスまたは要素（共にＢＦＮ　Ｉ／Ｏコントローラおよび高機能ノード内）が任意の所望の通信プロトコルまたはパケット構造と、ネットワーク接続を介して要素とハードウェアデバイスとの間でメッセージを送信するためのメッセージ内の任意の所望のデータフォーマットとを使用することができることを意味する。概して言えば、データメッセージング構造は、このデータメッセージングスキームが、伝送されるデータと、データメッセージ内のデータの構造、および／またはデータメッセージを復号化するために使用するメッセージプロトコルを定義または識別

するデータフォーマット（またはメタデータ）メッセージとを含むデータメッセージを提供するという点で、自己記述型であり得る。データメッセージおよびデータフォーマットメッセージを別々に異なる時間に送信して、メッセージのアドレスが、データメッセージ内のデータを復号化、解釈、および使用するために、データメッセージのフォーマットおよび通信プロトコルを理解することを可能にすることができる。場合によっては、各高機能ノードは、高機能ノード内の仮想コントローラまたは他の仮想要素に送信されたメッセージのデータ解釈および復号化を実施する、コントローラまたは他の中継サービス（仮想マシンまたはデバイスであり得る）を含み得る。

【００１５】

データメッセージングプロトコルのこれらの態様により、仮想要素がハードウェアの場所に依存しないことが可能になり、これは、制御システム要素を再構成する必要なく、制御システム要素を単一のノード（複数のサーバまたはブレードであり得る）または異なるノードのいずれかにおいて、異なるハードウェアデバイス（例えば、異なる処理デバイス）に移動させるか、または配置することができることを意味する。この特徴により、仮想デバイスが作動しているハードウェアデバイスが失敗した場合、過負荷になった場合、または他の理由で必要になった場合に、仮想デバイスを任意の利用可能なハードウェアデバイスに移動させることができるため、この特徴は、より堅牢で安価な冗長性プロビジョニングを可能にする。さらにまた、新しい制御アーキテクチャのデータメッセージングプロトコルのこれらの態様により、同じプロトコル、またはメッセージもしくはデータフォーマットを使用するために制御システム内のすべての仮想デバイスまたは要素を必要とすることなく、仮想デバイスまたは要素が任意の所望のフォーマットまたはメッセージプロトコルのデータメッセージを送信および受信することが可能になる。したがって、例えば、異なる仮想コントローラまたは仮想制御要素は、任意の所望の通信プロトコルおよび／またはデータフォーマットを使用して、共通の通信ネットワークを介して同じノード内または異なるノード内の他の仮想デバイスまたは仮想要素と通信することができる。この特徴により、同じ制御システムまたはネットワーク内の異なる製造業者が提供する仮想デバイスまたは要素の相互運用が可能になる。

【００１６】

さらにまた、制御システムアーキテクチャは、アクタモデルを使用して、システム内で通信を実施することができる。アクタモデルを使用すると、システム内の各制御エントリまたは他の仮想エンティティが、システム内のすべてのアクタとは無関係に動作する別個のアクタになる場合がある。各アクタは、単方向メッセージングを使用して（例えば、本明細書に参照される自己記述型メッセージプロトコルを使用して）、受信アクタが実行されるハードウェアの場所ではなく、受信アクタを指定するアドレスを介して下流のアクタにメッセージを送信するように構成される。その結果、各アクタは、実行および通信時に他のアクタと非同期に動作する自己完結型エンティティである。この機能は、制御システムの残部または制御システム内の他のアクタを再構成することなく、例えば、他のアクタに通知することなく、（負荷バランシング、フォールトトレランス、および冗長性の目的のために）個々のアクタをハードウェア内で動き回らせることができることを意味する。さらに、この方法でアクタモデルを使用することにより、他のアクタを再構成または変更することなく、メッセージストリームまたはスレッド内の一部のアクタで再構成活動を実施することが可能になり、システムを更新することがより容易になり、回復力が高まる。

【００１７】

一実施形態では、産業環境内で物理的機能を実施する複数のフィールドデバイスを有する産業制御システムは、産業環境内の複数のフィールドデバイスに連結された入力／出力プロセッサを含む入力／出力ノードと、１つ以上の仮想コントローラノードであって、１つ以上の仮想コントローラノードの各々が、１つ以上のプロセッサ、メモリ、メモリに記憶され、産業環境内でフィールドデバイスの制御を実施するように１つ以上のプロセッサ上で実行可能な１つ以上の仮想コントローラ、およびメモリに記憶され、ノードの１つ以上の仮想コントローラの動作を管理するように１つ以上のプロセッサ上で実行可能なスー

パーバイザを含む、１つ以上の仮想コントローラノードと、入力／出力ノードを１つ以上の仮想コントローラノードの各々に接続する通信ネットワークと、を含む。入力／出力プロセッサは、産業環境内のフィールドデバイスのうちの１つ以上からデバイス信号を受信し、デバイス信号を処理し、処理されたデバイス信号を、１つ以上の仮想コントローラへの伝達用に通信ネットワーク上に配置し、通信ネットワークを介して、仮想コントローラのうちの１つ以上から制御信号を受信し、受信された制御信号を処理し、処理された制御信号を産業環境内のフィールドデバイスのうちの１つ以上に送信する。

【００１８】

所望の場合には、１つ以上の仮想コントローラノードは、複数の仮想コントローラノードを含み、複数の仮想コントローラノードのうちの１つの１つ以上の仮想コントローラは、仮想コントローラ通信を再構成することなく、複数の仮想コントローラノードのうちの１つから、複数の仮想コントローラノードのうちの別の１つへと移動可能である。別の場合では、１つ以上の仮想コントローラは、仮想コントローラ通信を再構成することなく、仮想コントローラノードのメモリの１つの場所から、仮想コントローラノードのメモリの別の場所へと移動可能である。さらに、仮想コントローラノードのうちの少なくとも１つは、複数の異なるメモリデバイスを含むことができ、１つ以上の仮想コントローラは、仮想コントローラ通信を再構成することなく、仮想コントローラノードのうちの少なくとも１つの複数のメモリデバイスのうちの１つから、仮想コントローラノードのうちの少なくとも１つの複数のメモリデバイスのうちの別の１つへと移動可能であり得る。

【００１９】

同様に、入力／出力ノードは、産業環境内の物理的入力／出力デバイスを介してフィールドデバイスからデバイス信号を受信するように、産業環境内の物理的入力／出力デバイスに連結された仮想入力／出力ルーチンを含み得る。入力／出力ノードは同様にまたは代わりに、産業環境内のフィールドデバイスから直接デバイス信号を受信するように連結された仮想入力／出力ルーチンを含むことができ、仮想入力／出力ルーチンは、フィールドデバイスからのデバイス信号、およびフィールドデバイスに伝達された制御信号に対して入力／出力信号処理を実施するように、入力／出力ノード内の汎用プロセッサで実行される。また、入力／出力ノードは、自己記述型通信スキーム、および／または複数の異なる通信プロトコル、および／またはハードウェアの場所に依存しない通信アドレス指定スキームを使用して、デバイス信号を仮想コントローラノード内の仮想コントローラに通信する仮想入力／出力ルーチンを含み得る。

【００２０】

さらに、入力／出力ノードは、入力／出力ノードのメモリに記憶され、デバイス信号のうちの１つ以上を使用して、産業環境内のフィールドデバイスのうちの１つを制御するための１つ以上の制御信号を生成するように、入力／出力ノードのプロセッサで実行される、制御ルーチンをさらに含み得る。この場合、１つ以上の仮想コントローラノード内の仮想コントローラは、第１のレート以下で制御ルーチン実行を実施することができ、入力／出力ノード内の制御ルーチンは、第１のレートよりも大きい第２のレートで制御ルーチン実行を実施することができる。一実施形態では、第２のレートは、第１のレートの５倍以上であり得る。

【００２１】

同様に、入力／出力ノードは、デバイス信号を通信ネットワーク上で多重化し、複数の異なる通信プロトコルを介して複数のフィールドデバイスからデバイス信号を受信し、自己記述型通信スキームを使用して、デバイス信号を通信ネットワーク上で多重化し、かつ／または複数の異なる通信プロトコルを介して複数のフィールドデバイスからデバイス信号を受信し、複数の異なる通信プロトコルおよび自己記述型通信スキームを使用して、デバイス信号を通信ネットワーク上で多重化する、仮想入力／出力通信ルーチンを含む。自己記述型通信スキームは、第１の通信プロトコルにおけるデバイス信号と、第１の通信プロトコルを記述するプロトコル記述信号と、を含むデバイス信号データを含み得る。

【００２２】

また、所望の場合には、仮想コントローラノードのスーパーバイザは、仮想コントローラノード内のプロセッサの負荷バランシングを実施することができ、仮想コントローラノードのうちの少なくとも１つのスーパーバイザは、仮想コントローラを、仮想コントローラの仮想コントローラ通信を再構成することなく、仮想コントローラノードにおいて１つのプロセッサから別のプロセッサへと移動させるように構成され得る。

【００２３】

別の実施形態では、産業環境内で物理的機能を実施する複数のフィールドデバイスを有する産業制御システムは、１つ以上の通信チャネルを介して産業環境内の複数のフィールドデバイスに連結された入力／出力プロセッサを含む入力／出力ノードと、複数のコントローラノードであって、コントローラノードの各々が、１つ以上のプロセッサ、メモリ、メモリに記憶され、産業環境内でフィールドデバイスの制御を実施するように１つ以上のプロセッサ上で実行可能な１つ以上のコントローラ、およびメモリに記憶され、コントローラノードの１つ以上のコントローラの動作を管理するように１つ以上のプロセッサ上で実行可能なスーパーバイザを含む、複数のコントローラノードと、オープン通信プロトコルを使用して、入力／出力ノードを１つ以上のコントローラノードの各々に接続するオープンプロトコル通信ネットワークと、を含む。ここで、入力／出力プロセッサは、産業環境内のフィールドデバイスのうちの１つ以上からデバイス信号を受信し、１つ以上のコントローラへの伝達用にオープン通信プロトコルを使用して、デバイス信号を通信ネットワーク上に配置し、通信ネットワークを介して、コントローラのうちの１つ以上から制御信号を受信し、制御信号を産業環境内のフィールドデバイスのうちの１つ以上に送信する。

【００２４】

オープンプロトコル通信ネットワークは、Ｅｔｈｅｒｎｅｔ通信ネットワークであり得、かつ／またはＴＣＰ／ＩＰパケットベースのネットワーク通信スキームを使用し得る。また、入力／出力ノードは、オープン通信プロトコルに従って構成されたパケットを介して複数の異なる通信プロトコルでデータを送信することによって、および／またはハードウェアの場所に依存しない通信アドレス指定スキームを使用することによって、デバイス信号をコントローラノードのコントローラに通信し得る。

【００２５】

別の実施形態では、産業環境内で物理的機能を実施する複数のフィールドデバイスを有する産業制御システムは、１つ以上の通信チャネルを介して産業環境内の複数のフィールドデバイスに連結された入力／出力プロセッサを含む入力／出力ノードと、複数のコントローラノードであって、コントローラノードの各々が、１つ以上のプロセッサ、１つ以上のメモリ、１つ以上のメモリに記憶され、産業環境内でフィールドデバイスの制御を実施するように１つ以上のプロセッサ上で実行可能な１つ以上のコントローラ、を含む、複数のコントローラノードと、入力／出力ノードを１つ以上のコントローラノードの各々に接続する通信ネットワークと、を含む。この例では、入力／出力プロセッサは、産業環境内のフィールドデバイスのうちの１つ以上からデバイス信号を受信し、ハードウェアの場所に依存しない通信アドレス指定スキームを使用して、デバイス信号を通信ネットワーク上に配置する。

【図面の簡単な説明】

【００２６】

【図１】複数のＢＦＮ　Ｉ／Ｏコントローラまたはデバイス、複数の高機能ノード、および共通の通信ネットワークを介して通信可能に接続されたユーザワークステーションノードなどの他の計算ノードを有する、産業制御システムまたはプロセス制御システムのアーキテクチャの一例のブロック図である。

【図２】図１のＢＦＮ　Ｉ／Ｏコントローラおよび単一の高機能ノード内の仮想デバイスおよび要素のより詳細な配列例を例示するブロック図である。

【図３】図１および２のシステム内の異なる仮想要素間のデータフローを例示するデータフローダイアグラムである。

【図４】場所およびプロトコルに依存しないメッセージングを実施するように、図１およ

び２の制御システム内のさまざまな仮想要素によって実装され得る自己記述型データメッセージングスキームを例示するフローダイアグラムである。
【図５】本明細書に記載される、アクタベースのモデルおよび自己記述型データメッセージングスキームを使用して、出力に送信される制御信号を生成するＰＩＤ制御要素へのプロセス変数測定の通信を実施するアクタベースの制御ループを実装する方法を例示する、従来の制御ループダイアグラムと並置された例示的なアクタモデルダイアグラムである。
【図６】図１および２の制御システムに実装され得るローカル高速制御ネットワークのダイアグラムであり、単一の高機能ノード内の単一の制御中継器に連結された複数の高速ＢＦＮ　Ｉ／Ｏコントローラを含む。
【図７】図６のローカル高速制御ネットワーク上のＢＦＮ　Ｉ／Ｏコントローラのマルチコアプロセッサ内でタスクを分散させるための例示的なアーキテクチャのブロックダイアグラムである。
【図８】図６のローカル高速制御ネットワークの例示的な構成階層である。
【図９】制御機能を実施するために使用されるプロセス制御システム内の要素の論理的構造のダイアグラムである。
【図１０】Ｓ８８規格に適合する制御ネットワークの一般的なハードウェアプロトコルダイアグラムである。
【図１１】図１および２の制御システムによって使用される自己記述型メッセージプロトコルで使用され得るＯＰＣ－ＵＡプロトコルにデータを適合させるために、図９および１０の論理的ダイアグラムおよびハードウェアダイアグラムで使用するためのデータ変換方法またはスキームを例示するデータダイアグラムである。
【図１２】図１および２の制御システムの高機能ノードに関連付けられた従来のコントローラおよび仮想コントローラを含む制御ノードのセットの例示的な構成階層のユーザインターフェースダイアグラムである。
【図１３】図２の高機能ノードの仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ　Ｍｏｄｂｕｓコントローラに連結され得る例示的なＭｏｄｂｕｓサブネットワークを例示する。
【発明を実施するための形態】
【００２７】
　図１は、プロセスプラントで使用されるプロセス制御システムなどの産業制御システム１０の例示的なアーキテクチャのブロックダイアグラムである。例示的な産業制御システム１０（本明細書ではプロセス制御システム１０とも称される）は、ネットワーク１６を介して１つ以上の高機能ノード１４に連結された１つ以上の基本機能ノード（ＢＦＮ）入力／出力（Ｉ／Ｏ）コントローラを含む。この場合、ネットワーク１６は、スイッチ１８を有するＥｔｈｅｒｎｅｔネットワークとして例示されているが、ネットワーク１６は、任意のＩＰネットワーク（例えば、ＴＣＰ／ＩＰプロトコルを使用する任意のネットワーク）のような任意のパケットベースのネットワークなどの任意のオープンプロトコルネットワークを含む、他のタイプのネットワークであり得る。図１に例示されるように、プロセス制御ネットワーク１０は、ネットワーク１６バスに通信可能に接続されたさまざまな他の計算ノード２０を含む。計算ノード２０は、例えば、オペレータまたはユーザワークステーションノード（２２、２４）、構成ノード２６、１つ以上のデータヒストリアンまたはデータベースノード２８、およびプロセス制御プラントまたはシステムで使用される典型的なハードウェアデバイスを実装する任意の他のノードを含み得る。
【００２８】
　概して言えば、ＢＦＮ　Ｉ／Ｏコントローラ１２の各々は、プロセス制御システム１０、および特に、高機能ノード１４を、センサ、バルブ、ならびに他の測定および制御デバイスなどの制御されるプラント内のさまざまなフィールドデバイスに接続するコンピューティングプラットフォームである。ＢＦＮ　Ｉ／Ｏコントローラ１２の各々は、ＨＡＲＴ、Ｆｏｕｎｄａｔｉｏｎ　Ｆｉｅｌｄｂｕｓ、ＣＡＮ、Ｐｒｏｆｉｂｕｓ、ＷｉｒｅｌｅｓｓＨＡＲＴ等のＩ／Ｏデバイスなどの Ｉ／Ｏデバイスを含む従来のＩ／Ｏサブネットワークであり得る１つ以上のＩ／Ｏサブネットワークまたはデバイスに連結されたフロント

エンドプロセッサまたは高速コントローラを含む。当然のことながら、Ｉ／Ｏデバイスは、データの収集および制御の目的で、すなわち、フィールドデバイスからデバイス信号を受信し、それらのデバイス信号を処理、例えば、信号を解釈するなどして、信号に対してデータプロトコル変換またはプロトコルトンネリングを実施するために、プラント内のさまざまなフィールドデバイスに接続するＩ／Ｏチャネルまたはバスに連結される。さらにまた、ＢＦＮ　Ｉ／Ｏコントローラ１２は、ＨＡＲＴ、ＷｉｒｅｌｅｓｓＨＡＲＴ、Ｆｏｕｎｄａｔｉｏｎ　Ｆｉｅｌｄｂｕｓ、４～２０ミリアンペア、ＣＡＮ、Ｐｒｏｆｉｂｕｓ、またはその他の既知のプロトコルのいずれかなど、任意の所望の通信プロトコル構造を使用して、フィールドデバイスに直接連結され得る。ＢＦＮ　Ｉ／Ｏコントローラ１２は、制御されるプラントまたはシステム内のさまざまなハードウェア、例えば、フィールドデバイスへの通信接続（論理的に構成可能な接続であり得る）を提供する物理的構成要素を含み、かつネットワーク１６とサブネットワークとの間で通信活動を実施することができ、場合によっては、高速制御活動などの制御活動を実装することができる論理的構成要素を含む。しかしながら、特定のＢＦＮ　Ｉ／Ｏコントローラ１２の物理的構成要素および論理的構成要素は、概して、ネットワーク１６の観点から単一のノードを形成するように、一緒に密接に連結される。一般的な意味で、各ＢＦＮ　Ｉ／Ｏコントローラ１２は、本質的にプロプライエタリであり得、異なる製造業者によって提供され得るが、これらのデバイス１２は、以下に記載される方法で通信を実施して、さまざまな異なるタイプのＩ／Ｏデバイスへの（プロプライエタリデバイスへさえも）ネットワーク化された通信を可能にする一方、依然として、同じプロセス制御ネットワーク内のＢＦＮ　Ｉ／Ｏコントローラ１２の相互運用を可能にするであろう。同様に、理解されるように、ノード１２の入力／出力デバイスは、１つ以上のコントローラ、例えば、以下に記載される仮想コントローラから制御信号を受信することができ、（例えば、これらの制御信号を、フィールドデバイスに送信され得る信号フォーマットに変換またはパッケージ化するように）これらの制御信号を処理することができ、処理された制御信号を、入力／出力ラインを介してフィールドデバイスに提供して、フィールドデバイス、ひいては、プラントの制御を実施することができる。

【００２９】

高機能ノード１４は、各ノード１４のサーバファーム内の１つ以上のサーバなど、汎用コンピューティング処理ハードウェアまたは処理ハードウェアバンクを含み得る。概して言えば、高機能ノード１４は、データ分析、構成機能、メンテナンス機能、データストレージ機能等のような制御および他の機能を実施するために、ネットワーク１６、スーパーバイザ、およびノードのハードウェアで実行中のさまざまな仮想デバイスまたはサービスへの通信インターフェースを有する。スーパーバイザ（本明細書ではハイパーバイザとも称される）は、高機能ノード１４内で動作中の、さまざまな仮想コントローラ、仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ、データ中継サービス、通信サービス等を含む、さまざまな他の仮想デバイスまたは論理的デバイスの動作と連携することができる。スーパーバイザは、特定の高機能ノード１２内で負荷バランシングおよび冗長性サービスを提供するために、高機能ノード１２のハードウェア内の他の論理的デバイスまたは仮想デバイスを移動させるか、または能動的にスケジュールを組むこともできる。

【００３０】

さらにまた、さまざまな他の計算ノード２０は、ネットワーク１６と、特に、ＢＦＮ　Ｉ／Ｏコントローラ１２および高機能ノード１４とインターフェースして、ユーザインターフェース活動、ネットワークおよびプロセス制御システム構成活動、データの収集およびストレージ活動、データ分析活動等のような他の制御活動を実施することができる。しかしながら、多くの場合、他の計算ノード２０は、シンクライアントデバイスを含むか、またはシンクライアントデバイスを作動させることができ、シンクライアントデバイスは、他のノード内、および特に、高機能ノード１４内の仮想デバイスまたはエンティティとインターフェースして、ユーザに情報を提供し、ユーザが、高機能ノード１４の仮想デバイス内で実行される制御機能、メンテナンス機能、データ分析機能、警告機能、および他

の機能とインターフェースすることを可能にする。
【００３１】
図２は、ネットワーク１６のスイッチ１８を介して接続されたＢＦＮ　Ｉ／Ｏコントローラ１２のうちの１つ、および高機能ノード１４のうちの１つをより詳細に例示する。図２に例示されるように、ＢＦＮ　Ｉ／Ｏコントローラ１２は、論理的構成要素３０（ＢＦＮ Ｉ／Ｏコントローラ１２内のプロセッサ上で実行されるソフトウェアルーチンおよびモジュールを含む）と、１つ以上のプロセッサ３４（汎用プロセッサ、ＡＣＩＣ、プログラム可能な論理アレイ、再プログラム可能なゲートアレイ、プログラム可能な論理コントローラ（ＰＬＣ）等であり得る）、およびＢＦＮ　Ｉ／Ｏコントローラ１２を、制御されているプラントまたはシステム内のフィールドデバイスまたはその他のデバイスに接続する際に使用するためのさまざまな物理的Ｉ／Ｏコネクタ３６を含む、物理的構成要素３２とを含む。物理的Ｉ／Ｏコネクタ３６は、４～２０ｍａ規格のコネクタ、Ｆｉｅｌｄｂｕｓ、ＣＡＮ、Ｐｒｏｆｉｂｕｓ等のコネクタなのバスベースのコネクタ、ＨＡＲＴコネクタなどのチャネルコネクタ、ＷｉｒｅｌｅｓｓＨＡＲＴコネクタ、ＩＥＥＥ無線プロトコルコネクタなどの無線コネクタ等を含む、任意の所望のタイプのＩ／Ｏコネクタであり得る。ある場合では、Ｉ／Ｏコネクタハードウェア３６は、接続が行われた後に構成または指定されるプラント内のフィールドデバイスおよび他のハードウェアのアドレス指定またはタグ接続を可能にする、再プログラム可能なコネクタまたはデジタル構成可能なコネクタを含み得る。かかる構成可能な接続Ｉ／Ｏ構造の例は、米国特許番号第７，６８４，８７５号、同第８，３３２，５６７号、同第８，７６２，６１８号、同第８，９７７，８５１号、同第９，０８３，５４８号、同第９，４１１，７６９号、および同第９，４９５，３１３号、ならびに米国特許出願公開第２０１６／０２２６１６２号に記載されており、これらは、参照により本明細書に明示的に組み込まれ、本明細書ではＣＨＡＲＭ　Ｉ／Ｏと称される。
【００３２】
同様に、図２に例示されるように、Ｉ／Ｏコントローラ１２の論理的構成要素３０は、ネットワーク１６を介して通信を実施する通信アプリケーション（Ｃｏｍｍ）、Ｉ／Ｏコントローラ１２に記憶された制御モジュールを実装または実行して、例えば、ＢＦＮ　Ｉ／Ｏコントローラ１２内での高速制御ダウンを実施することができる制御アプリケーション、および物理的Ｉ／Ｏ構成要素３２または３６を介してサブネットワーク（例えば、フィールドデバイス）との通信を実施するＩ／Ｏアプリケーションなどのさまざまなサービスアプリケーションを含み得る。図２にも例示されるように、論理的構成要素３０は、例えば、入力制御モジュール（例えば、機能ブロック）、出力制御モジュール（例えば、機能ブロック）、比例－積分－微分（ＰＩＤ）もしくはモデル予測制御（ＭＰＣ）ブロックなどの制御計算モジュールもしくはブロック、警告ブロック等であり得る、１つ以上の制御モジュール３８を含むか、または記憶し得る。概して、ＢＦＮ　Ｉ／Ｏコントローラ１２は、その中の制御モジュールまたはブロックを動作または実行して、高速制御（例えば、１０ミリ秒のサイクル時間）を実施する。
【００３３】
図２に例示されるように、高機能ノード１４は、サーバファーム内のサーバ、別個のマザーボード上のプロセッサおよびメモリ等のような複数の異なるハードウェアデバイスを含むことができ、各サーバまたはプロセッサ構成要素は、例えば、サーバキャビネット内の異なるブレードまたはハードウェアデバイスとして構成される。高機能ノード１４は、ノードでのハードウェア監視を提供するだけでなく、さまざまなハードウェアデバイスで構成または実行される、ノード１４内のさまざまな他の仮想デバイスまたはエンティティの動作を制御および指揮するスーパーバイザまたはハイパーバイザ５０を含む。高機能ノード１４は、例えば、ノード１４の１つ以上のプロセッサ上でベースオペレーティングシステム（ＯＳ）として作動するＷｉｎｄｏｗｓ（登録商標）ＯＳなどの１つ以上のオペレーティングシステムを含み得る。同様に、高機能ノード１４は、任意の数の仮想マシン、デバイス、またはエンティティを含み得る。例えば、図２に例示されるように、高機能ノ

ード１４は、制御活動を実施するために、プロセスプラント内の従来のコントローラとしてエミュレートまたは動作することができる１つ以上の仮想コントローラ５２を含む。仮想コントローラ５２の各々は、標準の分散型コントローラでは典型的であるように、さまざまな制御モジュール、機能ブロック、警告モジュール、通信モジュール、ユーザインターフェースモジュール等を含むことができ、これらの制御モジュールまたは仮想コントローラ５２内の他の構造は、フィールドデバイスの処理およびフィールドデバイスへの伝達のために、通信ネットワーク１６を介してノード１２に送信される（任意の種類の）制御信号を発生させることができる。しかしながら、この場合、仮想コントローラ５２は、相互に依存せず、かつ高機能ノード１４内の他の仮想デバイスまたはエンティティに依存せずに作動し、ノード１４内の異なるプロセッサまたはサーバハードウェアデバイスで作動または実行することができ、制御を実施するため、例えば、センサ測定値を取得するため、制御信号を発生させてフィールドデバイスに送信するため、フィールドデバイスまたはオペレータインターフェース２２、２４からの警告、警報、およびその他のメッセージを発生させて送信または処理するため、指揮制御を実施する等のために、ネットワーク１６を介してＢＦＮ　Ｉ／Ｏコントローラ１２のうちの１つ以上に依存せずに作動し、それらと通信する。

【００３４】

同様に、図２に例示されるように、高機能ノード１４は、制御を実施することができるか、またはＥｔｈｅｒｎｅｔネットワーク接続１６を介してノード１４に連結された他の分散型制御サブネットワークの制御活動を追跡することができる、１つ以上の仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮコントローラまたはデバイス５４を含み得る。かかるサブネットワークとしては、例えば、Ｍｏｄｂｕｓネットワークを挙げることができる。図２に例示されるように、仮想コントローラ５２およびＥｔｈｅｒｎｅｔ　ＢＦＮコントローラ５４は、同じまたは別個のまたは異なるオペレーティングシステムで作動または実行することができる。

【００３５】

さらにまた、図２に例示されるように、高機能ノード１４は、仮想コントローラ５２、５０およびＢＦＮ　Ｉ／Ｏントローラ１２、ならびにそれらが取り付けられているサブネットワークからのプロセスおよび／または他のデータを収集、編成、および記憶し得る、もう１つのデータストレージ仮想マシン６２を含み得る。所望の場合には、データストレージ仮想マシン６２は、仮想ビッグデータアプライアンスまたは任意の他のタイプのデータストレージマシンとすることができる。さらに、高機能ノード１４は、例えば、仮想データストレージデバイス６２、仮想コントローラ５２、５０、ＢＦＮ　Ｉ／Ｏコントローラ１２等によって収集されたデータまたは任意の他のデータを使用して、データ分析を実施し得る、１つ以上の仮想データ分析デバイス６４を含み得る。繰り返しになるが、仮想マシンの各々は、任意の汎用プロセッサおよび異なるハードウェアデバイス（例えば、サーバまたはマザーボード）のいずれかに関連付けられたメモリ上など、ノード１６内の同じまたは異なる汎用処理ハードウェア上に実装することができる。

【００３６】

さらにまた、高機能ノード１４は、ネットワーク１６内のデータに対してＯＰＣ処理および変換を実施して、異なるソースからのデータまたは異なる製造業者からのデバイスの相互運用または解釈を可能にする仮想ＯＰＣ－ＵＡデバイス６８を含み得る。高機能ノード１４はまた、１つ以上の仮想構成アプリケーション、仮想オペレータインターフェースアプリケーション、仮想制御インターフェースアプリケーション、仮想メンテナンスインターフェースアプリケーション等７０を含み得る。仮想デバイス７０は、標準のオペレータ制御活動（ユーザインターフェースを含む）、構成活動（ユーザインターフェースを含む）、メンテナンス活動（ユーザインターフェースを含む）、警告処理および表示活動等を実装することができ、これらの仮想デバイス７０は、プラントまたはシステム１０内のユーザとインターフェースするために、ネットワーク１６を介してユーザワークステーションまたはシンクライアント２２、２４、２６等とインターフェースすることができる。

【００３７】
さらにまた、高機能ノード１４は、ＢＦＮコントローラ中継サービスまたはマシン７４を含み得る。概して言えば、コントローラ中継サービスまたはマシン７４は、例えば、高機能ノード１４のＢＦＮ　Ｉ／Ｏコントローラ１２と仮想コントローラ５２との間を流れるデータのデータ中継サービスを実施する。以下により詳細に論じられるように、これらのデータ中継サービス７０により、ネットワーク１６を介した通信がプロトコルに依存しないことが可能になる、すなわち、中継サービス７０は、制御システム１０内の異なるデバイス、仮想デバイス、論理的エンティティ、および仮想エンティティが所望の任意の通信プロトコルと任意のデータフォーマットとを使用し、依然としてネットワーク１６を介して通信を実施することを可能にする。この特徴はまた、制御システム１０が、ＢＦＮ　Ｉ／Ｏコントローラ１２と高機能ノード１４との間の通信を実施する際に使用する通信プロトコルおよびデータフォーマットの選択という意味でオープンアーキテクチャであることを可能にする。
【００３８】
理解されるように、ノード１４内の仮想マシンまたはデバイスの各々は、それら自体の（および異なる）オペレーティングシステムを有することができ、ひいては、他の仮想デバイスに依存せずに作動することができる。さらに、高機能ノード１４内のさまざまな仮想マシンの設計および構成は、互いに連携する必要がなく、ひいては、異なる製造業者によって提供され、ノード１４の設計をオープンアーキテクチャにすることができる。さらにまた、単一の高機能ノード１４内の仮想デバイスまたはエンティティの各々は、ノード１４内の同じまたは異なる物理的サーバまたは処理デバイス上で作動または実行することができ、他の仮想デバイスまたはノード１４自体を再構成することを必要とすることなく、かつ移動中の仮想デバイスのダウンタイムを限定または最小限に抑えるだけで、（例えば、ハイパーバイザ５０によって）ノード１４内の物理的ハードウェアまたは処理デバイス間で移動させることができる。この機能は、ハイパーバイザ５０が、ノード１４において物理的ハードウェアまたはサーバ間で仮想デバイスを移動させて、ノード１４で負荷のバランスをとり、ハードウェアデバイスのうちの１つが失敗した場合に冗長性の再構成または切り替えを実施し、ノード１４で他の仮想デバイスの動作等に大きな影響を与えることなく、特定の仮想デバイスによって一時的な高負荷処理を可能にする。
【００３９】
したがって、図１および２の新しいオープンアーキテクチャの産業制御システムは、１つ以上の基本機能ノード（ＢＦＮ）入力／出力（Ｉ／Ｏ）コントローラまたはデバイス１２、および１つ以上の制御スーパーバイザまたは高機能ノード１４がネットワーク１６を介して連結され、各々がＩ／Ｏサービス、デバイスサービス、制御サービス、および共通のサービスを含む一連の機能をサポートし得る、分散型アーキテクチャを含む。システム１０は、ハードウェアとソフトウェアとの組み合わせを含み、ＢＦＮ　Ｉ／Ｏコントローラ１２のハードウェアは、構成可能なスマート論理ソルバ（ＣＳＬＳ）、構成可能なＩ／Ｏキャリア、さまざまな構成可能なＩ／Ｏモジュールおよびサポートケーブル、ならびに電源を含む。システム１０はまた、オープンシステム（例えば、ＯＰＣ－ＵＡ）インターフェース６８、コントローラ中継器層７０、および他の特徴を提供するために使用される指揮コントローラ１４を含む。システム１０の指揮コントローラノード１４および他の部分は、構成可能な仮想システムの一部として仮想化およびパッケージ化され得るか、またはスタンドアロンハードウェアとしてインストールされ、作動することができる。
【００４０】
基本機能ノード１２を、最も基本的な形で、単一のデバイスで、または単一の測定でも使用することができ、その結果、すべての場合で基本機能ノード１２がＩ／Ｏを多重化する必要がないことが理解されよう。さらに、基本機能ノード１２はまた、他のデバイス、例えば、Ｅｔｈｅｒｎｅｔ　ＩＰデバイスがオープンプロトコルをサポートしていない場合、それらのデバイスに中継サービスを提供することができる。さらなる例として、基本機能ノード１２は、重量スケール、ＮＩＲデバイス、ゲージングシステム等のような他の

デバイスに中継サービスを提供することができる。
【００４１】
さらにまた、基本機能ノード１２を介して接続する代わりに、本明細書に記載される制御システムアーキテクチャを拡張して、Ｉ／Ｏデバイス、または実際のプロセス制御デバイスもしくはフィールドデバイスなどの他のデバイスをネットワーク１６に直接接続させることができる。この場合、デバイス（例えば、フィールドデバイス）は、ネットワークプロトコルとの通信をサポートし、通信、データ操作、およびバルブ診断などの特定の機能のために使用され得る（以下に記載されるアクタであり得る）ルーチンのうちの１つ以上、ただし、場合によってはごく少数を含み得る。
【００４２】
図１および２の制御アーキテクチャは、アプリケーションが、プラント制御システム１０内のネットワーク１６の任意の計算ノードまたはプロセッサノードにインストール可能であることを可能にするように設計されている。これらのアプリケーションは、システムの負荷要求および外乱に応答して、ノード間で分散されたり、場合によっては計算ノード間で移動されたりする場合がある。アプリケーションを依存せずに移行およびアップグレードする能力により、システム１０全体で非常に重要なスケーリングとアップタイム統計を達成することが可能になる。この能力は、リアクティブと称される場合があり、Ｅｒｌａｎｇ、ならびに最近ではＡｋｋａおよびＡｋｋａ．Ｎｅｔを含むさまざまなリアクティブアーキテクチャ上に構築される場合がある。
【００４３】
新しいオープンアーキテクチャはまた、自己記述型のデータ駆動型システムまたはデータメッセージングの使用を含み得る。自己記述型メッセージングアプローチでは、データおよびデータ記述の両方をシステム１０内のあらゆる構成要素（物理的および／もしくは論理的または仮想構成要素）の一部として記憶することができ、データおよびデータ記述の両方は、任意のデータ通信のエンドポイント間の接続の確立の一部として通信され得る。
【００４４】
新しいオープンアーキテクチャはまた、ハードウェアレベル、アプリケーションレベル、および特徴レベルでの冗長性を含む。ハードウェアレベルでは、データをシステム間で複製することができ、ハードウェア障害が検出されると、タイミングまたはデータをまったく、またはほとんど失うことなく、アプリケーションまたは仮想マシン全体を（例えば、ハイパーバイザ５０によって開始される）他のハードウェアプラットフォームに自動的に切り替えることができる。アプリケーションレベルでは、アプリケーションを、異なる計算ノードまたは（これもハイパーバイザ５０によって開始される）ノード内の異なるサーバもしくは物理的処理マシン上で複製することができ、接続を失うことなくデバイスまたはノード間で移行することができる。特徴レベルでは、測定値などの特徴を他の測定値で、またはより一般的な場合ではソフトセンサでバックアップすることができる。システム１０はまた、制御システムの基本アーキテクチャの一部として測定調和を実施し得る。
【００４５】
さらにまた、新しいオープンアーキテクチャは、データ駆動型インターフェースを含み得る。データ駆動型インターフェースを用いると、エンドポイントは、実装例の詳細を知らなくても、関係するデータについて構成要素を照会することができる。これをさらに一歩進めると、照会駆動型機能スタイルのインターフェースを使用して、物理的に分離されたサブシステムにわたってデータにアクセスすることができる。同様に、上記のように、この制御システムアーキテクチャでは、ハードウェアコンピューティングリソースをクラスタ化および仮想化して、非常にスケーラブルなハードウェアプラットフォームを提供することができ、これらの仮想化システムは、Ｄｏｃｋｅｒなどの技術を使用して、アプレットをインストールおよび実行することができる。
【００４６】
さらに、オープンシステムアーキテクチャは、アプリケーションが、システム１０内の任意の計算ノードにインストール可能であることを可能にするように設計される。したが

って、これらのアプリケーションは、ノード１２と１４との間で分散され得るか、または場合によっては、例えば、システムの負荷および障害に応答して、ノード１２と１４との間で移動され得る。アプリケーションを依存せずに容易に移行およびアップグレードする能力により、システム１０全体で非常に重要なスケーリングと利用可能な統計を達成することが可能になる。

【００４７】

背景として、分散型制御システム（ＤＣＳ）産業は、マイクロコントローラの台頭によって大部分が可能になった。初期のＤＣＳアーキテクチャの一部として、オペレータ、コントローラ、およびＩ／Ｏ機能は、専用の計算プラットフォームに分離され、プロプライエタリな冗長性ネットワーク技術を介して分散されていた。Ｅｍｅｒｓｏｎ　Ａｕｔｏｍａｔｉｏｎ　ＳｏｌｕｔｉｏｎｓのＤｅｌｔａＶ（商標）システムなどの次世代システムは、これらの概念を、制御機能がほぼすべてのノードで作動することを可能にすること、および計算能力がシステム内の任意のポイントで利用されることを可能にすることによっての両方に拡張しており、最近では、ＣＨＡＲＭＳ（商標）技術の点でＩ／Ｏ自体に拡張した。ＤｅｌｔａＶは、Ｅｔｈｅｒｎｅｔ、スイッチ、ＴＣＰ／ＵＤＰ／ＩＰ、ならびにＷｉｎｄｏｗｓおよびＬｉｎｕｘ（登録商標）などの一般的なオペレーティングシステムなどのＩＴベースの技術も採用している。ＤｅｌｔａＶなどのシステムは、制御を任意の場所で作動させることができるが、アーキテクチャは依然として、機能が特定の計算ボックスまたはマシンで作動されることを強要する。

【００４８】

さらに、非常に高いアップタイムを達成し、障害シナリオをサポートするため、およびオンラインアップグレードシナリオを提供するために、制御アーキテクチャは、典型的には、ネットワーク冗長性、コントローラ冗長性、Ｉ／Ｏ冗長性、およびアプリケーションステーションの冗長性をすべてハードウェア固有ベースで提供する。この冗長性はその目標をほぼ満たしているが、制御アプリケーションのスケールアップを困難にする。特に、制御アプリケーションは、手動で再割り当てし、システムにインストールまたは「ダウンロード」しなければならない。さらに、リモート監視、障害検出、およびその他のアプリケーションの台頭により、これらのＤＣＳアーキテクチャの負荷がさらに増加した。これらの要件の多くは、いわゆるＩｏＴまたはＩＩｏＴ要件によって駆動される。

【００４９】

本明細書に記載されるシステムで使用されるとき、これらの問題の一部に対する答えは、一部には、マルチコアプロセッサの使用、タイムセンシティブネットワーク（ＴＳＮ）、ＦＰＧＡなどプログラム可能なプラットフォーム、データ駆動型技法、およびメッセージ駆動型システムなどの新興のネットワークメッセージングパッシング技術の使用である。マルチコアプラットフォームを使用する１つの方法は、仮想化、および最近ではＤｏｃｋｅｒなどのコンテナベースの仮想化を用いてきた。仮想化およびコンテナベースの仮想化は、多くの産業で採用されている。しかしながら、図１および２のオープンアーキテクチャ制御システムで使用され得る１つの重要なアプローチは、アクタベースの方法（これは、コンピュータサイエンス産業では一般的な用語である）を利用する。このアクタベースのアプローチにより、システム１０のリアクティブ性が高まり、リアクティブアプリケーション開発は、大きなパラダイムシフトである。

【００５０】

概して、リアクティブシステムの原則は、特にメッセージ処理の使用によって、応答性、回復力、および弾性がより高いシステムを達成するように設計される。より具体的には、応答性システムは、計算の必要性、障害、およびユーザ要求の変化に可能な限り迅速に応答することができ、これにより、アプリケーションおよびユーザが動作の完了を無駄に待つのに多大な時間を費やさないことを確保する。例えば、アプリケーションが応答するためには、アプリケーションが使用量のスパイク増加に反応することができる必要がある。これらのスパイクは、何らかの形態の異常検出を実施するためのユーザ要求によって、またはユーザがシステム全体からデータを表示する一連のダッシュボード（ユーザインタ

ーフェース）を引き上げるのと同じくらい単純または一般的なものによって駆動され得る。アプリケーションの使用量が２倍になった場合、アプリケーションの応答時間も２倍になってはいけない。応答性を管理する最も容易な方法は、この増加した処理圧力に反応する容易なスケーラビリティを可能にするシステムを作成することである。アプリケーションの負荷が増加した場合にプロセッサリソースを過負荷にすることなく、サービスのインスタンスの数を増減できるようにアプリケーション処理を提供することにより、アプリケーションがこれと処理要求の増加に容易に反応することを可能にする。本明細書に記載されるシステムは、システム１０で使用される仮想化およびメッセージパッシングスキームに基づいて、この容易なスケールアップを可能にする。

【００５１】

さらに、アプリケーションを使用可能にするためには、アプリケーションは、供され得る広範な条件に耐える必要があり、そのため、回復力があるべきである。アプリケーションは、負荷の増加によるユーザに起因する問題、ならびにアプリケーション内で発生した問題に対して回復力があるべきである。例えば、とりわけ、より強力なアプリケーションを作成するために組み合わされている非常に多くのデータソースにより、ほとんどすべてのアプリケーションでエラーは一般的な態様である。これらの外部サービスのいずれかが失敗する可能性は常に存在し、その結果、アプリケーションでエラーに遭遇したときに何が起こるかを考慮することが重要である。回復力のあるアプリケーションを確保するために、アプリケーションは、障害源に応答することができるか、またはそれに耐えられなければならない。

【００５２】

同様に、許可の欠如などの潜在的なセキュリティ上の問題が発生した場合に、システムがいかに対処するかを考慮することも重要である。ユーザがユーザ自身のジョブをますます多くのサービスと統合することを検討するにつれて、システムアーキテクチャは、これらのサードパーティシステムの障害を取り扱うことができることが必要である。概して言えば、障害が発生した場合に、対処するためにシステム全体が苦戦しないこと、またはさらに悪いことに、単一の構成要素の障害の結果としてシステム全体が失敗することに苦戦しないことが重要である。理想的には、障害を可能な限り最小のユニットまで下位へ隔離し、そのユニットだけに障害をカプセル化することが望ましい。この設計の目標は、アプリケーションが依然として逆境に立ち向かうことができることを確保するのに役立つ。障害を考慮すると、ユーザに対して障害の内部詳細を強要することは望ましくないが、代わりに、ユーザにとって意味のある一部の詳細をユーザに提供することが望ましい。理想的なシナリオでは、システム１０は、課題を自動的に解決し、アプリケーションに自己修復させることができるべきである。したがって、スケーラビリティおよび回復力の両方が強力な接続を共有し、アプリケーションがスケーラビリティを念頭に置いて構築されている場合、アプリケーションは障害をよりよく取り扱うことができるであろう。アプリケーションが回復力を念頭に置いて構築されている場合、アプリケーションは増加した負荷を取り扱うことができるという点で、この記述の逆も真である。

【００５３】

リアクティブシステム、および特に、図１および２の制御システム内のリアクティブアプリケーションを提供するために、システム１０は、システム内のアプリケーションまたはエンティティ間の単方向メッセージング、およびメッセージングのための自己記述型データプロトコルの使用に依存する特定のメッセージパッシングプロトコルまたはアーキテクチャを使用することができる。メッセージパッシングアーキテクチャは、非同期通信の形態であり得、ここで、データは、ワーカーがファイアアンドフォーゲット方法で後のステージで処理するためにキューに入れられる。かかる単方向メッセージパッシングアーキテクチャを使用することにより、回復力があり、スケーラブルで応答性のアプリケーションを作成するためのいくつかの貴重な構築ブロックを提供する。特に、単方向メッセージパッシングアーキテクチャの使用は、構成要素（アプリケーション、デバイス等）間の隔離を提供する。隔離が達成されると、利用可能なＣＰＵ電力、利用可能なメモリ、障害の

リスク等のような要因に応じて、最も理想的な（ハードウェア）場所で実施する必要がある異なるタスクを展開することが可能である。隔離は、単一の構成要素が失敗した場合、システムの残部が同様に失敗することなく、動作することが可能であるため、回復力の重要な構成要素である。隔離はまた、失敗した構成要素が、依存サービスからバックオフして課題を引き起こすか、またはその状態をリセットするために再起動するかのいずれかによって、時間の経過とともに自己修復する機会を得ることを可能にする。

【００５４】

さらに、単方向メッセージパッシングスキームを使用することにより、場所の透過性を提供する。より具体的には、単方向メッセージパッシングは、送信側がメッセージを送信するために受信側のアドレスを提供することのみを必要とする。送信側は、受信側の（ハードウェア内の）実際の場所を知る必要がなく、かつそれにより、受信側は、制御システム内の任意の場所に位置することができ、送信側が知らなくても移動することができる。特に、送信側が受信側のアドレスでメッセージを送信する場合、フレームワークは、そのサービスが同じマシン上、同じノード内のサーバ上、またはまったく異なる計算プラットフォーム内のサーバ上に位置するかどうかにかかわらず、クラスタまたはノードのセット内のサービスまたは受信側の物理的な場所にメッセージをルーティングする複雑さを扱う。この特徴により、フレームワークは、変更についてアプリケーションに警報する必要なしに、作動中のタスクの場所を動的に再構成することができ、より容易なスケーリングが可能になる。システム障害を考慮すると、フレームワークがサービスを新しい場所に再展開することは完全に実行可能であり、これも場所の透過性の要件につながる。

【００５５】

メッセージパッシングを、アクタベースの同時並行性の使用によって有利に扱うこともできる。この方法論の例としては、ＥｒｌａｎｇおよびＡｋｋａが挙げられる。これらの技法は両方とも、システムを非依存アクタに分割するアクタモデルを実装する。アクタモデル内のアクタは、３つの重要な概念であるストレージ、処理、および通信をカプセル化する。アクタは、非ブロッキングで非同期である。

【００５６】

ＤＣＳは、データ集約型アーキテクチャである。その結果、測定デバイスから非常に大量のリアルタイムデータが抽出され、このデータを使用して、制御、ユーザ、およびエンタープライズレベルの決定を駆動する。リアクティブアプリケーションは、周囲の変化する世界に反応する。したがって、構成要素または作業ユニットは、初期の処理チェーンで他の構成要素によって伝播されたメッセージに反応する。この単方向のアクタベースのメッセージングの結果として、図３に示されるように、データはアプリケーションのステージを通って流れる。特に、図３は、単純なデータフローダイアグラム１００の一例を提供し、ここで、複数のデータコレクタ１０２は、データをデータアグリゲータ１０４に提供（またはファンイン）し、次いで、データアグリゲータ１０４は、データを別のデータアグリゲータ１０６に送信することができ、次いで、データアグリゲータ１０６は、データを複数の出力ブロック１０８に送信することができる。図３のデータコレクタ１０２は、例えば、センサ測定値であるか、または従来の制御システムのＡＩブロックに関連している場合がある。データアグリゲータ１０４は、従来の制御システムの制御ブロックまたは制御モジュールであり得、データアグリゲータ１０６は、従来の制御システムの警告取り扱いブロックであり得、出力ブロック１０８は、従来の制御システムユーザインターフェース、警告ステーション、警告インターロック等であり得る。当然のことながら、図３のデータフローダイアグラム１００は、産業制御システムまたはプロセス制御システム内の多くの他のタイプのデータフローを表すことができる。図３の矢印は、データがどのようにシステムを通って流れるかを例示しており、循環依存関係（すなわち、データメッセージングに関する単方向フロー）の欠如は、かかるシステムがどのように動作するかを理解するために必要なオーバーヘッドを低減することがすぐに明らかになる。

【００５７】

図３の例では、システム１０は、（例えば、パブリッシュ／サブスクライブ通信プロト

コル、ＨＴＴＰ　ＡＰＩ要求、またはＭＱＴＴなどの他の何らかの形態の遠隔取り込みプロトコルなどの制御システムプロトコルを介して）データを制御システムに取り込む。データがセンサデータである場合、センサ識別子構成要素（例えば、データアグリゲータ１０４または１０６のうちの１つ）は、センサ読み取り値に対してアグリゲーションおよびその他の処理を実施することを担い、次いで、これらの読み取り値を次の構成要素に伝播する。この次の構成要素は、ビュークライアントであり得、次いで、ビュークライアントは、データを、データを表示するディスプレイ、データを記録する履歴クライアント、ならびにデータおよび／または他のアプリケーションに対してアクションをとる制御アプリケーションに提供する。データフローを考慮する際の主要な目的は、動作がチェーン内で逆方向に呼び出されることに依存することを避けるべきであることである。代わりに、各サービスは、チェーンの前の要素（複数可）から受信したデータに応じて正しく作用することができるべきである。この単方向のデータフローにより、構成要素の試験が容易であること、コードベースを初めて使用する開発者にとって組み合わせが理解しやすいこと、およびまた、アプリケーションがプルベースモデルではなくプッシュベースモデルで作業することができることを確保する。プルベースモデルにアプリケーションが存在すると、複数の非依存サービスがすべてプル動作の対象からデータを回収するために競合するときに、コンテンションの課題に直面するため、コンテンションの課題の潜在性につながる。

【００５８】

さらに、図３の例では、ボックスまたはオブジェクト（例えば、コレクタ１０２、アグリゲータ１０４および１０６、ならびに出力１０８）の各々は、アクタモデルのアクタである。アクタモデルは、同時並行動作をモデル化する手段として設計された計算のモデルである。アクタモデルは、低レベルの同時並行プリミティブを抽象化する手段として、非依存エンティティ間でメッセージパッシングを使用することに依存する。Ｅｒｌａｎｇは、アクタモデルの実装例で最も有名な例であり、テレコム用の高度な同時並行アプリケーションの開発を支援するようにＥｒｉｃｓｓｏｎによって設計されている。アクタモデルは、本明細書に記載される制御システムが、マルチスレッドの詳細を気にする必要がない一方で、任意の特定のハードウェアマシンですべてのリソースの大部分を利用可能にもしながら、システムが任意の数のコアを有するマシンで作動することができるような方法でコードを使用することを可能にする、強力な抽象化を形成する。

【００５９】

本明細書に記載される制御システムアーキテクチャの別の重要な態様は、構成要素（例えば、仮想デバイス、仮想構成要素またはエンティティ、ハードウェアデバイス、およびアクタ）間のデータまたはデータメッセージングが自己記述型であることである。自己記述型データメッセージングプロトコルをサポートするには、制御システム１０の構成要素は、データおよびデータ記述の両方を独立してまたは一緒に受信することができなければならない。このようにして、アクタは、データ記述を使用して、最初に受信中のデータを解釈し、次いで、データに作用することができる。

【００６０】

データ記述およびデータの両方を自己記述型データシステムで送信する方法を図４に詳細に例示する。特に、図４のダイアグラム１２０に例示されるように、アクタ１２２は、プラント（またはプラント内の別のアクタ）１２４からデータを受信し、かつ同様に、データ１２４の受信と同時に、または受信前もしくは受信直後のその他の時点で受信し、アクタ１２０は、アクタ１２２がデータ１２４を復号化および理解することを可能にする方法で、データ１２４を記述する参照データ１２６を受信する。データ１２４および参照データ（メタデータとも称される）１２６は、例えば、図４の例に示されるように、圧縮された形態で伝送されても、ＪＳＯＮ形態で拡張されてもよい。しかしながら、データは、任意の所望のプロトコルおよび／またはデータフォーマットで表現または送信され得る。いずれの場合でも、参照またはメタデータ１２６は、受信されているデータ１２４を解釈するためにアクタ１２２によって使用される。

【００６１】

図１および２のシステムでは、コントローラ中継サービスマシン７４は、仮想マシン５２、５４等がアクタとしてモデル化されている場合、図４のスキームを使用して、ノード１４内のさまざまな他の仮想デバイスのデータ復号化および解釈を実施することができることが理解されよう。また、本明細書に記載されるアクタベースのモデルは、各仮想デバイスまたは論理的構成要素が別個のアクタであるように、ＢＦＮ　Ｉ／Ｏコントローラ１２内に描写された仮想デバイス、および高機能ノード１４内の仮想デバイスまたはエンティティで使用され得ることを理解されたい。この場合、ＢＦＮコントローラ中継器７４は、本明細書に記載される自己記述型データメッセージングスキームに基づいてデータ復号化を実施し、解釈されたデータを他の仮想マシン（例えば、仮想コントローラ５２、仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ５４、データ分析マシン６４等）のうちの１つ以上に提供することができる。同様に、ＢＦＮコントローラ中継器７４は、他の仮想マシン５０、５２、６２、６４のうちの１つから送信されたデータメッセージを外部デバイス（例えば、ＤＮＣ　Ｉ／Ｏコントローラ１２）に、仮想デバイスによって送信されたデータメッセージと組み合わせて、これらのデバイスまたはメッセージのデータ記述を提供することによって、提供するか、または適合させることができる。

【００６２】

しかしながら、重要なことには、仮想コントローラ５２内の制御ブロックまたはモジュールの各々、データ分析マシン６４内の各データ分析ブロック等のような、仮想デバイス内の実際の仮想エンティティのさまざまなもの、または所望の場合には、それらの各々は、アクタモデル内の別個のアクタであってもよい。同様に、この場合、ＢＦＮ　Ｉ／Ｏコントローラ１２内の論理的エンティティの各々は、アクタモデル内の別個のアクタであり得る。結果として、データ中継またはデータメッセージングは、（例えば、本明細書に記載される自己記述型データスキームを使用して）各アクタによって実施され得る。この場合、この機能は別個のアクタによって提供されるため、コントローラ中継サービスマシン７４は、高機能ノード１４には必要ない場合がある。

【００６３】

同様に、上記のように、基本機能ノード１２を介して接続する代わりに、制御システムアーキテクチャを拡張して、プロセス制御デバイスまたはフィールドデバイスなどの他のデバイスをネットワーク１６に直接接続させることができる場合、デバイス（例えば、フィールドデバイス）は、ネットワークプロトコルとの通信をサポートし、通信、データ操作、およびかかる通信を可能にするバルブ診断などの特定の機能に使用され得る１つ以上のアクタを含むことができる。ここで、アクタは、制御システム内でフィールドデバイスレベルに下位へ広がっている。

【００６４】

いずれの場合も、アクタベースのアプローチは、例えば、バッチ制御および状態ベースの制御を含むさまざまなタイプの制御に非常に適している場合がある。バッチ制御を用いて、アクタの各々は、バッチがレシピを進行するにつれて、他の関係するアクタにメッセージを送信する。同様に、状態ベースの制御またはそれについての単純なシーケンス制御を用いて、アクタは、シーケンスが進行するにつれて、他の関係するアクタにメッセージを送信する。これらの場合、アクタ間でメッセージを送信する以外に、アクタ間の密接に連結された連携の必要はない。さらにまた、アクタ間のメッセージは任意の所望のフォーマットおよびプロトコルのものであり得、ＨＡＲＴまたはＦｉｅｌｄｂｕｓプロトコルメッセージなどの他のプロトコルメッセージを包含し得ることが理解されよう。同様に、アクタは、バスベースのネットワークで使用されるのと同じ基本原則を使用して、メッシュネットワークを介して通信することができるため、通信ネットワークが（例えば、有線もしくは無線ネットワーク、またはこれら２つの組み合わせに基づく）メッシュ通信システムである場合でも、アクタベースのモデルを使用することができる。

【００６５】

従来のプロセス制御システムに適用されるアクタモデルの使用の一例として、従来のプロセス制御モジュールの各機能ブロックは、各アクタが次いで（他のアクタに対して非同

期に）実行され、その結果をチェーン内の次のアクタに送信し、これにより、機能を下位のチェーンに駆動させるときに、アクタモデルの別個のアクタとして指定され得る。このマッピングは、例えば、カスケードプロセス制御ループでよく機能する。

【００６６】

アクタベースのモデルをプロセス制御システムのプロセス制御ループに適用する別の例として、図５は、ＢＦＮ　Ｉ／Ｏコントローラ１２のうちの１つ以上と高機能ノード１４のうちの１との間で分離された単純なＰＩＤ制御ループ１３０を実施する１つの方法を例示する。この例では、プロセス制御ループは、フィールドデバイス内またはＢＦＮ　Ｉ／Ｏコントローラ１２のうちの１つに配設されたＡＩ（アナログ入力）ブロック１３２を含み、高機能ノード１４の仮想コントローラデバイス５２に実装されたＰＩＤ制御ブロック１３４に測定信号を提供し、バルブ内またはＢＦＮ　Ｉ／Ｏコントローラ１２のうちの１つに配設されたＡＯ（アナログ出力）ブロック１３６に制御出力を提供し、バルブを位置決めする現在のバルブのフィードバック測定を受信する。従来の制御システムでは、ブロック１３２、１３４、および１３６は、プロプライエタリであり、同じプロトコルまたはメッセージングスキームに適合し得、これらのブロックは、専用または事前構成された通信パスを介して事前構成されたハードウェアに信号を送信して、制御通信を実装するように構成され得る。ブロック１３２、１３４、１３６はすべて同じプロトコルのものであるため、送信および受信ブロックは、これらのメッセージが同じプロトコルおよびデータフォーマットに適合している場合、常にデータメッセージをどのように読み取るかを知っている。

【００６７】

しかしながら、図５の下部は、アクタベースのモデルを単純な制御ループ１３０に適用して、単方向および自己記述型のメッセージングを提供するか、または提供することができ、これにより、図１および２のネットワーク１６を介して（および同じノード内のアクタ間、または単一ノードの物理的マシンもしくはプロセッサ間でさえ）メッセージプロトコルおよびフォーマットの独立性を可能にする、方法を例示する。特に、アクタベースのモデル１４０では、複数のデータコレクタブロック１４２Ａおよび１４２Ｂを提供して、（ＡＩブロック１３２に関連付けられた）センサ測定値および（ＡＯブロック１３６に関連付けられた）バルブ位置測定値を収集する。これらのデータコレクタブロック１４２Ａおよび１４２Ｂは、異なるフィールドデバイス内および／または同じもしくは異なるＢＦＮ　Ｉ／Ｏコントローラデバイス１２内に位置し得る。しかしながら、図５に例示されるように、ブロック１４２Ａおよび１４２Ｂの各々は、データ中継器ブロック１４４Ａおよび１４４Ｂに接続され、それらの各々は、自己記述型データプロトコルを使用して、ネットワーク１２にわたってデータメッセージングを提供する。ブロック１４４Ａおよび１４４Ｂの各々は、ブロック１４２Ａおよび１４２Ｂからのデータを同じまたは異なるメッセージングプロトコルパケットにカプセル化することができ、ベースメッセージ（データメッセージとも称される）内のデータを記述する他のメタデータメッセージを提供することができる。ブロック１４４Ａおよび１４４Ｂは、（自己記述型メッセージスキームに関連付けられた）これらのメッセージの対を、この場合、高機能ノード１４のうちの１に配設された仮想コントローラ５２のうちの１つに存在し得る下流ブロック１４６Ａおよび１４６Ｂにアドレス指定されたネットワーク１６にわたって送信することができる。ブロック１４４Ａおよび１４４Ｂは、異なるメッセージプロトコルおよび／または異なるメッセージフォーマットもしくはデータフォーマットを使用することができ、ひいては、データコレクタブロック１４２Ａおよび１４２Ｂは、同じプロプライエタリなプロトコルまたはスキームに適合する必要はないが、異なるプロプライエタリなプログラミングを使用する完全に異なる製造業者によって提供され得ることに留意されたい。さらに、中継器ブロック１４４Ａおよび１４４Ｂは、（例えば、互いに）完全に異なるデータフォーマットまたはメタデータ記述を使用してメッセージを送信することができ、これもまた、これらのブロックの使用時の独立性およびオープンさを高める。

【００６８】

さらにまた、中継器ブロックまたはアクタ１４６Ａおよび１４６Ｂの第２の対は、それぞれブロック１４４Ａおよび１４４Ｂからデータメッセージおよびメタデータメッセージを受信し、内部の情報を使用して、データ（例えば、センサ測定データ）を復号化する。この復号化プロセスは、メッセージのメッセージプロトコル（例えば、パケットフォーマット）を判定すること、ならびにメタデータメッセージに基づいてメッセージ内のデータの意味を定義するデータフォーマットを判定すること、を含み得る。次いで、中継器ブロック１４６Ａおよび１４６Ｂは、復号化されたデータをＰＩＤ制御ブロック１４８に提供する。中継器ブロック１４６Ａおよび１４６Ｂは、互いにおよびＰＩＤブロック１４８とは独立して動作し得ることに留意されたい。プロプライエタリな制御ルーチンを設けることができるＰＩＤ制御ブロック１４８は、センサ測定入力を使用し、その出力で制御信号を生成することができる。ＰＩＤ制御ブロックからの制御信号は、ネットワーク１６を介した通信に適したメッセージ（メッセージパケット）内のデータを符号化することができ、そのデータを、ＢＦＮ　Ｉ／Ｏコントローラ１２のうちの１つの中の論理的エンティティにアドレス指定されたネットワーク１６にわたるメタデータメッセージと共に送信することができる、別の中継器ブロック１５０（アクタモデル内の別個のアクタ）に提供され得る。次いで、メッセージは、（メタデータおよびデータメッセージ内のデータに基づいて）ネットワーク１６を介して受信されたメッセージ内のデータを復号化する、適切なＢＦＮ　Ｉ／Ｏコントローラ１２内のさらなる中継ブロック１５２によって受信される。さらに、中継器ブロック１５２によって復号化されたメッセージは、ネットワーク１６にわたって送信される場合、任意のメッセージプロトコル（例えば、任意のパケットベースのプロトコル）であり得、中継器ブロック１５２がメタデータを使用して、データメッセージのフォーマットを復号化または理解するため、任意のメッセージもしくはデータフォーマットまたはスキームを使用し得る。いずれの場合も、中継器ブロック１５２は、次いで、例えば、ＢＦＮ　Ｉ／Ｏコントローラ１２のうちの１つまたはフィールドデバイス内の論理的構成要素等の中にあり得る制御ブロック１５４に制御信号を提供して、例えば、バルブの位置を変更させる。

【００６９】

述べられるように、ブロック１４２Ａ、１４２Ｂ、１４４Ａ、１４４Ｂ、１４６Ａ、１４６Ｂ、１４８、１５０、１５２、および１５４の各々は、同じまたは異なる物理的ハードウェアで動作し得る別個のアクタである。これらのアクタの各々は、互いに非同期の単方向を使用し、ひいては、いずれのプルベースの通信も実施しない。さらにまた、アクタまたはブロックの各々は、本明細書に記載される自己記述型データプロトコルを使用してデータを伝送し、通信プロトコルおよびフォーマットを独立させることができる。場合によっては、自己記述型データプロトコルのメタデータメッセージは、データメッセージと同時（例えば、わずかに前またはわずかに後）に送信され得るか、または下流アクタによる使用のために、周期的にまたは他の非周期的な時間に下流アクタに送信され得る。つまり、メタデータメッセージは、受信アクタはデータメッセージの復号化に使用するためのメタデータを記憶することができるため、自己記述型メッセージプロトコルのデータメッセージほど頻繁に送信する必要がない場合がある。したがって、メタデータメッセージは、典型的には、データメッセージよりも少ない頻度で送信され得、メタデータメッセージは、メタデータが変更、追加、または別様に改変された場合、および／またはメタデータの特定の上流アクタからデータメッセージを受信し得るシステムに追加された新しいアクタに通知するためにのみ送信され得る。

【００７０】

さらにまた、図５に例示されるように、アクタベースのモデルは、従来の制御ループのＡＩブロック１３２にほぼ対応するアクタ１４２Ａおよび１４４Ａを有し、従来の制御ループ１３０のＰＩＤブロック１３４に対応するアクタ１４６Ａ、１４６Ｂ、１４８、および１５０を有し、従来の制御ループ１３０のＡＯブロック１３６に対応するアクタ１４２Ｂ、１４４Ｂ、１５２、および１５４を有する。当然のことながら、チェーンまたはアクタモデル１４０に他のアクタを追加して、他のアクションを実施することができる。例え

ば、中継器ブロックを２つのアクタに分割することができ、そのうちの一方は、メタデータスキームに従ってデータメッセージをフォーマット化し、そのうちの他方は、第１のアクタのデータメッセージを特定のパケットベースのプロトコルのパケットにフォーマット化して、そのメッセージを、ネットワーク１６を介して送信する。

【００７１】

理解されるように、本明細書で論じられるアクタベースのモデルの使用は、単方向のメッセージングを提供し、自己記述型メッセージングプロトコルの使用を可能にし、制御システム内のさまざまな論理的要素の非同期動作を可能にし、これにより、制御システム１０内で実行されるアプリケーションに上で論じられるリアクティブな利点を提供する。例えば、このアクタベースのモデルを使用すると、ユーザが中継器ブロックを制御ブロックとは独立して更新することを可能にする。この特徴は、ユーザまたはシステムが、データプロトコルまたはデータメッセージングフォーマットが変更されるたびに（例えば、実装されるパケットベースのメッセージングプロトコルに新しい更新が提供されるとき）、制御ブロック（例えば、制御ブロック通信）に接触するか、またはそれを再構成する必要なしに、中継器ブロックを変更または再構成することを可能にする。一方、この特徴は、データメッセージングまたは中継器ブロックを変更する必要なしに、（例えば、これらのブロックのプロプライエタリな動作を変更または更新するために）制御ブロックを更新または変更することを可能にする。次いで、この特徴は、変更に直接関連付けられていない機能を有する他のアクタを再構成することなく、変更に関連付けられたアクタを中心に再構成活動を行うことを可能にする。この動作により、システム１０の構成がより容易になり、より隔離され、エラーの検出と修正がより容易になり、システムまたはコントロール内のアプリケーションのエラーに対する回復力が高まる。

【００７２】

前述したように、本明細書に記載される制御システムアーキテクチャは、ワークロードの変化および構成要素の障害の両方を容易に取り扱うことができる。特に、比較的複雑な制御システムを扱うときに発生する可能性のある多くの異なるタイプのエラーおよび問題が存在する。いくつかの例としては、ハードウェア障害、ソフトウェア障害、ネットワーク接続障害、メンテナンス障害、および過剰な処理障害が挙げられる。

【００７３】

概して言えば、アプリケーションまたは論理的エンティティをホストする計算ノードの物理的構成要素が失敗すると、ハードウェア障害が発生する。この障害は、いくつかの（例えば、センサ）アクタをホストしているノードが失敗した場合を網羅する。典型的には、ハードウェア障害としては、ハードドライブ障害、プロセッサ障害等のようなものが挙げられる。ハードウェア障害の場合、システム（例えば、図２のハイパーバイザ５０）は、そのノード上に位置するアクタ（例えば、センサ）表現を置換デバイスまたはノードへ容易に移行することができる（例えば、センサをバックアップセンサ、ソフトセンサ、または障害条件ハンドラーに移行することができる）。したがって、アクタまたは他の論理的エンティティを、制御システム１０の同じノード上の他のハードウェアまたは他のノード上のハードウェアの任意の組み合わせに流すことができる。ここで、メッセージはハードウェア固有の場所ではなく、他のアクタにアドレス指定されているため、フロントエンドノードは、メッセージを正しくリダイレクトすることができるが、これらのメッセージの内部には関係がない。したがって、（本明細書に記載される通信はハードウェアの場所に依存しないようにできるか、またはそのようにセットアップされるため）ルーチン、コントローラ、中継器を再構成する必要なしに、かつ特に、移動する構成要素の通信を再構成する必要なしに、図２のスーパーバイザまたはハイパーバイザ５０は、例えば、制御ルーチン、仮想コントローラ、中継サービス等のような構成要素を１つのハードウェアデバイスから（またはハードウェアデバイス内の異なるメモリまたはプロセッサに）移動させることができる。

【００７４】

不正な動作が発生し、ソフトウェアが未だ取り扱っていない例外をスローすることにつ

ながると、ソフトウェア障害が発生する。このタイプの障害としては、アプリケーションに落ち度がある汎用事例、例えば、アプリケーションが接続を担当するデータベースにアクセスできない場合、またはアプリケーションがゼロで除算しようとする場合、が挙げられる。このタイプの障害を取り扱うために、システムは、個々のセンサまたはアプリケーションを監視し、障害が発生した場合に、典型的にはアクタまたは表現を再起動することによって、アクタが既知の作業状態に戻されることを保証するスーパーバイザ（例えば、ハイパーバイザ５０）を含む。

【００７５】

ネットワーク接続障害は、ルータ障害、スイッチ障害、またはその他のネットワークハードウェア障害の結果として、マシンが相互に通信することができない場合に発生する。同様に、環境メンテナンスルーチン障害としては、データヒストリアンから大量のデータを収集するなど、いくつかの潜在的に長時間作動されるタスクを作動することが挙げられる。これらのすべての場合において、障害によりアプリケーションからリソースが奪われ、アプリケーションが一時停止する可能性がある。この後者の状況は通常、障害とは見なされないが、アプリケーションが十分な時間内にハートビート要求に応答しない可能性があるため、他のマシンからは障害として認識される可能性がある。このタイプの障害を修正するために、システムは、アクタまたは表現が利用可能な処理能力がより高い場所に移動することを保証するスーパーバイザ（例えば、図２のハイパーバイザ５０）を含む。

【００７６】

さらに、アクタ（例えば、センサ）が他のアクタよりも多くの処理を必要とする場合、過剰処理障害が発生する可能性がある。例えば、１つのセンサアクタが１秒ごとに１回よりも頻繁に読み取り値を発生させる場合があり、これは、処理持続時間への波及効果を有する。この場合、システムは、他のセンサが応答し続けること防ぐことになるため、他のセンサアクタの処理時間を奪うことは望まない。このタイプの障害を取り扱うことができるようにするために、システムは、センサまたは他のアクタが、独自の専用スレッドにあるマシンであるか、すべて一緒に完全に異なるマシンであるかどうかにかかわらず、利用可能な処理能力がより高い場所に移動することができることを保証するスーパーバイザ（例えば、図２のハイパーバイザ５０）を含む。

【００７７】

場合によっては、ＢＦＮ　Ｉ／Ｏコントローラハードウェアは、デュアルコアプロセッサを利用することができ、コアのうちの１つは、高速制御活動（例えば、１０ｍｓおよび２０ｍｓ実行）に使用され、他のコアは、他のすべて（例えば、Ｉ／Ｏ、メッセージ取り扱い、５０ｍｓおよび１００ｍｓ制御実行等）に使用される。ＢＦＮ　Ｉ／Ｏコントローラキャリアは、冗長ハードウェアＢＦＮをサポートすることができ、キャリアは、Ｉ／Ｏベースプレートへの接続を提供する。同様に、高速制御活動は、例えば、高機能ノード１４内の仮想コントローラの実行レートよりも少なくとも５倍または少なくとも１０倍大きい実行速度またはレートで動作し得る。

【００７８】

別の例では、本明細書に記載される制御アーキテクチャは、高速制御活動および低速制御活動を同時に実施し、依然として、これらの両方またはすべての制御速度が定格速度で適応および実施されることを保証することができる。図６は、Ｅｔｈｅｒｎｅｔローカル高速制御ネットワーク１６Ａを介して接続された３つのＢＦＮ　Ｉ／Ｏコントローラ１２を含む例示的な高速制御ネットワークを例示する。ＢＦＮ　Ｉ／Ｏコントローラ１２のすべては、この例では、高機能ノード１４内にあってもよく、同じローカル高速制御ネットワークを共有する１つのＢＦＮコントローラ中継器７４に連結される。ＢＦＮコントローラ中継器７４およびＢＦＮ　Ｉ／Ｏコントローラ１２は、ローカル高速制御ネットワーク１６Ａを介して構成、パラメータ変更、および制御データを通信する。ＢＦＮ　Ｉ／Ｏコントローラ１２はまた、ローカル高速制御ネットワーク１６Ａを介して、高速パラメータおよび入力データを他のＢＦＮに通信する。各ＢＦＮ　Ｉ／Ｏコントローラ１２は、５０ｍｓ間隔で低速パラメータと入力データをブロードキャストし、ＢＦＮ制御の実行がＢＦ

Ｎ　Ｉ／Ｏコントローラ１２間の高速パラメータ転送と同期されないことが留意されよう。この場合、高速制御パラメータが定格速度において高速制御ネットワーク１６Ａ上で通信されることを確実にするために、高速制御ネットワークは、バス上の高速制御メッセージを優先して、これらのメッセージがバスを介して高速レート（例えば、１０ｍｓまたは２０ｍｓ）で通信されることを確実にする時間依存ネットワーク（ＴＳＮ）スイッチを備えたＴＳＮであるか、またはＴＳＮを含み得る。高速制御ネットワークリンク１６Ａを形成するためにネットワーク１６でＴＳＮスイッチを使用することは、ＢＦＮ　Ｉ／Ｏコントローラ１２と高機能ノード１４との間で同じネットワークを介して高速制御および低速制御の両方を可能にするため望ましい。

【００７９】

ある場合には、ＢＦＮ制御モジュール（従来のモジュールまたは上に記載されるアクタベースモデル内のアクタである得る）は、アクティブなＢＦＮ　Ｉ／Ｏコントローラ１２で実行される。ＢＦＮ　Ｉ／Ｏコントローラ１２は、ローカル入力、ならびにローカル高速制御ネットワーク１６Ａから受信した高速パラメータを、ＢＦＮ制御モジュールまたはアクタへの入力データとして使用する。制御実行の終了時に、制御サブシステムは、出力値を（例えば、フィールドデバイスに接続された）ローカルＩ／Ｏサブシステムに書き込み、１０ｓごと（高速パラメータデータの場合）または５０ｍｓごと（低速パラメータデータの場合）のいずれかでローカル高速制御ネットワーク１６Ａにわたって転送するために、Ｉ／Ｏデータと高速パラメータデータとをキューに入れる。当然のことながら、これらの時間は単なる例であり、他のタイミングを使用することができる。

【００８０】

ＢＦＮ　Ｉ／Ｏコントローラ１２で高速制御を実施するために、各コントローラ１２は、高速および中速（または低速）制御に使用される２つのコアまたはプロセッサを含み得る。図７は、これら２つのコア（コア０およびコア１）の間でタスクを分割して、高速制御と低速制御とを実施する１つの方法を例示する。この例では、コア０は高速制御に使用され、コア１は低速制御に使用される。Ｉ／Ｏスキャンと制御実行との同期された性質により、コア０では１０ｍｓおよび２０ｍｓのスクリューツースクリュー（ｓｃｒｅｗ－ｔｏ－ｓｃｒｅｗ）、コア１では５０ｍｓおよび１００ｍｓのスクリューツースクリューの性能の制御が可能になる。所望の場合は、１００ｍｓより遅い制御実行レートを、高機能ノード１４のうちの１つにある仮想コントローラ５２で実行することができる。

【００８１】

図７に例示されるように、コア０は、高速制御ネットワーク１６Ａ上で通信を実施する通信サブシステム２０２と、高速コンピューティング制御活動を（例えば、１０または２０ｍｓサイクルレートで）実施する高速制御ブロック２０４とを含む。コア０の高速制御ブロックは、すべてコア１内にあるＰ２Ｐ（ピアツーピア）サブシステム通信ブロック２０６、低速制御ブロック２０８、およびＩ／Ｏサブシステム通信ブロック２１０に接続されている。さらに、通信サブシステムブロック２０２をＩ／Ｏシステムブロック２１０に接続して、高速制御データがＩ／Ｏサブシステムブロック２１０から高速制御ネットワーク１６Ａに直接提供されるか、またはその逆も同様に提供されることを可能にすることができる。Ｐ２Ｐブロック２０６は、低速制御の速度またはレート（例えば、５０ｍｓ）でピアツーピア通信を実施し、低速制御ブロック２０８は、低速制御活動（例えば、中速制御算出、警告の取り扱い、デバイス間通信の取り扱い等）を実施し、Ｉ／Ｏサブシステムブロック２１０は、Ｉ／Ｏサブシステム内のデバイス（例えば、フィールドデバイス）との通信を実施する。さらに、診断サブシステムブロック２１２は、サブシステム診断を実施するためにコア１で実行される。ＢＦＮ　Ｉ／Ｏコントローラ１２内のマルチコアプロセッサのコア間のタスクのこの分割により、所望のレートでの高速制御および低速制御の両方が可能になる。さらに、ＴＳＮを高速制御ネットワーク１６Ａとして使用することにより、高機能ノード１４などの他のデバイスとの高速および低速制御通信の両方が可能になり、これにより、所望の場合、いくつかの高速制御が高機能ノード１４で実施されることが可能になる。

【００８２】
所望の場合、ＢＦＮ　Ｉ／Ｏコントローラ１２は、警報情報、例えば、ＡＤＶＩＳＥ＿ＡＬＭ、ＣＯＭＭ＿ＡＬＭ、ＦＡＩＬＥＤ＿ＡＬＭ、およびＭＡＩＮＴ＿ＡＬＭなどのハードウェア警告をサポートすることができる。これらのイベントは、イベントおよび警告情報の処理を担当するアクタにブロードキャストされる。これらの警告の各々は、プロセス警告と同じ警告プロパティ（例えば、警告抑制）を有する。ハードウェア警告は、問題の全般的な指示を与え、その診断を使用して、正確な問題をさらに診断する。例示的なハードウェア警告を下の表１にまとめる。
【表１】

表１－ハードウェア警告

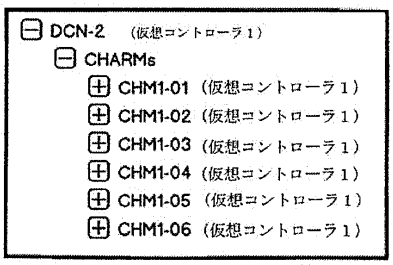
警告タイプ	条件	コメント
ADVISE	I／O整合性の問題（例えば、オープンループ、1つのスライスバスチャネルの不良）	特定の条件がI／O設計で進化する。内部条件またはセンサレベル条件であるかどうかを指示する試みが行われる。
COMM	有効なI／Oとの通信なし	
	割り当てられたBFNとの通信なし	
FAILED	HWまたはSWエラーを検出	
MAINT	不良I／Oが原因ではないBFN整合性の問題（例えば、失敗した二次ローカル高速制御ネットワークまたはI／Oバス終端エラー）	特定の条件がBFN設計で進化する。

【００８３】
ある場合には、制御システムの構成中に、各特定のＢＦＮ　Ｉ／Ｏコントローラ１２をローカル高速制御ネットワーク１６Ａ上の仮想コントローラ５２に割り当てることができ、次いで、コントローラ１２を試運転および構成することができる。試運転は、高機能ノード１４上の仮想構成システム７０で作動する構成アプリケーション、または図１のワークステーション２６のうちの１つなどのクライアントまたは他の計算ノードで提供されるＢＦＮコンテキストメニューを通じて実施され得る。ＢＦＮ　Ｉ／Ｏコントローラカードの ＥＥＰＲＯＭを使用して、ＢＦＮ　Ｉ／Ｏコントローラ１２のアドレス、名前、ダウンロードＣＲＣ、および最終ダウンロード時間を保持することができる。ＢＦＮ　Ｉ／Ｏコントローラ１２は、パワーアップすると、この情報を読み取って、パワーダウンされる前と同じ位置にあるかどうかを判定する。ＢＦＮ　Ｉ／Ｏコントローラ１２が、同じ位置で依然として試運転されていると判定し、仮想コントローラ構成と一致するフラッシュ内の有効な構成を有する場合、ＢＦＮ　Ｉ／Ｏコントローラ１２は、構成状態に遷移する。
【００８４】
上記のように、ＢＦＮ　Ｉ／Ｏコントローラの構成は、仮想コントローラ５２の下のローカル高速制御ネットワークサブシステムを介して利用可能にされ得る。図８は、特定の仮想コントローラ５２の下またはその一部としてのＢＦＮ　Ｉ／Ｏコントローラネットワークの拡大図を描写する構成アプリケーションの画面表示を例示する。この例では、３つのＢＦＮ　Ｉ／Ｏコントローラ１２（ＢＦＮ－１、ＢＦＮ－２、ＢＦＮ－３）が仮想コントローラ５２のローカル高速制御ネットワークの下に例示されている。ＢＦＮ－１の下のＩ／Ｏモジュールサブシステムは、そのＢＦＮに構成されたすべてのＩ／Ｏモジュールを含むことに留意されたい。ＢＦＮの下でＩ／Ｏモジュールサブシステムを拡張すると、ＢＦＮの下で有効にすることができるすべての可能なＩ／Ｏモジュールがユーザに提示されることがある。各Ｉ／Ｏモジュールには番号、すなわち、ＣＨＭ１－０１からＣＨＭ１－

１２～ＣＨＭ８－０１からＣＨＭ８－１２（８つのキャリア×１２＝９６個のＩＯモジュール／ＢＦＮ）を付けることができ、構成システムにより、ユーザがこの画面でチャネルまたはモジュールの割り当てを行うことが可能になる。下の表２は、デバイスタグへのチャネル（ＣＨＡＲＭチャネル）の割り当て例を例示する。

【表２】

表２



名前	タイプ	記述	ダウンロードの必要性	デバイスタグ：
CHM1-01	ＡＩジェネリックＣＨＡＲＭ		不要	CFC-2CHM1-01
CHM1-02	ＡＯジェネリックＣＨＡＲＭ		不要	CFC-2CHM1-02
CHM1-03	ＤＩジェネリックＣＨＡＲＭ		不要	CFC-2CHM1-03
CHM1-04	ＤＯジェネリックＣＨＡＲＭ		不要	CFC-2CHM1-04
CHM1-05	ＡＩジェネリックＣＨＡＲＭ		必要	CFC-2CHM1-05
CHM1-06	ＡＯジェネリックＣＨＡＲＭ	ストリームバルブ	不要	CFC-2CHM1-06

【００８５】

さらに、ＢＦＮおよびそのＩ／Ｏモジュールに関するハードウェア警告をハードウェアエントリまたはタブによって定義することができる。

【００８６】

さらにまた、ローカル高速制御ネットワークコンテキストメニューにより、新しいＢＦＮ　Ｉ／Ｏコントローラのプレースホルダを作成することが可能になる場合がある。仮想コントローラに物理的に接続された割り当てられていないＢＦＮは、割り当てられていないＢＦＮコンテンツビューに表示される場合がある。割り当てられていないＢＦＮのフォーマットは、割り当てられていないＢＦＮのタイプ、モデル、および改訂を表示し得る。さらにまた、割り当てられていないＢＦＮ内のＢＦＮを、ＢＦＮメニューからのドラッグアンドドロップまたはコンテキストメニューのいずれかの選択によって、作成されたプレースホルダに割り当てることができる。ＢＦＮは、割り当てを実施するために物理的に存在すべきである。割り当てが行われると、仮想コントローラは、基本識別情報を受信して、名前とアドレスとを含む対応するデバイスモジュールを作成する。識別情報をＩ／Ｏデータおよび高速パラメータ公開で使用して、データのソースを一意に識別する。

【００８７】

一例では、機能していないＢＦＮ　Ｉ／Ｏコントローラの復旧時に、ＢＦＮ　Ｉ／Ｏコントローラが、同じ位置で、仮想コントローラと同じ構成で依然として試運転されていることを検出した場合、構成はローカルコールドリスタートメモリからランタイムコン構成メモリにコピーされ、Ｉ／Ｏモジュールがダウンロードされ、制御実行が開始される。さらにまた、ＢＦＮ　Ｉ／Ｏコントローラは、合計ダウンロードをサポートすることができ、ユーザが個々のＩ／Ｏモジュールをダウンロードすることができる場合とできない場合とがある。合計ダウンロードの場合、ＢＦＮ　Ｉ／Ｏコントローラは、ダウンロードを処理し、ダウンロードの妨害効果を最小限に抑えるために、どの構成要素が変更されたかを判定する（すなわち、「完全一致」ダウンロードである）。例えば、１つの制御戦略（モジュール）またはＩ／Ｏモジュールのみが変更された場合、そのアイテムのみが更新される。ＢＦＮ　Ｉ／Ｏコントローラは、ダウンロードで変更されたすべてのＩ／Ｏモジュールを構成し得る。ＢＦＮ　Ｉ／Ｏコントローラが、Ｉ／Ｏモジュールとの失われた通信を再確立したことをこれまでに検出した場合、ＢＦＮ　Ｉ／Ｏコントローラは、モジュールに必要な構成シーケンスを送信して、モジュールをオンラインに戻すことができる。これにより、障害回復がもたらされる。

【００８８】

さらに、ＢＦＮ　Ｉ／Ｏコントローラは、入力モジュールと出力モジュールとを含む、任意の数とタイプのＩ／Ｏモジュールをサポートし得る。一例では、以下に記載されるタ

イプの合計９６個のＩ／Ｏモジュールをサポートすることができる。特に、サポートされ得る例示的なＢＦＮ　Ｉ／Ｏ入力モジュールを表３にまとめる。

【表３】

表３－ＢＦＮ入力ＩＯモジュールのタイプ

クラス	ハードウェアのタイプ	機能性
アナログ入力（AI）	AI　4～20mA　HART　I／0モジュール	HARTアナログ入力4～20mA アナログ入力4～20mA アナログ入力0～20mA
	AI　4～20mA　HART（本質的に安全）I／0モジュール	
	AIジェネリックI／0モジュール	
離散入力（DI）	DI　NAMUR　I／0モジュール	離散入力 パルスカウント入力
	DI　24VDC低位I／0モジュール	
	DI　24VDC隔離I／0モジュール	
	DI　120VAC隔離I／0モジュール	
	DI　230VAC隔離I／0モジュール	
	IS　DI　NAMUR　I／0モジュール	
	DIジェネリックI／0モジュール	
Thermocouple__mv	LS熱電対／mV　I／0モジュール	タイプB熱電対入力 タイプE熱電対入力 タイプJ熱電対入力 タイプK熱電対入力 タイプN熱電対入力 タイプR熱電対入力 タイプS熱電対入力 タイプT熱電対入力 ±20ミリボルト入力 ±50ミリボルト入力 ±100ミリボルト入力
	LS熱電対／mV（本質的に安全）IOモジュール	
	LS熱電対／mVジェネリックI／0モジュール	
RTD入力	LS　RTD／抵抗入力IOモジュール	Pt　100　RTD入力 Pt　200　RTD入力 Pt　500　RTD入力 Pt　1000　RTD入力 Ni　120　RTD入力 Ni　100　RTD入力 Ni　200　RTD入力 Ni　500　RTD入力 Ni　1000　RTD入力 Cu　10　RTD入力 抵抗RTD入力
	LS　RTD／抵抗入力（本質的に安全）IOモジュール	
	LS　RTDジェネリックI／0モジュール	
電圧	LS　AI　0～10VDC隔離I／0モジュール	0～5ボルト入力 0～10ボルト入力 1～5ボルト入力 －1～＋1ボルト入力 －5～＋5ボルト入力 －10～＋10ボルト入力
	LS　AI　0－10ジェネリックI／0モジュール	
電力	LS　24VDC電力I／0モジュール	24　VDC電力

【００８９】

サポートされ得る例示的なＢＦＮ　Ｉ／Ｏ出力モジュールを表４にまとめる。

【表４】

表４－ＢＦＮ出力ＩＯモジュールのタイプ

クラス	ハードウェアのタイプ	機能性
アナログ出力（AO）	AO　4～20mA　HART　I／O モジュール	HART アナログ出力 4～20mA アナログ出力 4～20mA アナログ出力 0～20mA
	AO　4～20mA　HART（本質的に安全）I／O モジュール	
	AO ジェネリック I／O モジュール	
離散出力（DO）	DO ジェネリック I／O モジュール	離散出力 モメンタリ出力 連続パルス出力
	DO　100mA エネルギー限定 I／O モジュール	
	DO　24　VDC 高位 I／O モジュール	
	DO　24　VDC 隔離 I／O モジュール	
	DO　VAC 隔離 I／O モジュール	
	IS　DO　45mA　I／O モジュール	

【００９０】

さらにまた、（図２の）ＯＰＣ－ＵＡサービス６８と組み合わせた（図２の）ＢＦＮコントローラ中継器サービス７４は、ＢＦＮ　Ｉ／ＯコントローラへのＯＰＣマッピングを提供することができる。このマッピングの一例を図９～１１に関して例示する。図９では、組織的構成要素は、制御システム１０内のすべての名前付きアイテムを含む。図９のダイアグラムに示される例は、エリアおよび制御戦略（制御モジュール）を含む。プリミティブは、機能ブロック定義およびパラメータ定義で構成されている。定義および使用量は、機能ブロック、制御戦略、パラメータ、および警告へのインターフェースを定義する。インスタンスは、特定のデバイスまたはＩ／Ｏチャネルなどの特定のタグ付けされたアイテムを含む。デバイスは、コントローラ、Ｉ／Ｏモジュール、およびＨＡＲＴデバイスなどのスマートデバイスを含む。

【００９１】

さらに、制御戦略は、図１０に描写されるＡＮＳＩ／ＩＳＡ－８８モデルに従うことができる。ここでは、警告とイベントとがこの構造に結び付けられている。結果として、マッピング層は、制御戦略（モジュール）からＯＰＣ－ＵＡを介して閲覧およびアクセスされ得るオブジェクトへのインターフェースを提供する。このマッピングの基本的なオブジェクトモデルの一例を図１１に例示する。図９～１１のマッピングを使用して、上に記載される自己記述型メッセージプロトコルにメタデータメッセージを提供することができることに留意されたい。

【００９２】

さらにまた、仮想コントローラ５２は、制御戦略（例えば、制御モジュール）を実行し、ＤＣＳからＢＦＮ　Ｉ／Ｏコントローラ１２へのインターフェースを提供し、Ｍｏｄｂｕｓ／ＴＣＰ接続をサポートすることができる。構成および動作を簡素化するために、シャドウモジュールを使用して、ＢＦＮ　Ｉ／Ｏコントローラ１２で実行中の高速モジュールをシャドウすることができる。

【００９３】

構成中、構成アプリケーション（例えば、図２の仮想構成アプリケーション７０）は、仮想コントローラを追加する機能を含むコンテキストメニュー選択を制御ネットワーク上に提供することができる。図１２は、階層内の制御ネットワークアイテムの下のエクスプローラビューに仮想コントローラを例示する構成アプリケーションまたはデバイスによってユーザに提示され得る例示的な構成画面を例示する。

【００９４】

この場合、割り当てられたモジュールタブは、そのコントローラに割り当てられた制御モジュールを示す。ハードウェア警告のタイプは、ＡＤＶＩＳＥ、ＣＯＭＭ、ＦＡＩＬＥＤ、およびＭＡＩＮＴであり得る。ローカル高速制御ネットワークサブシステムは、ＢＦ

Ｎ　Ｉ／Ｏコントローラが現れる所である。さらに、割り当てられたＩ／Ｏおよび割り当てられた無線Ｉ／Ｏは、リモートＩ／Ｏ、ＷＩＯＣ、およびその他のデバイスからのリモートＩ／Ｏ割り当てを保持する（オープン制御システムプロトタイプの場合は不要）。

【００９５】

一例では、図２の仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ５４は、インテリジェントフィールドデバイスからのデータにアクセスするためのプラットフォームを提供する。仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ５４は、履歴データの収集、警告、および限定された制御をサポートするために、インテリジェントデバイスデータへのアクセスを可能に、Ｍｏｄｂｕｓ　ＴＣＰおよびＥｔｈｅｒＮｅｔ／ＩＰなどの任意の所望のプロトコルをサポートすることができる。サポートするＤＣＳ構成により、ポートの下に物理的デバイスと論理的デバイスとを備えるポートを定義することが可能になる。論理的デバイスは、アクセスするデバイス内の特定のデータアイテムを定義する信号を包含し得る。次いで、デバイスデータは、デバイス信号タグ（ＤＳＴ）を介して制御戦略（モジュール）パラメータにマッピングされ得る。ポート、デバイス、および信号は、プロトコルに応じて異なる構成プロパティを有し得る。一例では、仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ５４内のモジュールの機能ブロックの能力は、モータ制御および指揮制御をサポートするように構成されるが、アナログおよび高度な制御をサポートするように構成することができる。ある場合には、仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ５４は、任意の他のＩ／Ｏを直接参照することができない場合があり、そのため、Ｉ／Ｏを仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ５４に割り当てることができない場合がある。さらにまた、デバイスデータは、デバイスに接続された仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ５４内でのみ参照することができる場合がある。すなわち、デバイスデータを別のコントローラから直接外部から参照することはできない。

【００９６】

さらに、Ｍｏｄｂｕｓ　ＴＣＰインターフェースにより、プログラム可能な論理コントローラ（ＰＬＣ）、ドライブ、モータ制御センタ（ＭＣＣ）、分析器、およびＭｏｄｂｕｓ　ＴＣＰを通信する同様のデバイスなどのＭｏｄｂｕｓデータソースを制御システム１０に統合することが可能になる。Ｍｏｄｂｕｓ　ＴＣＰインターフェースは、Ｍｏｄｂｕｓサーバ（スレーブデバイス）との間でデータの読み取りおよび書き込みを行うＭｏｄｂｕｓクライアント（マスタ）であり得る。Ｍｏｄｂｕｓサーバデバイスは、図１３に例示されるように、直接Ｍｏｄｂｕｓ　ＴＣＰデバイスまたはＭｏｄｂｕｓ　ＴＣＰゲートウェイ下のＭｏｄｂｕｓシリアルデバイスであり得る。当然のことながら、多くの利用可能なＭｏｄｂｕｓ　ＴＣＰゲートウェイのいずれかを使用することができる。

【００９７】

さらにまた、ＥｔｈｅｒＮｅｔ／ＩＰ（ＥＩＰ）インターフェースにより、ＰＬＣ、可変速ドライブ、ＭＣＣ、分析器、海中デバイス、およびＥＩＰを通信する同様のデバイスなどのＥＩＰデータソースを制御システムに統合することが可能な場合がある。ＥＩＰインターフェースは、ＥＩＰ　Ｉ／Ｏアダプタおよび他のスキャナとの間でデータの読み出しおよび書き込みを行うＥＩＰ　Ｉ／Ｏスキャナであり得る。ＥＩＰインターフェースネットワークでは、一般的なリング、スター、およびリニアトポロジを使用することができる。

【００９８】

仮想コントローラ５２は、１つ以上のＢＦＮ制御戦略（モジュール）を記憶することができ、かつＢＦＮ　Ｉ／Ｏコントローラ１２が割り当てられている仮想コントローラ５２で作動中のシャドウモジュールを介してパラメータデータへのオンライン視認およびアクセスを提供することができる。ＢＦＮ　Ｉ／Ｏコントローラ１２がダウンロードされると、シャドウモジュールが、ＢＦＮ　Ｉ／Ｏコントローラ１２で作動中のＢＦＮモジュールから作成される。さらに、ＢＦＮ　Ｉ／Ｏモジュールデータは、ローカル高速制御ネットワーク上の任意のＢＦＮおよび仮想コントローラ５２自体からアクセス可能であってもよい。出力は、出力Ｉ／Ｏモジュールを所有するＢＦＮ　Ｉ／Ｏコントローラ１２のモジュールから駆動され得る。高速スキャン周期入力データは、所望の場合、ローカル高速制御

ネットワークを介して５０ミリ秒以上のレートで通信することができ、ネットワーク１６上のすべてのＢＦＮ　Ｉ／Ｏコントローラ１２は更新を受信する。入力データ、Ｉ／Ｏ診断および警報情報、ならびにフィールドデバイスプロトコルデータ（ＨＡＲＴデータなど）のような他のデータを、より遅いレート（例えば、１００ｍｓ）で仮想コントローラ５２に送信することができる。

【００９９】

さらにまた、Ｅｔｈｅｒｎｅｔ通信スイッチおよびＩ／Ｏポート（ＩＯＰ）を包含するハードウェアは、仮想コントローラ、ローカル高速制御ネットワーク、および制御ネットワークのネットワーク接続を物理的に隔離するサポートを含み得る。冗長ネットワーク接続がサポートされ得る。さらにまた、ＢＦＮ　Ｉ／Ｏモジュールが別のＢＦＮ　Ｉ／Ｏモジュールからのデータを使用するように構成されていると、データのソースは高速パラメータ考慮され、データの使用量は高速パラメータ参照と考慮される。追加的に、非高速パラメータを使用して、仮想コントローラ５２とＢＦＮ　Ｉ／Ｏコントローラ１２との間でデータを交換することができる。さらに、参照を仮想コントローラ５２からＢＦＮ　Ｉ／Ｏコントローラ１２に構成する場合、非高速パラメータ機構を使用することができる。同様に、システム１０は、Ｉ／Ｏモジュールの自動検知をサポートすることができる。Ｉ／Ｏモジュールタイプが構成時に定義されていない場合、自動検知時に情報が充填される。しかしながら、Ｉ／Ｏモジュールタイプが入力されていて、自動検知されたタイプが一致しない場合、調停ダイアログが示されることがある。Ｉ／Ｏモジュールが構成されているものと一致しない場合にも、調停ダイアログが発生することがある。

【０１００】

追加的に、仮想コントローラノードレベルで利用可能な情報としては、ノード自体の診断、通信、割り当てられたＩ／Ｏ、割り当てられたモジュール、リスタート、冗長性、Ｍｏｄｂｕｓ、および仮想コントローラに属するローカル高速制御ネットワークが挙げられ得る。所望の場合、ＯＰＣ－ＵＡインターフェース６８を介して履歴と傾向をサポートすることができる。警告検出とタイムスタンプは、ＢＦＮ　Ｉ／Ｏコントローラ１２で実行され、正確なタイムスタンプと警告データを提供することができる。ＢＦＮ　Ｉ／Ｏコントローラ１２からの警告は、プロセスカテゴリ内であってもよい。アセット管理を使用して、例えば、仮想コントローラ５２およびＢＦＮ　Ｉ／Ｏコントローラ１２の下にＨＡＲＴおよび他のプロトコルＩ／Ｏモジュールを位置付けることができる。アセット管理は、図２の仮想ＡＭＳデバイスマネージャ７０によって提供される。

【０１０１】

ソフトウェアに実装される場合、本明細書に記載されるアプリケーション、サービス、仮想デバイス、仮想マシン、仮想エンティティ等のいずれかを、コンピュータもしくはプロセッサのＲＡＭもしくはＲＯＭ等における磁気ディスク、レーザディスク、固体メモリデバイス、分子メモリストレージデバイス、または他のストレージ媒体などの任意の有形の非一時的コンピュータ可読メモリに記憶することができる。本明細書に開示される例示的システムは、他の構成要素の中でも、ハードウェア上で実行されるソフトウェアおよび／またはファームウェアを含むように開示されているが、かかるシステムは単に例示的であるに過ぎず、限定的であると見なされるべきではないことに留意されたい。例えば、これらのハードウェア、ソフトウェア、およびファームウェア構成要素のうちのいずれかまたはすべてが、ハードウェアにのみ、ソフトウェアにのみ、あるいはハードウェアおよびソフトウェアの任意の組み合わせで、埋め込まれ得ることが企図される。したがって、本明細書に記載される例示的なシステムは、１つ以上のコンピュータデバイスのプロセッサで実行されるソフトウェアで実装されるものとして記載されているが、提供される例がかかるシステムを実装する唯一の方法ではないことを当業者は容易に理解するであろう。

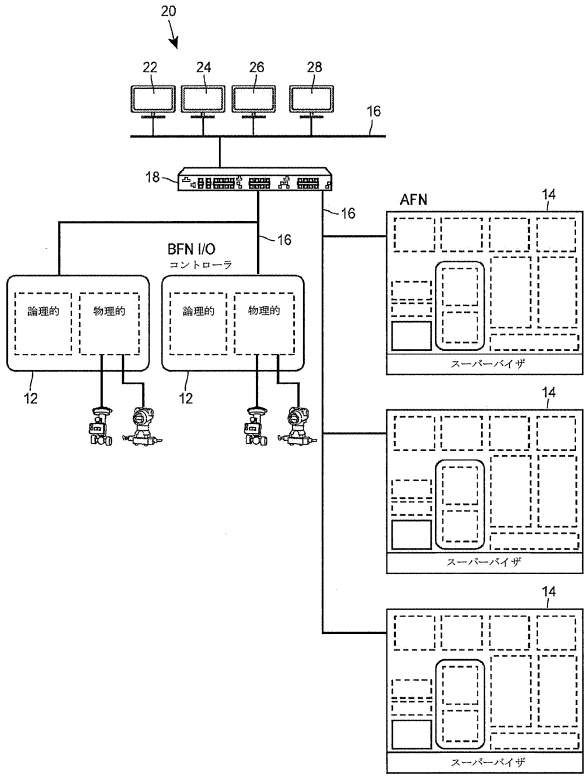
【０１０２】

したがって、本発明は具体的な例に関して記載されてきたが、これらの例は例解的であるに過ぎず、本発明の限定であることを意図せず、変更、追加、または削除が、本発明の主旨および範囲から逸脱することなく、開示される実施形態に対して行われ得ることが当

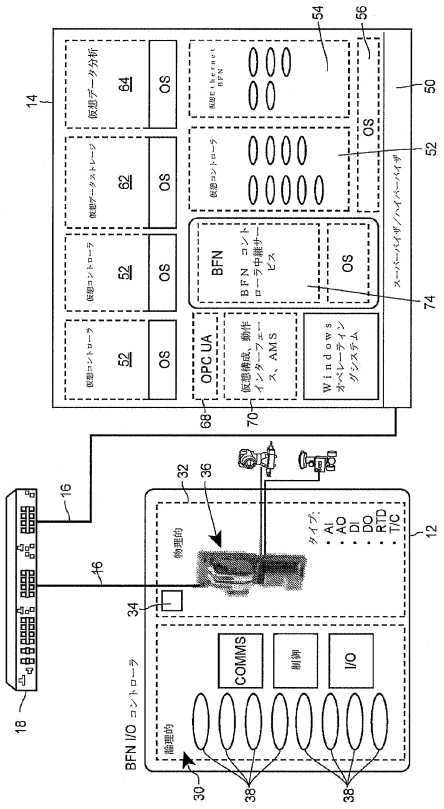
業者には明らかであろう。

【図面】

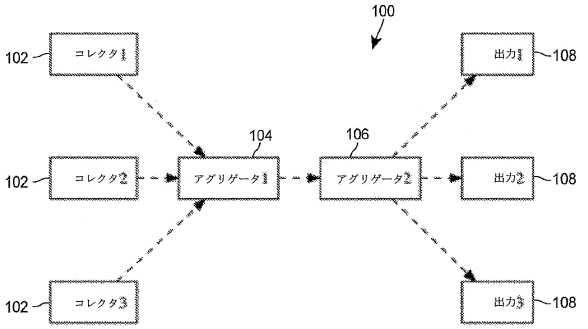
【図１】



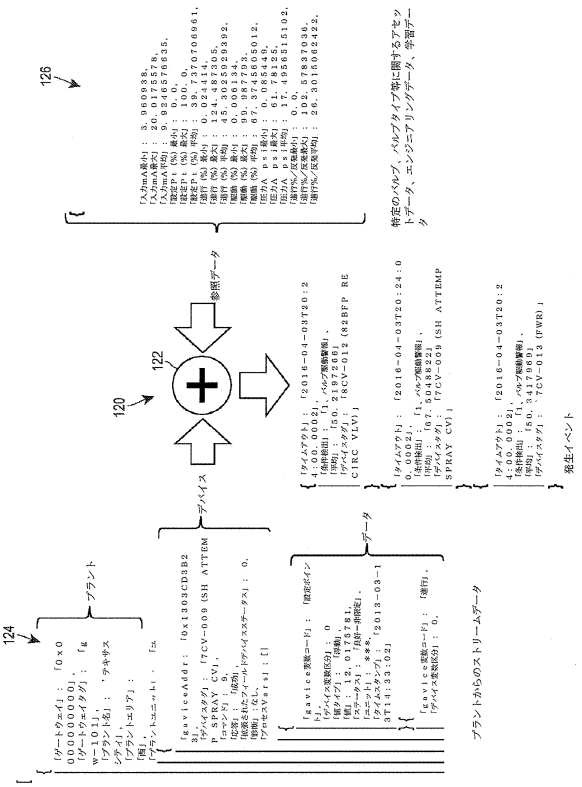
【図２】



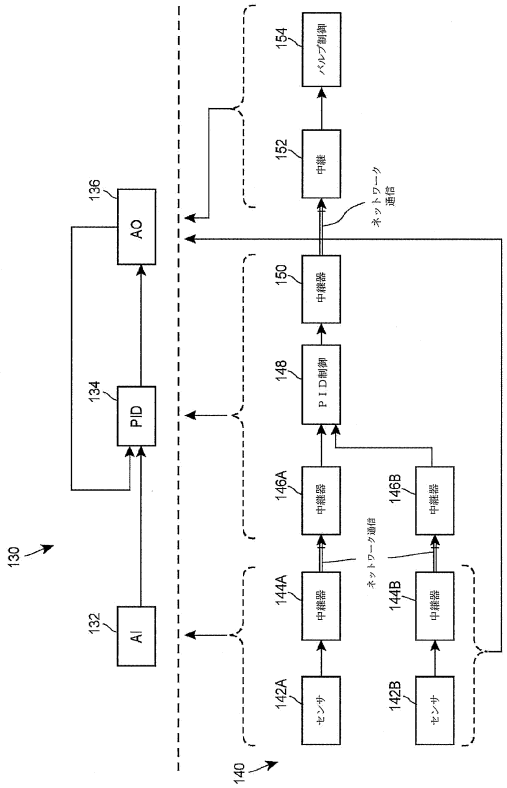
【図３】



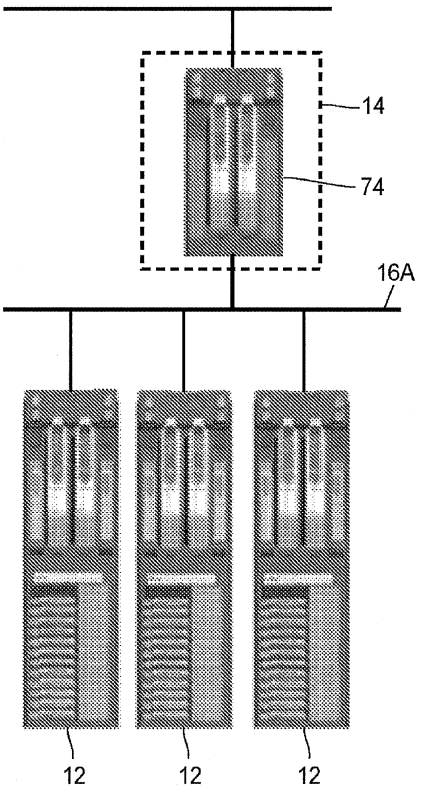
【図４】



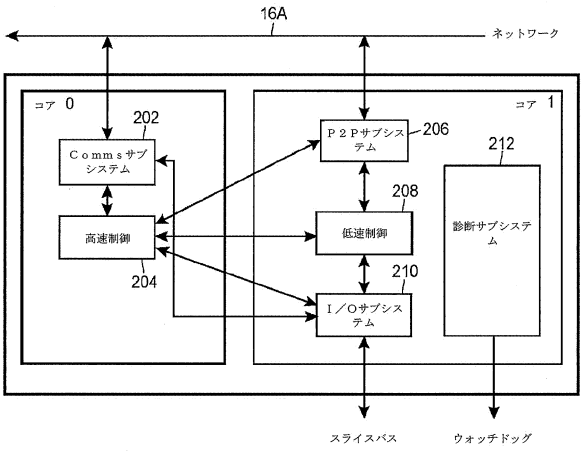
【図５】


130
132
AI
134
PID
136
AO
140
142A
センサ
142B
センサ
144A
中継器
144B
中継器
146A
中継器
146B
中継器
148
ＰＩＤ制御
150
中継器
152
中継
154
バルブ制御
ネットワーク通信
ネットワーク通信

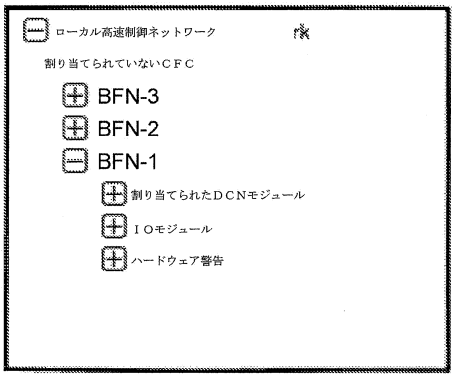
【図６】


14
74
16A
12
12
12

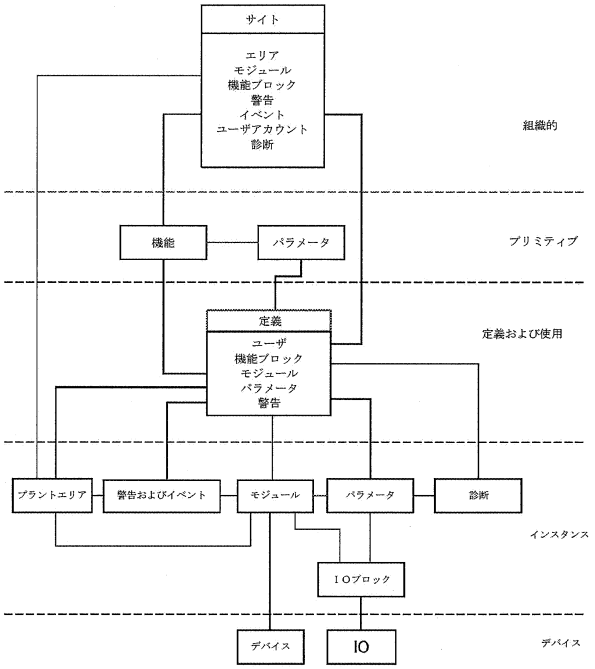
【図７】


16A
ネットワーク
コア 0
202
Ｃｏｍｍｓサブシステム
高速制御
204
コア 1
Ｐ２Ｐサブシステム
206
212
208
低速制御
診断サブシステム
210
Ｉ／Ｏサブシステム
スライスバス
ウォッチドッグ

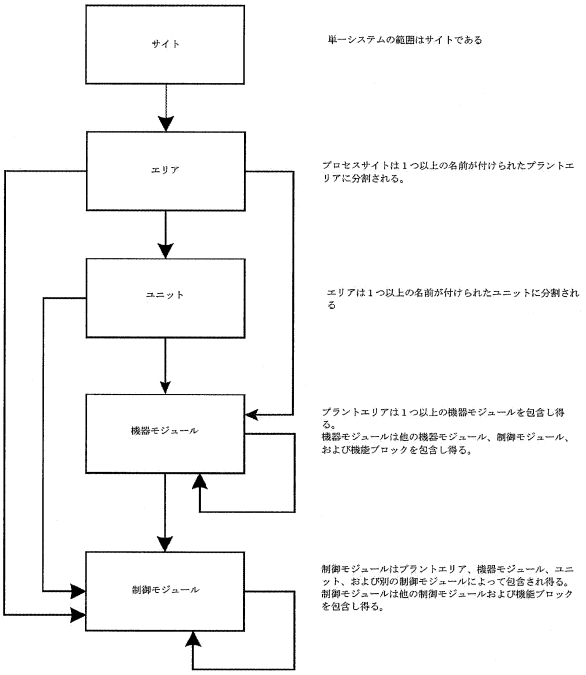
【図８】


ローカル高速制御ネットワーク
割り当てられていないＣＦＣ
BFN-3
BFN-2
BFN-1
割り当てられたＤＣＮモジュール
ＩＯモジュール
ハードウェア警告

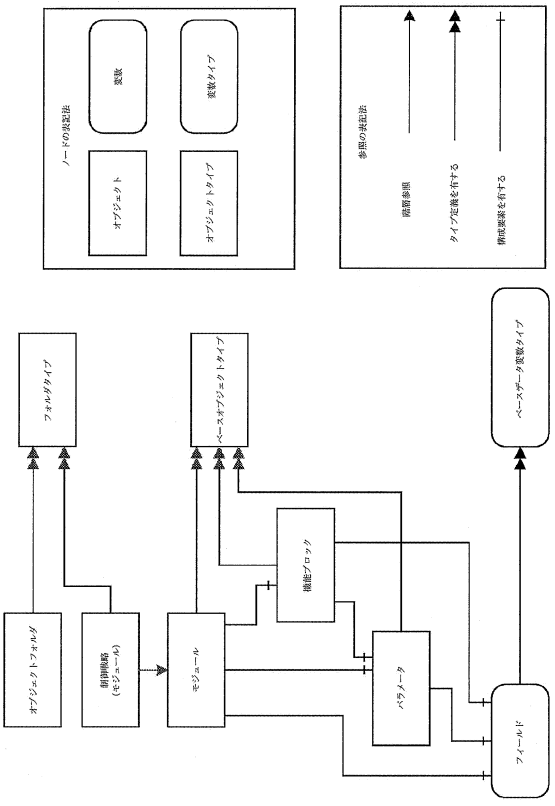
【図９】



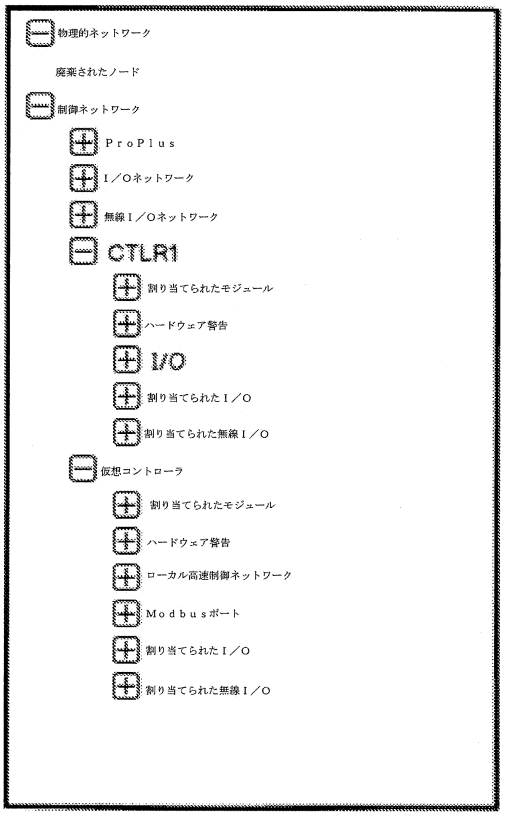
【図１０】



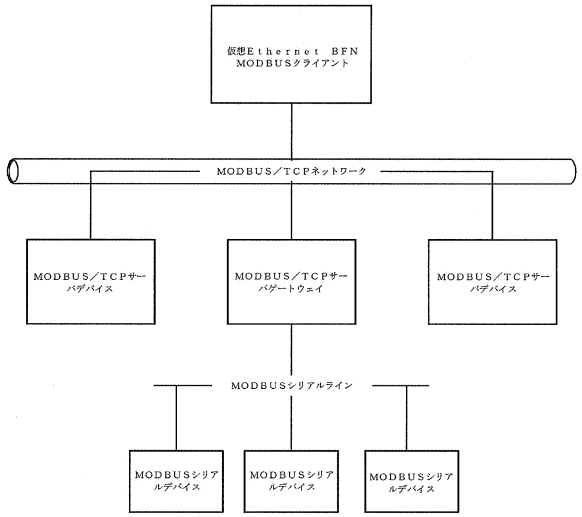
【図１１】



【図１２】



【図１３】


仮想Ｅｔｈｅｒｎｅｔ　ＢＦＮ
ＭＯＤＢＵＳクライアント
ＭＯＤＢＵＳ／ＴＣＰネットワーク
ＭＯＤＢＵＳ／ＴＣＰサーバデバイス
ＭＯＤＢＵＳ／ＴＣＰサーバゲートウェイ
ＭＯＤＢＵＳ／ＴＣＰサーバデバイス
ＭＯＤＢＵＳシリアルライン
ＭＯＤＢＵＳシリアルデバイス
ＭＯＤＢＵＳシリアルデバイス
ＭＯＤＢＵＳシリアルデバイス

フロントページの続き

(33)優先権主張国・地域又は機関
米国(US)
アメリカ合衆国 テキサス州 ７８６４１ レアンダー ワーフィールド ９
(72)発明者 ギャリー・ケイ・ロウ
アメリカ合衆国 テキサス州 ７８６３３ ジョージタウン ミシェル コート １１０
審査官 山村 秀政
(56)参考文献 国際公開第２０１７／０６４５６０（WO，Ａ１）
欧州特許出願公開第２８２９９３１（ＥＰ，Ａ２）
国際公開第２０１５／１６９３５２（WO，Ａ１）
特開平０６－２１４６０１（ＪＰ，Ａ）
特開平０７－０３６５１６（ＪＰ，Ａ）
特開２０１５－０２６２７９（ＪＰ，Ａ）
(58)調査した分野 (Int.Cl.，ＤＢ名)
Ｇ０５Ｂ １９／０４２