(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6953450号
(P6953450)
(45)発行日　令和3年10月27日(2021.10.27)　(24)登録日　令和3年10月1日(2021.10.1)

(51)Int.Cl.　　F I
G06F 9/448 (2018.01)　　G06F 9/448
G06F 8/35 (2018.01)　　G06F 8/35

請求項の数 12　(全 46 頁)

(21)出願番号　特願2018-563458(P2018-563458)
(86)(22)出願日　平成29年5月30日(2017.5.30)
(65)公表番号　特表2019-527396(P2019-527396A)
(43)公表日　令和1年9月26日(2019.9.26)
(86)国際出願番号　PCT/US2017/035021
(87)国際公開番号　WO2017/210202
(87)国際公開日　平成29年12月7日(2017.12.7)
審査請求日　令和2年2月26日(2020.2.26)
(31)優先権主張番号　62/344,192
(32)優先日　平成28年6月1日(2016.6.1)
(33)優先権主張国・地域又は機関
米国(US)
(31)優先権主張番号　15/255,857
(32)優先日　平成28年9月2日(2016.9.2)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　505384841
ザ　マスワークス,　インク
THE MATHWORKS, INC.
アメリカ合衆国　マサチューセッツ州　01760　ナティック　アップル　ヒル　ドライブ　3
3 Apple Hill Drive, Natick, MA01760 (US).
(74)代理人　100087642
弁理士　古谷　聡
(74)代理人　100082946
弁理士　大西　昭広
(74)代理人　100121061
弁理士　西山　清春

最終頁に続く

(54)【発明の名称】モデルアダプタを作成するためのシステム及び方法

(57)【特許請求の範囲】
【請求項１】
モデル要素と前記モデル要素間のリンクとを含むモデルについて、
前記モデル要素及び前記リンクの少なくとも一部が、プログラミング言語によって定義され、かつ前記モデルの実行可能エンティティを形成しており、
前記モデル要素及び前記リンクの前記少なくとも一部の挙動が、前記プログラミング言語によって暗黙的に定義された計算上の実施形態を含み、
前記モデルが、第１のグラフィカル表示形式である場合に、
（ａ）１以上のプロセッサにより、前記モデル及び前記計算上の実施形態を分析し、
（ｂ）前記１以上のプロセッサにより、前記分析に基づいて、前記計算上の実施形態の構造を変更することなく調節可能な前記モデルの前記実行可能エンティティの調節可能な属性を自動的に決定し、前記調節可能な属性が所定の特徴を有し、当該所定の特徴に基づいて、前記モデルの前記実行可能エンティティが実行され、
（ｃ）前記第１のグラフィカル表示形式とは異なる前記モデルの第２のグラフィカル表示形式で前記モデルと共に表示するために、前記１以上のプロセッサにより、前記調節可能な属性のうちの少なくとも１つを抽出し、前記調節可能な属性のうちの前記少なくとも１つが、
（ｉ）前記モデルの所与の実行可能エンティティの実行を呼び出すためのエントリポイント、（ｉｉ）前記モデルの前記所与の実行可能エンティティの実行速度、及び（ｉｉｉ）前記モデルの前記所与の実行可能エンティティによって使用されるサービスのタ

イプのうちの少なくとも１つを含み、かつ
　　対応する所定の特徴を有し、
（ｄ）インターフェースを生成し、
（ｅ）前記調節可能な属性のうちの前記少なくとも１つを前記インターフェースに明示的に表示することにより、前記調節可能な属性のうちの前記少なくとも１つを、前記対応する所定の特徴とは異なる特徴を有するように調節することを可能にすること
を含む、方法。
【請求項２】
前記所与の実行可能エンティティは、前記モデル要素又はリンクのうちの１以上を含む、請求項１に記載の方法。
【請求項３】
前記モデル及び前記計算上の実施形態を分析することは、前記モデルを分割することにより、前記モデルに含まれる複数の前記実行可能エンティティを決定することを含む、請求項１又は請求項２に記載の方法。
【請求項４】
前記インターフェースは、１以上の明示的イベント又は１以上の暗黙的イベントによってトリガー駆動されるように調節される、請求項１～３の何れか一項に記載の方法。
【請求項５】
前記１以上の明示的イベントは、ファンクションコール又はメッセージであり、前記１以上の暗黙的イベントは、トリガ信号又は周期的タイマーである、請求項４に記載の方法。
【請求項６】
第１の実行可能エンティティは、前記モデル内の第１の階層レベルにあり、第２の実行可能エンティティは、前記モデル内の前記第１の階層レベルとは異なる第２の階層レベルにあり、
前記第１の実行可能エンティティの第１のインターフェース及び前記第２の実行可能エンティティの第２のインターフェースは、同じメカニズム又は異なるメカニズムによってトリガー駆動されるように調節される、請求項１～５の何れか一項に記載の方法。
【請求項７】
前記調節可能な属性のうちの前記少なくとも１つを前記第２のグラフィカル表示形式の前記モデル内に表示することを含む、請求項１～６の何れか一項に記載の方法。
【請求項８】
前記モデルは、前記第１のグラフィカル表示形式及び前記第２のグラフィカル表示形式で実行可能である、請求項１～７の何れか一項に記載の方法。
【請求項９】
前記モデルのためのコードを生成することをさらに含み、前記コードは、前記調節可能な属性のうちの前記抽出された少なくとも１つに対応する部分を含む、請求項１～８の何れか一項に記載の方法。
【請求項１０】
複数の実行可能エンティティを含むグラフィカルモデルコンポーネントについて、
（ａ）前記グラフィカルモデルコンポーネントを前記複数の実行可能エンティティに分割し、
（ｂ）前記実行可能エンティティのうちの少なくとも１つの実行可能エンティティの１以上の調節可能な属性を識別し、前記１以上の調節可能な属性が、（ｉ）前記少なくとも１つの実行可能エンティティの実行を呼び出すためのエントリポイント、（ｉｉ）前記少なくとも１つの実行可能エンティティの実行速度、及び（ｉｉｉ）前記少なくとも１つの実行可能エンティティによって使用されるサービスのタイプのうちの少なくとも１つを含み、
（ｃ）１以上のプロセッサにより、前記グラフィカルモデルコンポーネントのためのアダプテーション層を確立し、前記アダプテーション層が、前記少なくとも１つの実行可能

エンティティの前記１以上の調節可能な属性に対応する１以上のアクセスポイントを含み、
（ｄ）前記アダプテーション層に含まれる前記１以上のアクセスポイントを、１以上のグラフィカルアフォーダンスを通して公開し
（ｅ）前記１以上のプロセッサにより、前記グラフィカルモデルコンポーネントに含まれかつ前記１以上のグラフィカルアフォーダンスにリンクされたロジックを、前記少なくとも１つの実行可能エンティティの前記１以上の調節可能な属性にバインドすること、並びに
（ｆ）次の（ｆ１）及び（ｆ２）：
（ｆ１）前記１以上のグラフィカルアフォーダンスにリンクされた前記ロジックを使用して、前記グラフィカルモデルコンポーネントの実行中に、（ｉ）前記エントリポイントを通して前記少なくとも１つの実行可能エンティティの実行を呼び出すこと、（ｉｉ）前記少なくとも１つの実行可能エンティティの前記実行速度を決定すること、及び（ｉｉｉ）前記少なくとも１つの実行可能エンティティによって使用される前記サービスのタイプを提供することのうちの少なくとも１つを実施すること、及び
（ｆ２）前記グラフィカルモデルコンポーネントのためのコードを生成し、前記生成されたコードが、前記１以上のグラフィカルアフォーダンスにリンクされた前記ロジックを実施することにより、（ｉ）前記エントリポイントを通して前記少なくとも１つの実行可能エンティティの実行を呼び出すこと、（ｉｉ）前記少なくとも１つの実行可能エンティティの前記実行速度に従うこと、及び（ｉｉｉ）前記少なくとも１つの実行可能エンティティによって使用される前記サービスのタイプを提供することのうちの少なくとも１つを実施すること
のうちの少なくとも一方
を含む、方法。
【請求項１１】
モデル要素と前記モデル要素間のリンクとを含むモデルについて、
前記モデル要素及びリンクの少なくとも一部が、プログラミング言語によって定義され、かつ前記モデルの実行可能エンティティを形成しており、
前記モデル要素及び前記リンクの前記少なくとも一部の挙動が、前記プログラミング言語によって暗黙的に定義された計算上の実施形態を含む場合に、
（ａ）１以上のプロセッサにより、前記計算上の実施形態の構造を変更することなく調節可能な前記モデルの前記実行可能エンティティの調節可能な属性を識別し、前記調節可能な属性が所定の特徴を有し、当該所定の特徴に基づいて、前記モデルの前記実行可能エンティティが実行され、
前記調節可能な属性は、（ｉ）前記実行可能エンティティの実行を呼び出すためのエントリポイント、（ｉｉ）前記実行可能エンティティの実行速度、及び（ｉｉｉ）前記実行可能エンティティによって使用されるサービスのタイプのうちの少なくとも１つを含み、
前記調節可能な属性のうちの少なくとも１つが、対応する所定の特徴を有し、前記対応する所定の特徴とは異なる特徴を有するように調節され、
（ｂ）前記１以上のプロセッサにより、前記モデルのためのコードを生成し、前記コードが、前記調節可能な属性に対応する１以上の第１のコード部分と、前記計算上の実施形態に対応する１以上の第２のコード部分とを含み、前記調節可能な属性のうちの前記少なくとも１つの調節によって、前記１以上の第２のコード部分に影響を及ぼすことなく、前記１以上の第１のコード部分の調節がもたらされる、方法。
【請求項１２】
複数の実行可能エンティティを含むグラフィカルモデルコンポーネントについて、
（ａ）前記グラフィカルモデルコンポーネントの前記複数の実行可能エンティティの調節可能な属性を識別し、
前記調節可能な属性が、（ｉ）前記複数の実行可能エンティティの実行を呼び

出すためのエントリポイント、（ｉｉ）前記複数の実行可能エンティティの実行速度、及び（ｉｉｉ）前記複数の実行可能エンティティによって使用されるサービスのタイプのうちの少なくとも１つを含み、

前記グラフィカルモデルコンポーネントのためのアダプテーション層が、前記複数の実行可能エンティティの前記調節可能な属性のためのアクセスポイントを含み、

（ｂ）少なくとも１つのプロセッサにより、前記グラフィカルモデルコンポーネントのためのコードを生成し、前記コードが、前記複数の実行可能エンティティの前記調節可能な属性のための前記アクセスポイントにリンクされた選択可能なロジックを実施する１以上のコード部分を含み、前記選択可能なロジックが、（ｉ）前記グラフィカルモデルコンポーネントの前記複数の実行可能エンティティのうちの少なくとも１つの実行を呼び出すこと、（ｉｉ）前記複数の実行可能エンティティのうちの前記少なくとも１つの前記実行速度に従うこと、及び（ｉｉｉ）前記複数の実行可能エンティティのうちの前記少なくとも１つによって使用される前記サービスのタイプを提供することのうちの少なくとも１つのためのアクセスメソッドを実施すること

を含む、方法。

【発明の詳細な説明】

【図面の簡単な説明】

【０００１】

以下の説明では、添付の図面が参照される。

【図１】一実施形態による、モデルコンポーネントを表示する例示的モデルエディタウィンドウを示す概略図である。

【図２】一実施形態による、例示的ユーザーインターフェースウィンドウを示す概略図である。

【図３】一実施形態による、図１の例示的モデルコンポーネントを含む例示的親モデルを示す概略図である。

【図４】一実施形態による、例示的モデリング環境を示す部分概略図である。

【図５Ａ】一実施形態による、例示的方法のフロー図を示す部分図である。

【図５Ｂ】一実施形態による、例示的方法のフロー図を示す部分図である。

【図６】一実施形態による、モデルコンポーネントについての例示的依存関係グラフを示す概略図である。

【図７Ａ】一実施形態による、例示的方法のフロー図を示す部分図である。

【図７Ｂ】一実施形態による、例示的方法のフロー図を示す部分図である。

【図７Ｃ】一実施形態による、例示的方法のフロー図を示す部分図である。

【図８】一実施形態による、例示的ユーザーインターフェースウィンドウを示す概略図である。

【図９】一実施形態による、例示的再利用環境を示す概略図である。

【図１０】一実施形態による、例示的モデルコンポーネントを示す概略図である。

【図１１】一実施形態による、例示的ユーザーインターフェースウィンドウを示す概略図である。

【図１２】一実施形態による、図１０の例示的モデルコンポーネントを含む例示的モデルを示す概略図である。

【図１３】一実施形態による、例示的コンピュータシステム又はデータ処理システムを示す概略図である。

【図１４】一実施形態による、例示的分散コンピューティング環境を示す概略図である。

【発明を実施するための形態】

【０００２】

例示的実施形態の詳細な説明

グラフィカルモデルは、離散システム、イベントベースのシステム、時間ベースのシステム、状態ベースのシステム、データフローベースのシステム等のような種々のタイプの物理システムをシミュレートするために使用される場合がある。モデルは、グラフィカル

モデルが実行されたときに種々の動作を実施するブロックのような種々のモデル要素を含む場合がある。ブロックは、モデル化されるシステムの挙動を表すアルゴリズム又は手順を定義するように、種々の接続要素を用いて１つに接続される場合がある。モデル要素は、要素の動的システム、状態、接合点、物理的コンポーネント等を表す場合がある。接続要素は、信号、状態遷移、イベント、物理的接続、データフロー、制御フロー、ファンクションコール等を表す場合がある。モデルは、グラフィカルモデリング環境内で構築され、実行される場合があり、グラフィカルモデリング環境は、ワークステーション又は他のデータ処理装置上で実行されるアプリケーションプログラムである場合がある。

【０００３】

所与のモデルは、システムの動作をシミュレートし、例えば、近似することができる。システムの例としては、気象システム、金融市場、プラント、コントローラ等のような物理的システムが挙げられる。モデルは、モデル化されるシステムをシミュレートするために実行される場合があり、モデルの実行は、モデルのシミュレーションと呼ばれることもある。

【０００４】

モデリング環境は、宣言型言語を実施することができる。宣言型言語は、計算のロジックを、その制御フローを記述することなく表現する言語である。宣言型言語は、それをプログラミング言語プリミティブのシーケンスとしてどのように達成するのかを記述するのではなく、問題ドメインに関してプログラムが達成しなければならないことを記述することができる。場合によっては、宣言型言語は、変数を一度だけ割り当てる単一割り当てを実施する場合がある。宣言型言語の例としては、マサチューセッツ州ネイティックのマスワークス社のＳｉｍｕｌｉｎｋ（登録商標）モデルベースの設計環境；Ｍｏｄｅｌｉｃａ（登録商標）協会のＭｏｄｅｌｉｃａ（登録商標）モデリング言語；ナショナルインスツルメンツ社のＬＡＢＶＩＥＷ（登録商標）グラフィカルプログラミングシステム；ハードウェア記述言語（ＨＤＬ）；Ｐｒｏｌｏｇ言語；及びＨａｓｋｅｌｌ言語などが挙げられる。モデルのモデル要素及び接続要素の少なくとも一部の挙動は、宣言型言語によって暗黙的に定義された計算上の実施形態を含む場合がある。

【０００５】

グラフィカルモデルは、複数の階層レベルを含む場合がある。例えば、モデル要素の種々のグループは、モデル内に階層を確立するモデルコンポーネントとして編成される場合がある。コンポーネントの例としては、サブシステム、サブモデル、サブ仮想機器（サブＶＩ）等が挙げられる。第１の階層レベルでは、モデル要素のグループが、サブシステムブロック又はモデル参照ブロックのような単一のブロックによって表される場合がある。コンポーネントは、それ自体が他のコンポーネントを含む場合があり、モデル内に複数の階層レベルを確立することができる。サブシステムのようなコンポーネントは、ライブラリに保存される場合があり、モデル内の他の場所や、他のモデルで再利用される場合がある。サブシステムの実行挙動は、コンテキストに依存する場合がある。すなわち、データ型、データ次元、及びサンプル時間といった、コンポーネントに含まれるモデル要素のグループのパラメータの少なくとも一部は、未定義であってもよい。これらのパラメータの値は、サブシステムが実行されるモデルから継承される場合がある。一部の実施形態では、あるサブシステムのモデル要素のサブセットの実行が、モデルの他のモデル要素の実行と交互に行われる場合がある。実施形態によっては、サブシステムのモデル要素のサブセットは、原子的に実行される場合がある。さらに、一部の実施形態では、サブシステムは、条件付き実行のために構成される場合があり、サブシステムは、条件が満たされたときに実行される場合がある。条件付きで実行されるサブシステムの例としては、トリガー型、イネーブル型、アクション・イテレーター型、トリガー・イネーブル型、及びファンクションコール型のものがある。

【０００６】

サブモデルは、モデル要素のグループを含む場合もあり、所与の階層レベルに、単一のモデル参照ブロックによって表される場合がある。サブモデルは、ライブラリに保存され

る場合があり、モデル内の他の場所で、または他のモデルで、再利用されることができる。サブモデルの実行挙動を、サブモデルが配置されたモデルから独立させることができ、サブモデルのモデル要素を単位として実行することができる。

【０００７】

動的モデルは、１以上の速度で実行されることができる。例えば、モデルは、第１の速度で実行される第１の部分と、第１の速度よりも速いことがある第２の速度で実行される第２の部分とを含む場合がある。第１の部分のモデル要素は、第１のタスクで実行される一方、第２の部分のモデル要素は、第２のタスクで実行される場合があり、第２のタスクには、第１のタスクよりも高い優先度が割り当てられる場合がある。

【０００８】

時には、モデルが設計された後に、ターゲットプラットフォーム上のグラフィカルモデルの一部又は全部から、コードを実施することが望ましい場合がある。例えば、ユーザは、ワークステーション上でグラフィカルモデルを開発する場合がある。モデルが完成し、システムの挙動を正しくモデル化すると、ユーザーは、そのモデルのためのコードを生成することができる。生成されたコードは、１以上のエントリーポイント関数を含む場合があり、生成されたコードの実行は、エントリーポイント関数を呼び出すことによって実施される場合がある。単一速度モデルの場合、生成されたコード内にｓｔｅｐ（）関数が定義される場合がある。ｓｔｅｐ（）関数は、例えばモデル内に指定された速度で、周期的に呼び出される場合がある。複数の速度を有するモデルの場合、例えば各速度について１つ、個別のｓｔｅｐ（）関数が定義される場合がある。

【０００９】

ユーザは、リアルタイムハードウェアプラットフォーム又は環境における場合のように、モデルをターゲット装置上でリアルタイムに実行することを望む場合がある。リアルタイム実行は、種々のターゲット装置により異なる形で実施される場合がある。例えば、ターゲット装置は、リアルタイムオペレーティングシステム（ＲＴＯＳ）を含む場合がある。そのような場合、ターゲットが種々の処理動作をリアルタイムに（例えば、実行中のシステムからのデータを時間的に遅れをとることなく）実施することができるようにするために、モデルは、ターゲット上で所定の時間間隔内に実行されなければならない場合がある。ＲＴＯＳは、ハードウェアクロックを使用して、プログラムをリアルタイムに実行するために必要な計時サービスを提供することができる。生成されたコードは、モデルの種々の部分を実行するタスクを含む場合があり、タスクは、リアルタイムオペレーティングシステムによる実行に備えてスケジューリングされる場合がある。ターゲットによっては、割り込みサービスルーチン（ＩＳＲ）又は他のメカニズムを使用して、リアルタイム実行を実施するものがある。

【００１０】

ユーザは、速度ベースのマルチタスク型モデリングスタイル及びファンクションコールベースのモデリングスタイルのような異なるモデリングスタイルを使用して、グラフィカルモデルを作成することができる。速度ベースのマルチタスク型モデリングスタイルでは、当初設計されたモデルが、異なる周期的速度で実行される種々の部分を含む場合がある。例えば、モデルは、１以上の速度で動作するように構成された、Ｉｎｐｏｒｔブロックや定数ブロック等のような、入力データを提供する種々のモデル要素を含む場合がある。速度遷移ブロックは、異なる速度で動作するモデル部分間のデータ転送を処理するために使用される場合がある。ファンクションコールベースのモデリングスタイルでは、モデルは、ファンクションコールコンポーネント及び／又はモデル要素を含む場合がある。モデリング環境は、モデルを分析し、例えば、モデルの実行中にモデルのエントリポイントをいつ呼び出すか、及び／又はモデルの実行中にモデルのエントリポイントをどのような速度で呼び出すか等を、暗黙的に決定することができる。

【００１１】

モデリングスタイルによっては、モデルのために生成されたコードが実行されることが意図されたターゲット装置のランタイム環境に基づいて選択される場合もある。例えば、

環境が速度ベースの割り込みをサポートしている場合、速度ベースのマルチタスク型モデリングスタイルが使用される場合がある。場合によっては、モデルは、２以上のモデリングスタイルを有する場合がある。

【００１２】

あるモデリングスタイルで作成されたモデルは、別のモデリングスタイルに簡単に変換することができない場合がある。例えば、モデルをあるモデリングスタイルから別のモデリングスタイルに変換するためには、モデル自体に変更を加えなければならない場合がある。速度ベースからファンクションコールベースに変換するためには、モデルの挙動を定義している種々のモデル要素を、条件付きサブシステム又はサブモデルにグループ化しなければならない場合がある。モデルに変更を加えると、一部の製品設計フローに遅延が生じることがある。例えば、一部の設計フローでは、特に、モデルが安全機能や基幹機能を実施するものである場合、モデルは、膨大な数の検証やその他のテストを受ける場合がある。もしそのようなモデルに変更が加えられた場合、変更後のモデルは、検証プロセス全体を受けなければならず、これにより、製品設計に著しい遅延が生じることがある。

【００１３】

モデルは、複数の階層レベルを含む場合があり、各階層レベルは、１以上のモデルコンポーネントによって表され、または、１以上のモデルコンポーネントを含む場合がある。本開示のシステム及び方法は、最も低い階層レベルのモデルコンポーネントから開始して、モデルに含まれる種々のモデルコンポーネントを分析することができる。本システム及び方法は、モデルコンポーネントをその実行可能エンティティに分割することができる。本システム及び方法はさらに、実行可能エンティティのエントリポイント；実行可能エンティティの実行に際して、存在し、適用される制約及び関係；実行可能エンティティによって使用されるサービスのタイプ；並びに、実行可能エンティティのデータ処理要件及びデータ待ち時間スキームを識別することができる。本システム及び方法は、モデルコンポーネントのアダプテーション層におけるエントリポイント、関係、制約、サービス、待ち時間、及びデータ処理要件に関する情報を記憶することができる。この情報は、例えば、モデルアダプタが作成されたときに対象となったモデルコンポーネントを含む親モデルコンポーネントのような、次に高い階層レベルについて生成され、アダプテーション層に記憶された類似の情報と集約、すなわち結合される場合がある。ただし、アダプテーション層は、実行可能エンティティをどのように呼び出すことができるか、または、前記タイプのサービスがどのように提供されるかを、指定も制限もしない。実行可能エンティティが呼び出される態様、及び、前記タイプのサービスが提供される特定の形は、調節可能である。

【００１４】

１以上の階層レベルに、当該１以上の階層レベルのためのアダプテーション層のエントリポイントをエクスポートするインターフェイスが設けられる場合がある。所与のインターフェースは、実行可能エンティティのエントリポイントに対応するアクセスポイントを含む場合がある。ユーザ指定のスケジューリングロジックが、アクセスポイントに関連付けられる場合がある。例えば、ユーザによって選択され、構成された種々のモデル要素は、接続要素によってアクセスポイントにリンクされる場合がある。これらのモデル要素は集合的に、実行可能エンティティの実行をスケジューリングするためのスケジューリングロジックを定義することができる。スケジューリングは、実行可能エンティティの順序を指定する場合がある。アダプテーション層は、ユーザ指定のスケジューリングロジックが、現在の階層レベル及び現在の階層レベルに集約された任意の下位の階層レベルについて決定された関係及び制約を満たすか否かを判定することができる。アダプテーション層は、実行可能エンティティを呼び出す特定の態様を指定するスケジューリングロジックを、エントリポイントにバインドすることができる。例えば、アダプテーション層は、コンポーネントの種々のエントリポイントを、スケジューリングロジックによって実行される種々の実行可能エンティティに結び付けることができる。アダプテーション層は、スケジューリングロジックがさらに、実行可能エンティティによって使用されるタイプのサービス

を提供するか否かを判定することができる。提供しない場合、アダプテーション層ロジックは、例えばモデリング環境と協働して、当該サービスを提供することができる。

【００１５】

エントリポイントを呼び出すために使用されることがあるスケジューリングロジックは、モデルコンポーネントの機能的挙動を変更しない。さらに、スケジューリングロジックは、モデルコンポーネントが当初設計されたときの呼び出しスタイルとは異なる呼び出しスタイルを使用することができる。例えば、アダプテーション層のためのインターフェースのエントリーポイントは、呼び出しスタイルに関して中立である場合がある。アダプテーション層は、関係及び制約を、モデルコンポーネントが当初設計されたときの呼び出しスタイルから、実行可能エンティティのエントリポイントにバインドされたスケジューリングロジックの呼び出しスタイルに変換することができる。したがって、本システム及び方法によれば、モデルコンポーネントをモデル階層の他のレベルで使用することが可能となり、そのモデルコンポーネントが当初設計されたときの呼び出しスタイルとは異なる呼び出しスタイルを使用することが可能となる。

【００１６】

モデルコンポーネント及びアダプテーション層は、親モデルに含まれる場合もあれば、親モデルコンポーネント（これはさらに、モデルに含まれる場合がある）に含まれる場合もある。親モデル又は親モデルコンポーネントの種々の要素は、アダプテーション層インターフェースのアクセスポイントに接続される場合がある。これらの要素は、モデルの実行可能エンティティがいつ呼び出されるかを明示的に定義するユーザー選択機能を表す場合がある。したがって、アダプテーション層によれば、モデルコンポーネントの実行スケジュールのユーザ制御が可能となる。本システム及び方法は、親モデル又は親モデルコンポーネントの要素の機能及び呼び出しスタイルを、モデルコンポーネントのエントリポイントにバインドすることができる。本システム及び方法はさらに、アクセスポイントに接続された親モデル又は親モデルコンポーネントの要素の機能及び呼び出しスタイルが、関係、制約、又は制限に違反していないことをチェックすることができ、したがって、実行可能エンティティの実行を制御することができる。親モデル又は親モデルコンポーネントは、例えばモデリング環境内で実行される場合がある。アダプテーション層は、親モデル又は親モデルコンポーネントとモデルコンポーネントとの間の呼び出しスタイルの変換を管理することができる。アダプテーション層はさらに、モデルコンポーネントとのデータの交換によって、モデルコンポーネントの実行を調整することができる。実行の代わりに、又は実行に加えて、モデルコンポーネント及び実行呼び出しスタイル間のバインディングを含む、親モデル又は親モデルコンポーネントのためのコードが生成される場合がある。

【００１７】

図１は、モデリング環境によって生成され、データ処理システムのディスプレイ上に表示されることがある例示的モデルエディタウィンドウ１００を示す概略図である。モデルエディタウインドウ１００は、ユーザーインターフェース要素及び／又はウインドウ要素（ウィジェット）のような複数のグラフィカルアフォーダンスを含む場合があり、そのうちの少なくとも一部は、とりわけ、モデルを構築し、編集し、実行し、及び保存するために、ユーザによって操作される場合がある。例えば、モデルエディタウィンドウ１００は、メニューバー１０２、ツールバー１０４、及びキャンバス１０６を含む場合がある。メニューバー１０２は、複数のコマンドを含む場合があり、これらのコマンドは、ファイル、編集、眺め、表示等のようなドロップダウンカテゴリに編成される場合がある。ツールバー１０４は、頻繁に使用されるコマンドを実行するための複数のグラフィカルコマンドボタンを含む場合がある。例えば、ツールバー１０４は、とりわけ、新規ボタン１０８、開くボタン１１０、保存ボタン１１２、及び実行ボタン１１４を含む場合がある。

【００１８】

ユーザは、１以上のライブラリからモデル要素タイプを選択することができ、選択されたモデル要素タイプのインスタンスをキャンバス１０６に追加することにより、モデルを

構築し、又は改訂することができる。選択されるモデル要素の少なくとも一部は、グラフィック要素である場合がある。モデルは、時間ベースの部分、すなわち動的部分、状態ベースの部分、データフローベースの部分、制御フロー部分、イベントベースの部分、メッセージベースの部分等といった、異なる実行ドメインにしたがって動作する種々の部分を有する場合がある。例えば、モデルは、図表のようなグラフィカルな状態ベースのコンポーネントを含む場合がある。

【００１９】

実行可能なセマンティクスを有するグラフィカルモデルコンポーネント１１６は、開くことができ、キャンバス１０６上に表示されることができる。モデルコンポーネント１１６は、電源オンオフ（ＰＵＰＤ）機能を実施する場合があり、初期化サブシステム１１８、リセットサブシステム１２０（ラベル「マイリセット」が付されている）、終了サブシステム１２２、及びアルゴリズムモデル部分１２４を含む場合がある。アルゴリズムモデル部分１２４は、自動車の速度に基づいて、推定位置、例えば自動車の推定位置を計算する場合がある。アルゴリズムモデル部分１２４は、複数の相互接続されたモデル要素を含む場合がある。アルゴリズムモデル部分１２４を形成する種々のモデル要素は、アルゴリズムモデル部分１２４を通る２本の独立した例えば未接続の経路１２６及び１２８に沿って配置される場合がある。経路１２６及び１２８は、Ｉｎｐｏｒｔブロック１３０及び１３２、利得ブロック１３４及び１３６、合算ブロック１３８及び１４０、単位遅延ブロック１４２及び１４４、並びにＯｕｔｐｏｒｔブロック１４６及び１４８を、それぞれ含む場合がある。合算ブロック１３８及び１４０、並びに単位遅延ブロック１４２及び１４４は、例えばアキュムレータの形を有する離散積分器の機能を実施する場合がある。アルゴリズムモデル部分１２４のモデル要素のうちの１以上は、内部状態を含む場合がある。例えば、単位遅延ブロック１４２は、内部状態変数ｘを含む場合があり、内部状態変数ｘは、単位遅延ブロック１４２上に状態変数アイコン１５０によってグラフィカルに表示される場合がある。状態変数ｘは、自動車の現在位置又は開始位置を表す場合があり、センサから得られる場合がある。

【００２０】

アルゴリズムモデル部分１２４は、マルチレート動的システムを表す場合があり、速度ベースのマルチタスク型モデリングスタイルを使用して実施される場合がある。具体的には、経路１２６と経路１２８は、異なる速度で実行される場合がある。例えば、経路１２６は、１秒の周期的速度で実行される場合がある一方、経路１２８は、２秒の周期的速度で実行される場合がある。経路１２６のＩｎｐｏｒｔブロック１３０は、１秒のサンプル時間を有するように構成される場合があり、経路１２６の他のモデル要素は、そのサンプル時間を継承する場合がある。経路１２８のＩｎｐｏｒｔブロック１３２は、２秒のサンプル時間を有するように構成される場合があり、経路１２８の他のモデル要素は、そのサンプル時間を継承する場合がある。

【００２１】

初期化サブシステム１１８、マイリセットサブシステム１２０、及び終了サブシステム１２２は、モデルコンポーネント１１６の実行の初期化、リセット、及び終了フェイズの中でそれぞれ実行される場合がある。例えば、サブシステム１１８、１２０、及び１２２は、初期化イベント、リセットイベント及び終了イベントのようなイベントを監視するイベントトリガー型サブシステムである場合がある。サブシステム１１８、１２０、及び１２２は、各イベントの発生に応答して実行されるユーザ指定の機能、例えば明示的動作を含む場合がある。

【００２２】

一部の実施形態では、明示的な初期化、リセット、又は終了の動作は、それらを条件付きで実行されるサブシステムに含めること以外の他の形で、モデルコンポーネント内に定義される場合がある。例えば、明示的な初期化、リセット、又は終了の動作は、条件付きで実行されるサブモデルのような他のコンポーネント又はコンテナを使用して実施される場合がある。一部の実施形態では、明示的な初期化、リセット、又は終了の動作は、サブ

システム、サブモデル、又は、他のコンポーネント又はコンテナを使用することなく定義される場合がある。

【００２３】

分割エンジンは、モデルコンポーネント１１６を分析し、モデルコンポーネント１１６を形成する種々の実行可能エンティティを識別することができる。一部の実施形態では、実行可能エンティティは、サブシステム、サブモデル、又はモデル要素のグループである場合がある。分割エンジンは、モデルコンポーネント１１６の実行グラフを構築し、又は検査することができる。分割エンジンは、モデルコンポーネント１１６の５つのパーティション、すなわち、初期化サブシステム１１８、マイリセットサブシステム１２０、終了サブシステム１２２、経路１２６、及び経路１２８を識別することができる。

【００２４】

一部の実施形態では、例えばモデルエディタウィンドウ１００上に、パーティション凡例ダイアログのようなウィンドウが生成され、表示される場合がある。パーティション凡例ダイアログは、分割エンジンによって識別された実行可能エンティティに関する情報を提供する場合がある。

【００２５】

図２は、一実施形態による、例示的ユーザーインターフェースウィンドウを示す概略図である。例示的ユーザーインターフェースウィンドウは、パーティション凡例ダイアログ２００の形を有する場合がある。パーティション凡例ダイアログ２００に表示される情報は、複数の行及び列を有する表のように配置される場合があり、行と列の交点に、データを含むレコード又はセルが定義される。例えば、パーティション凡例ダイアログ２００は、複数の行２０２ａ～２０２ｅを含む場合があり、各行が、モデルコンポーネント１１６の特定の実行可能エンティティのエントリポイントに対応している。また、パーティション凡例ダイアログ２００は、色列２０４、注釈列２０６、説明列２０８、及び値列２１０を含む場合がある。パーティションは、色分け及び注釈を使用して、モデルコンポーネント１１６上に表される場合がある。様々なパーティションについて、モデルコンポーネント１１６上で使用される特定の色及び注釈が、パーティション凡例ダイアログ２００の色列２０４及び注釈列２０６に含まれる場合がある。説明列２０８は、パーティションの呼び出しスタイルを識別する情報を含む場合があり、値列２１０は、パーティションの呼び出しスタイルの値を示す場合がある。

【００２６】

例えば、行２０２ａは、経路１２６が赤色に着色され、ラベル「Ｄ１」で注釈されており、呼び出しスタイルが離散サンプル時間であり、その値が他のサンプル時間に対して１であることを示している。行２０２ｂは、経路１２８が緑色に着色され、ラベル「Ｄ２」で注釈されており、呼び出しスタイルが離散サンプル時間であり、その相対値が２であることを示している。行２０２ｃは、初期化サブシステム１１８が青色に着色され、ラベル「Ｉｎｉｔ」で注釈されており（初期化サブシステム１１８が開いていた場合）、呼び出しスタイルがトリガー型であり、トリガーの値が［Ｉｎｉｔ］であることを示している。行２０２ｄは、終了サブシステム１２２がオレンジ色に着色され、ラベル「Ｔｅｒｍ」で注釈されており（終端サブシステム１２２が開いていた場合）、呼び出しスタイルがトリガー型であり、トリガーの値が［Ｔｅｒｍ］であることを示している。行２０２ｅは、マイリセットサブシステム１２０が紫色に着色され、ラベル「マイリセット」で注釈されており（マイリセットサブシステム１２０が開かれていた場合）、呼び出しスタイルがトリガー型であり、トリガーの値が［Ｒｅｓｅｔ］であることを示している。

【００２７】

色以外の他のグラフィカルアフォーダンスが使用されてもよいものと理解すべきである。

【００２８】

図３は、モデルエディタウィンドウ１００上で開くことができる例示的親モデル３００を示す概略図である。親モデル３００は、モデル参照ブロック３０２によって親モデル３

００内に表されたモデルコンポーネント１１６を含む。モデル参照ブロック３０２は、グラフィカルアフォーダンスを含み、グラフィカルアフォーダンスは、分割エンジンによってモデルコンポーネント１１６のために決定された種々の調節可能な属性に対応するアクセスポイントを含むアダプテーション層インターフェース３０４の形を有する場合がある。例えば、アダプテーション層インターフェース３０４は、初期化アクセスポイント３０６、マイリセットアクセスポイント３０８、終了アクセスポイント３１０、１ｓ＿Ｒｕｎアクセスポイント３１２、及び２ｓ＿Ｒｕｎアクセスポイント３１４を含む場合がある。これらのアクセスポイントのうちの１以上は、モデルコンポーネント１１６の実行可能エンティティを実行のために呼び出すことができる呼び出し場所として実施される場合がある。アクセスポイントによっては、他の形で実施される場合もある。

【００２９】

一部の実施形態では、アダプテーション層インターフェース３０４は、実行可能エンティティによって使用されるサービスのため、及びモデルコンポーネント１１６とのデータ通信のための、アクセスポイント若しくは他のポート又はエントリポイントをさらに含む場合がある。例えば、アダプテーション層インターフェース３０４は、Ｉｎｐｏｒｔブロック１３０、１３２に対応する２つのデータ入力アクセスポイント３１３、３１５と、Ｏｕｔｐｏｒｔブロック１４６、１４８に対応する２つのデータ出力アクセスポイント３１７、３１９とを含む場合がある。モデルコンポーネントが、定数ブロック、データ記憶メモリ読み取りブロック、又は他のソースブロックからのデータのような他の入力データを使用する場合、そのような入力データのために、アクセスポイントが設けられる場合がある。例えば定数ブロックの値やデータ記憶メモリ読み取りブロックのメモリ位置を変更するために、それらのアクセスポイントに、ロジックが接続される場合がある。

【００３０】

アダプテーション層インターフェースを用いたモデルコンポーネント１１６の表示によれば、アダプテーション層インターフェースを用いないサブシステムやサブモデルブロックのような表示形式と比較して、モデルコンポーネント１１６の第２の表示形式を表すことができる。

【００３１】

親モデル３００は、ファンクションコールベースのモデリングスタイルを実施する場合がある。例えば、親モデル３００は、ファンクションコールを発行するモデル要素を含む場合があり、これらのモデル要素の少なくとも一部を使用して、モデルコンポーネント１１６の種々の実行エンティティの実行をスケジューリングする場合がある。例えば、親モデル３００のモデル要素は、モデル参照ブロック３０２のアダプテーション層インターフェース３０４のアクセスポイント３０６～３１４と、例えばグラフィカルに又は論理的に、関連付けられる場合がある。具体的には、親モデル３００は、親モデル３００の実行中に電力信号３１８及びリセット信号３２０を供給する信号ビルダーブロック３１６と、信号ビルダーブロック３１６から電力信号３１８及びリセット信号３２０を受信する状態図（スケジューラ）３２２とを含む。スケジューラ状態図３２２は、親モデル３００の実行中に複数のファンクションコールを発行することができる。例えば、スケジューラ状態図３２２は、ＳｔａｒｔＵｐ（）ファンクションコール３２４、Ｒｅｓｅｔ（）ファンクションコール３２６、ＳｈｕｔＤｏｗｎ（）ファンクションコール３２８、及びＲｕｎ（）ファンクションコール３３０を発行することができる。親モデル３００は、Ｒｕｎ２（）ファンクションコール３３４を発行する関数発生器ブロック３３２をさらに含む場合がある。

【００３２】

スケジューラ状態図３２２のＳｔａｒｔＵｐ（）ファンクションコール３２４は、アダプテーション層インターフェース３０４の初期化アクセスポイント３０６に接続される場合がある。これに応答して、マッピングエンジンは、ＳｔａｒｔＵｐ（）ファンクションコール３２４を、初期化イベントにバインドすることができる。スケジューラ状態図３２２のＲｅｓｅｔ（）ファンクションコール３２６は、アダプテーション層インターフェー

ス３０４のマイリセットアクセスポイント３０８に接続される場合がある。これに応答して、マッピングエンジンは、Ｒｅｓｅｔ（）ファンクションコール３２６を、リセットイベントにバインドすることができる。スケジューラ状態図３２２のＳｈｕｔＤｏｗｎ（）ファンクションコール３２８は、アダプテーション層インターフェース３０４の終了アクセスポイント３１０に接続される場合がある。これに応答して、マッピングエンジンは、ＳｈｕｔＤｏｗｎ（）ファンクションコールを、終了イベントにバインドすることができる。スケジューラ状態図３２０のＲｕｎ（）ファンクションコール３３０は、アダプテーション層インターフェース３０４の１ｓ＿Ｒｕｎアクセスポイント３１２に接続される場合がある。これに応答して、マッピングエンジンは、Ｒｕｎ（）関数コール３３０を、アルゴリズムモデル部分１２４の１秒の離散サンプル時間にバインドすることができる。関数発生器ブロック３３２のＲｕｎ２（）ファンクションコール３３２は、アダプテーション層インターフェース３０４の２ｓ＿Ｒｕｎアクセスポイント３１４に接続される場合がある。これに応答して、マッピングエンジンは、Ｒｕｎ２（）ファンクションコール３３４を、アルゴリズムモデル部分１２４の２秒の離散サンプル時間にバインドすることができる。

【００３３】

親モデル３００は、２つの定数ブロック３３６及び３３８をさらに含む場合があり、これらは、入力１アクセスポイント３１３及び入力２アクセスポイント３１５に接続される場合がある。親モデル３００は、２つのＯｕｔｐｏｒｔブロック３４０及び３４２をさらに含む場合があり、これらは、出力１アクセスポイント３１７及び出力２アクセスポイント３１９に接続される場合がある。

【００３４】

親モデル３００は、実行されることができる。実行中、親モデル３００の種々のモデル要素によってファンクションコールが発行される。これらのファンクションコールは、モデルコンポーネント１１６の実行可能エンティティの実行を呼び出すために使用される。代替的に又は追加的に、親モデル３００のためのコードが生成される場合がある。生成されたコードは、アダプテーション層を表すコードを含む場合があり、アダプテーション層では、親モデル３００のファンクションコールを使用して、モデルコンポーネント１１６の実行可能エンティティの実行を呼び出すことができる。なお、アダプテーション層が当初、例えば「１」に設定された定数ブロックに対してアクセスポイントを公開し、その後、その定数ブロックの値が、例えばそのアクセスポイントに接続されたロジックによって変更された場合、生成されたコードは、その定数ブロックの新たな値を含む場合がある。

【００３５】

モデルコンポーネント１１６（図１）は当初、速度ベースのマルチタスク型モデリングスタイルを使用して設計されたにも関わらず、親モデル３００は、ファンクションコールベースのモデリングスタイルを使用して、モデルコンポーネント１１６の実行を呼び出す。それにもかかわらず、異なるモデリングスタイルを有するモデル内でモデルコンポーネント１１６の再利用を可能にするために、モデルコンポーネント１１６に対して、そのアルゴリズム挙動に影響を及ぼす変更は加えられなかった。なお、モデルコンポーネントのためのアダプテーション層を作成することによって、例えばユーザは、ロジック又は機能を例えば種々のモデル要素の形でアダプテーション層インターフェースに接続することにより、モデルコンポーネントの実行を明示的に定義することができる。このように、ユーザは、モデルコンポーネントを実行するシミュレーションに暗黙的に頼るのではなく、モデルコンポーネントの実行を制御することができる。モデルコンポーネントへのアダプテーション層インターフェイスの追加により、プラットフォームに依存しないモデルコンポーネントのバージョンを得ることができる。より高い階層レベルから例えばモデル要素のようなロジックをアダプテーションインターフェース層に接続することにより、モデルコンポーネントに対して、リアルタイム環境のような特定のプラットフォーム環境を指定することができる。さらに、モデルコンポーネントの実行可能エンティティにより使用されるサービスを、例えばモデルに含まれるロジックによって、モデル化することができる。

なお、モデルコンポーネントによって使用される入力データは、例えばモデルに含まれるロジックによって、定数ブロックまたは他のソースブロック等を通して変更される場合がある。

【００３６】

図４は、一実施形態による、例示的モデリング環境４００を示す部分概略図である。モデリング環境４００は、ユーザーインターフェース（ＵＩ）エンジン４０２、モデルエディタ４０４、コード発生器４０６、コンパイラ４０８、シミュレーションエンジン４１０、及びモデルアダプタ４１２を含む場合がある。ＵＩエンジン４０２は、ワークステーション、ラップトップ、タブレット、又は他のデータ処理装置のディスプレイ上に、グラフィカルユーザーインターフェース（ＧＵＩ）及び／又はコマンドラインインターフェース（ＣＬＩ）のような１以上のユーザーインターフェース（ＵＩ）を生成し、表示することができる。ＧＵＩ及びＣＬＩによれば、モデル編集ウィンドウ１００のような、モデリング環境４００に対するユーザーインターフェースを得ることができる。モデルエディタ４０４は、ユーザ入力に応答して又はプログラムによって、開く、作成、編集、及び保存のような選択された動作をモデルに対して実施することができる。

【００３７】

シミュレーションエンジン４１０は、インタープリター４１４、モデルコンパイラ４１６、及び、ソルバー４１８ａ～４１８ｃのような１以上のソルバーを含む場合がある。モデルコンパイラ４１６は、ＩＲビルダー４２０のような１以上の中間表現（ＩＲ）ビルダーを含む場合がある。一部の実施形態では、１以上のＩＲビルダーが、ソルバー４１８に含まれ、またはソルバー４１８に関連付けられる場合がある。また、コード発生器４０６は、最適化器４２１を含む場合がある。

【００３８】

モデルアダプタ４１２は、分割エンジン４２２、アダプテーション層ビルダー４２４、マッピングエンジン４２６、集約エンジン４２８、インターリーブエンジン４３０、及びデータスケジューリング調整エンジン４３２を含む場合がある。

【００３９】

モデリング環境４００は、親モデル３００（図３）及び／又はモデルコンポーネント１１６（図２）のようなコンピュータベースの実行可能なグラフィカルモデル又はモデルコンポーネントを取得することができる。シミュレーションエンジン４１０は、１以上のソルバー４１８ａ～４１８ｃを使用してモデルを実行することができ、例えば、モデルをコンパイル及び実行し、又は逐次解釈実行することができる。ソルバーの例としては、数値積分技術を利用することができる１以上の固定ステップ連続時間ソルバー、及び、例えばＲｕｎｇｅ－Ｋｕｔｔａ法とＤｏｒｍａｎｄ－Ｐｒｉｎｃｅ法のペアを利用した１以上の可変ステップソルバーが挙げられる。固定ステップソルバーを用いる場合、ステップサイズは、モデルのシミュレーション全体を通じて一定のままである。可変ステップソルバーを用いる場合は、ステップサイズは、例えば誤差許容値を満たすように、ステップごとに変えることができる。適当なソルバーに関する非網羅的説明は、マスワークス社のＳｉｍｕｌｉｎｋ（登録商標）ユーザーズガイドにある（２０１６年３月）。

【００４０】

コード発生器４０６は、モデル３００のためのコード４３４のようなコードを生成することができる。例えば、ユーザーインターフェースエンジン４０２は、ユーザが選択することができるコード生成ボタンをＧＵＩ内に提供し、若しくはサポートすることができ、あるいは、ユーザーインターフェースエンジン４０２は、ユーザが例えばＧＵＩまたはＣＵＩに入力したコード生成コマンドを受け取ることができる。また、コード生成コマンドは、例えば特定のイベントが発生したときに、プログラムによって呼び出される場合がある。コード生成コマンドが駆動されたことに応答して、コード発生器４０６は、モデル３００のためのコード４３４のようなコードを生成することができる。生成されたコード４３４の挙動は、実行可能モデル３００の挙動と機能的に同等である場合がある。

【００４１】

生成されたコード４３４は、テキストソースコードのようなテキストコードである場合があり、テキストコードは、例えばコンパイラ４０８によってコンパイルされ、ターゲットマシン又は装置上で実行される場合がある。ターゲットマシン又は装置は、モデリング環境及び／又はシミュレーションエンジンを含まない場合がある。生成されたコード４３４は、Ａｄａ、Ｂａｓｉｃ、Ｃ、Ｃ＋＋、Ｃ＃、ＳｙｓｔｅｍＣ、ＦＯＲＴＲＡＮ、ＶＨＤＬ、Ｖｅｒｉｌｏｇ、ザイリンクスＦＰＧＡライブラリのようなベンダ又はターゲットに固有のＨＤＬコード、アセンブリコード等といった１以上のプログラミング言語に準拠する場合がある。生成されたコード４３４は、ヘッダファイル、ｍａｉｎファイル、ｍａｋｅファイル、及び他のソースファイルを含む場合がある。コンパイラ４０８は、生成されたコードを、マイクロプロセッサやデジタル信号プロセッサ（ＤＳＰ）のようなターゲットハードウェアによる実行に備えて、コンパイルすることができる。一部の実施形態では、生成されたコード４３４は、ハードウェア合成ツールチェーンによって取得される場合があり、ハードウェア合成ツールチェーンは、生成されたコード４３４から、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）やシステムオンチップ（ＳｏＣ）のようなプログラマブルハードウェアデバイスを構成することができる。モデル３００及び生成されたコード４３４は、メモリに記憶される場合があり、例えばデータ処理装置のハードドライブやフラッシュメモリのような不揮発性メモリに記憶される場合がある。モデリング環境４００は、データ処理装置のメインメモリにロードされ、データ処理装置のメインメモリから実行される場合がある。

【００４２】

一部の実施形態では、コード発生器４０６及び／又はコンパイラ４０８は、独立したアプリケーションプログラムのように、モデリング環境４００から独立している場合がある。コード発生器４０６及び／又はコンパイラ４０８は、モデリング環境４００を実行するデータ処理装置とは異なるデータ処理装置上で実行される場合がある。そのような実施形態では、コード発生器４０６は、例えばメモリからモデル３００を取得し、例えばモデルリング環境４００との間で情報をやりとりすることなく、モデル３００のためのコード４３４を生成する場合がある。

【００４３】

一部の実施形態では、ユーザーインターフェースエンジン４０２、モデルエディタ４０４、コード発生器４０６、コンパイラ４０８、シミュレーションエンジン４１０、及びモデルアダプタ４１２のうちの１以上は、本明細書に記載された方法を実施するプログラム命令を含む１以上のソフトウェアモジュール又はライブラリを通して実施される場合がある。ソフトウェアモジュールは、ワークステーション、サーバ、又は他のデータ処理マシン又は装置のメインメモリ、不揮発性メモリ及び／又はコンピュータ読み取り可能媒体のようなメモリに記憶され、１以上のプロセッサによって実行される場合がある。これらのプログラム命令を記憶し、実行するために、光、磁気、又は光磁気媒体を含む非一時的コンピュータ読み取り可能媒体のような他のコンピュータ読み取り可能媒体が使用される場合がある。一部の実施形態では、ユーザーインターフェースエンジン４０２、モデルエディタ４０４、コード発生器４０６、コンパイラ４０８、シミュレーションエンジン４１０、及びモデルアダプタ４１２のうちの１以上は、順序論理回路を生成するように構成及び配置されたハードウェアレジスタ及び組み合わせ論理回路を含む場合がある。一部の実施形態では、記載した方法を実施するために、ファームウェアのような、ソフトウェアとハードウェアの種々の組み合わせが使用される場合がある。

【００４４】

モデリング環境４００は、モデル又はモデルコンポーネントの種々の部分の計算上の実施形態を、それらのモデル部分に含まれるモデル要素のセマンティクスに基づいて作成することができる。計算上の実施形態は、モデル部分に関して設計上指定された挙動を有する場合がある。モデルアダプタ４１６は、計算上の実施形態のエントリポイントを識別し、エントリポイントのグラフィカルアフォーダンスを表示することができる。これらのグラフィカルアフォーダンスには、例えばモデル要素のようなユーザ指定のロジックを接続

することができる。ユーザ指定のロジックは、計算上の実施形態の実行を明示的に呼び出すことができる。なお、ユーザ指定のロジックは、モデル部分の呼び出しスタイルとは異なる呼び出しスタイルを実施することができる。

【００４５】

図５Ａ及び図５Ｂは、一実施形態による、アダプテーション層を作成するための例示的方法のフロー図を示す部分図である。分割エンジン４２２は、ステップ５０２に示されるように、モデルコンポーネントを分析し、モデルコンポーネントをその実行可能エンティティに分割することができる。例えば、分割エンジン４２２は、モデルコンポーネントの依存関係グラフを構築し、又は取得することができる。依存関係グラフは、モデルコンポーネントに含まれる動作又は手順、並びに／あるいは、それらの動作又は手順間のデータ及び／又は制御の依存関係に関する情報を含む場合がある。

【００４６】

図６は、分割エンジン４２２によって、あるモデルコンポーネントのために構築（又は取得）される場合がある例示的依存関係グラフ６００を示す概略図である。依存関係グラフ６００は、モデルコンポーネントの種々の実行可能エンティティに対応する複数の部分グラフを含む場合がある。例えば、依存関係グラフ６００は、初期化部分グラフ６０２、リセット部分グラフ６０４、終了部分グラフ６０６、及び、２つのタスクベースの部分グラフ６０８、６１０を含む場合がある。各部分グラフは、１以上のエントリポイントを含む場合がある。例えば、初期化部分グラフ６０２は、イベントベースであってもよく、初期化イベントのエントリポイント６１２を含む場合がある。リセット部分グラフ６０４は、イベントベースであってもよく、リセットイベントのエントリポイント６１４を含む場合がある。終了部分グラフ６０６は、イベントベースであってもよく、終了イベントのエントリポイント６１６を含む場合がある。第１のタスク部分グラフ６０８は、タスク識別子０（ｔｉｄ０）が割り当てられたタスクによって実行される場合がある。このタスクｔｉｄ０は、例えば２秒ごとに周期的に実行される場合がある。第２のタスク部分グラフ６１０は、タスク識別子１（ｔｉｄ１）が割り当てられたタスクによって実行される場合があり、例えば１秒ごとに周期的に実行される場合がある。これらの部分グラフは、各自の実行可能エンティティの動作を表す１以上のノードをさらに含む場合があり、ノードは、実行フローを表す弧によって相互接続される場合がある。

【００４７】

例えば、初期化部分グラフ６０２は、弧６２４～６２６によって相互接続されたノード６２０～６２３を含む場合がある。リセット部分グラフ６０４は、ノード６２８～６３０及び弧６３２、６３３を含む場合がある。終了部分グラフ６０６は、ノード６３４～６３６及び弧６３８、６３９を含む場合がある。第１のタスク部分グラフ６０８は、ノード６４０～６４５及び弧６４６～６５１を含む場合がある。第２のタスク部分グラフ６１０は、ノード６４６～６５１及び弧６５２～６５６を含む場合がある。

【００４８】

部分グラフによって受け取られ又は生成されたデータは、依存関係グラフ上に表されることができる。例えば、初期化部分グラフ６０２のノード６２２は、破線矢印６５４で示されるように、メモリ、例えばＮＶＲＡＭからデータを読み出し、破線矢印６５６で示されるように、ｓｔｅｐ＿２ｓ部分グラフ６０８のノード６４３にデータを提供することができる。また、リセット部分グラフ６０４のノード６３０は、破線矢印６５７で示されるように、メモリ、例えばＮＶＲＡＭからデータを読み出し、破線矢印６５８で示されるように、第１のタスク部分グラフ６０８のノード６４３にデータを提供することができる。ノード６４３は、破線矢印６６０で示されるように、終了部分グラフ６０６のノード６３６にデータを提供することができる。第１のタスク部分グラフ６０８のノード６４０は、破線矢印６６２で示されるように、依存関係グラフ６００の外部のエンティティからデータ、例えば入力１を受け取ることができる。第２のタスク部分グラフ６１０のノード６４６は、破線矢印６６４で示されるように、依存関係グラフ６００の外部のエンティティからデータ、例えば入力１を受け取ることができる。さらに、第１のタスク部分グラフ６０

８のノード６４５及び第２のタスク部分グラフ６１０のノード６５１は、破線矢印６６６及び６６８で示されるように、依存関係グラフ６００の外部のエンティティにデータを提供することができる。

【００４９】

図６の依存関係グラフ６００は例示を目的とするものであり、他のグラフ及び／又はテキスト表現が構築され、及び／又は評価されてもよいものと理解すべきである。

【００５０】

分割エンジン４２２は、依存関係グラフ６００を調べることによって、モデルコンポーネントの種々の実行可能エンティティを識別することができる。例えば、何れの他の部分グラフとも独立した部分グラフ、例えば、別の部分グラフへの弧を有しない部分グラフは、実行可能エンティティとして識別されることができる。

【００５１】

一部の実施形態では、依存関係グラフ６００は、ＩＲビルダー４２０によって構築される場合がある。

【００５２】

図５に戻ると、分割エンジン４２２は、ステップ５０４に示されるように、例えば依存関係グラフ６００を分析することによって、モデルコンポーネントの実行可能エンティティの調節可能な属性を識別することができる。調節可能な属性には、モデルコンポーネントの実行可能エンティティのエントリポイント、実行可能エンティティによって使用されるサービス、及び実行可能エンティティによって処理される入力データが含まれ得る。また、分割エンジン４２２は、ステップ５０６に示されるように、モデルコンポーネントが当初設計されたときにエントリポイントに関連付けられた呼び出しスタイル、及び、実行可能エンティティによって使用されるサービスのスタイルを識別することができる。分割エンジン４２２は、依存関係グラフ６００を調べることによって、呼び出しスタイルを識別することができる。例えば、依存関係グラフ６００は、部分グラフ６０２、６０４及び６０６に対応する実行可能エンティティは、イベントベースである一方、部分グラフ６０８及び６１０に対応する実行可能エンティティは、周期的タスクベースであることを示している。

【００５３】

分割エンジン４２２は、ステップ５０８に示されるように、実行可能エンティティのエントリポイント間に何らかの関係、制約、又は制限が存在するか否かを判定することができる。例えば、部分グラフ６０８及び６１０に関連するタスクを調べることによって、分割エンジン４２２は、タスクｔｉｄ１は１秒のサンプル時間で実行される一方、タスクｔｉｄ０は２秒のサンプル時間で実行されることを決定する場合がある。分割エンジン４２２は、タスクｔｉｄ１は、タスクｔｉｄ０の２倍の速度で実行される必要があることを結論付けることができる。分割エンジン４２２は、決定された関係、制約、又は制限をメモリに記憶することができる。

【００５４】

関係又は制約の他の例としては、次が挙げられる。

１．タスクＡは、タスクＢが開始（又は完了）する前（又は後）に開始されなければならない。

２．タスクＡとタスクＢは、相互排他的でなければならない。

３．タスクＡは、他のすべてのタスクが開始（又は完了）する前（又は後）に開始（又は終了）されなければならない。

４．第１グループのタスクは、第２グループのタスクが開始（又は完了）する前（又は後）に開始（又は完了）されなければならない。

５．第１グループのタスクと第２グループのタスクは、相互排他的でなければならない。

６．タスクＡは、タスクＢよりも優先することができる。

７．タスクＡは、優先不可である。

８．メモリ読み取りは、所与のタスク内で、又は２以上のタスク間で、書き込みよりも前

に行われなければならない。
９．あらゆる読み込みの後に書き込みが行われなければならない。
【００５５】
また、分割エンジン４２２は、ステップ５１０に示されるように、モデルコンポーネントの実行可能エンティティのスケジュールされた実行と、実行可能エンティティとのデータ通信又は実行可能エンティティ間のデータ通信のスケジュールとの関係を決定することができる。例えば、データスケジューリング調整エンジン４３２は、実行可能エンティティを分析し、実行可能エンティティが、例えば、Ｉｎｐｏｒｔブロック、データ記憶メモリ読み取りブロック、定数ブロック又は他のソースブロックのようなモデル要素のデータ入力によって示されるような入力データを使用することを決定する場合がある。
【００５６】
分割エンジン４２２はさらに、ステップ５１２に示されるように、実行可能エンティティを分析し、実行可能エンティティがサービスを使用する否かを判定することができる（図５Ｂ）。実行可能エンティティによって使用されるサービスの例としては、計時サービス、エラー処理サービス、及びデータ転送サービスが挙げられる。例えば、分割エンジン４２２は、実行可能エンティティが、実行時の現在時刻、又は、例えばデルタタイム値のような最後の実行からの時間を使用することを決定する場合がある。場合によっては、分割エンジン４２２は、実行可能エンティティが、サービスとしてデータを取得し、別の実行可能エンティティと共有されたデータを読み出し及び／又は書き込みすること等を決定する場合がある。
【００５７】
さらに、データスケジューリング調整エンジン４３２は、ステップ５１４に示されるように、実行可能エンティティのうちの１以上について、データ待ち時間スキームを決定することができる。データ待ち時間スキームは、実行可能エンティティの呼び出しからその実行可能エンティティによって使用される入力データの読み出しまでの間の待ち時間を示すことができる。例えば、あるコンポーネントが、単位時間（例えば、秒、ミリ秒、マイクロ秒等）当たり一回実行されるようにスケジューリングされた速度ベースの実行可能エンティティを含み、各実行時に、実行可能エンティティが、新しい入力データを受け取るものと仮定する。データスケジューリング調整エンジン４３２は、実行可能エンティティの呼び出しからその新しい入力データの読み出しまでの間の許容待ち時間を決定し、又は読み出すことができる。また、データスケジューリング調整エンジン４３２は、実行可能エンティティの実行が完了したときと、実行可能エンティティによって生成された出力データが利用可能になるときとの間の許容待ち時間を決定し、又は読み出すことができる。一部の実施形態では、データスケジューリング調整エンジン４３２は、入力データ及び出力データに対して、１以上のデフォルト待ち時間値を使用する場合がある。一部の実施形態では、デフォルト待ち時間値は、ユーザ指定の値で置換される場合がある。例えば、ユーザ設定可能な待ち時間オプションが提供される場合があり、所望の値が、プログラムによって、又はオプション設定ダイアログの１以上のエントリを通じて指定される場合がある。種々の許容可能な待ち時間は、モデルコンポーネントについての待ち時間スキームとして編成される場合があり、待ち時間スキームは、アダプテーション層に含められる場合がある。
【００５８】
一部の実施形態では、実行可能エンティティは、データスケジューリング調整エンジン４３２によって得られる待ち時間挙動を指定することができる。
【００５９】
実行可能エンティティにより許容される実行可能エンティティの実行から入力データの読み出し若しくは出力データの発行までの間の待ち時間が長いほど、実行可能エンティティの実行は、より効率的になる。例えば、待ち時間が長くなるほど、より多くの並列性が使用される場合がある。
【００６０】

アダプテーション層ビルダー４２４は、ステップ５１６に示されるように、モデルコンポーネントのためのアダプテーション層を構築することができる。一部の実施形態では、アダプテーション層は、例えばコマンドに応答してプログラムによって、モデルコンポーネントのために構築される場合がある。例えば、ユーザは、アダプテーション層を構築するためのコマンドをＣＬＩに入力することができる。一部の実施形態では、モデルコンポーネントにプロパティが関連付けられる場合があり、アダプテーション層は、例えばＴｒｕｅ又はＦａｌｓｅのような設定値に基づいて自動的に作成される場合がある。プロパティの値は、プログラムによって指定されてもよいし、あるいは、モデルコンポーネントに関連付けられたオプション設定ダイアログの１以上のエントリによって指定されてもよい。一部の実施形態では、アダプテーション層は、モデルコンポーネントが例えば後の再利用に備えてライブラリに保存される場合に、構築されることがある。アダプテーション層は、モデルコンポーネントの種々の実行可能エンティティのエントリポイントのアクセスポイントを含む場合がある。また、アダプテーション層は、実行可能エンティティの実行に関する制約、関係、又は制限、並びにモデルコンポーネントによって受信された入力データ及びモデルコンポーネントによって生成された出力データについての１以上のデータ待ち時間スキームに関する情報を含む場合がある。

【００６１】

また、一部の実施形態では、アダプテーション層は、計時サービスやデルタタイムサービスのような、実行可能エンティティによって使用される種々のサービスに関する情報を含む場合がある。一部の実施形態では、アダプテーション層は、計時サービスやデルタタイムサービスのような１以上のサービスを、実行可能エンティティに提供することができる。

【００６２】

アダプテーション層ビルダー４２４は、ステップ５１８に示されるように、モデルコンポーネントの第２の表示形式を表示することができる。例えば、アダプテーション層ビルダー４２４は、例えばモデルエディタウインドウのキャンバス上に表示されるようなモデルコンポーネントのグラフィカル表現を有する１以上のアクセスポイントを有するアダプテーション層インターフェースを含む場合がある。アダプテーション層インターフェースなしで表示されたモデルコンポーネントは、モデルコンポーネントの第１の表示形式である場合がある。第１の表示形式のモデルコンポーネントと第２の表示形式のモデルコンポーネントはいずれも、実行可能である場合がある。

【００６３】

終了ステップ５２０に示されるように、その後、処理は完了する場合がある。

【００６４】

図７Ａ～図７Ｃは、一実施形態による、アダプテーション層を使用するための例示的方法を示すフロー図の部分図である。

【００６５】

アダプテーション層が作成されたときに対象となったモデルコンポーネントは、親モデル又は親モデルコンポーネントに含められてもよい。モデルアダプタ４１２は、ステップ７０２に示されるように、アダプテーション層の１以上のアクセスポイントを親モデル又は親モデルコンポーネントの中に公開する指示を受け取る場合がある。この指示は、ユーザが、例えばテキスト又はグラフィックによって指定することができる。例えば、ユーザは、例えばマウスクリック又は他のコマンドを用いて、モデルコンポーネントを選択することができる。これに応答して、モデルコンポーネントに対して使用可能なコマンドのリストを含むダイアログのようなユーザーインターフェースウィンドウが、表示される場合がある。リストに含まれるコマンドの１つは、「アダプテーションインタフェースの構築」コマンドである。「アダプテーションインターフェースの構築」コマンドの選択に応答して、ポップアップダイアログのようなウィンドウが、モデルエディタウィンドウに表示される場合がある。

【００６６】

図８は、一実施形態による、アダプテーションインターフェースを構築する際に使用される例示的ウィンドウ８００を示す概略図である。ウィンドウ８００は、ポップアップダイアログの形を有する場合があり、モデルコンポーネントのアダプテーション層のアクセスポイントに関する種々の指示を受けとるための複数のデータエントリフィールドを含む場合がある。一部の実施形態では、ダイアログ８００は、モデルコンポーネントの各実行可能エンティティについて１以上のエントリを含む場合がある。これらのエントリは、各実行可能エンティティのアクセスポイントを公開すべきか否か、及び、アクセスポイントを公開すべき場合、そのアクセスポイントの呼び出しスタイル、に関するテキスト情報を受け取る場合がある。選択されたアクセスポイントは、現在の階層レベルで公開されてもよいし、何らかの他の階層レベルで公開されてもよく、又は別のモデル場所で公開されてもよい。

【００６７】

例えば、ダイアログ８００は、実行可能エンティティを初期化するためのエントリ８０２を含む場合がある。エントリ８０２は、現在の階層レベルから初期化実行可能エンティティを呼び出す際に使用されるアクセスポイントを公開するか否かの指示を受け取るため、及びその初期化実行可能エンティティを呼び出すために使用される呼び出しスタイルに関する情報を受け取るための、テキストボックス８０４を含む場合がある。一部の実施形態では、テキストボックス８０４は、ドロップダウン矢印８０６を含む場合があり、これが選択された場合、現在の階層レベルから初期化実行可能エンティティを呼び出すために使用可能な呼び出しスタイルのリストが表示される場合がある。表示することができる呼び出しスタイルの例としては、エッジトリガー、ファンクションコール、及び暗黙的タスクが挙げられる。エッジトリガーは、（例えば、０から１へ）立ち上がる及び／又は（例えば、１から０へ）立ち下がる制御信号、イベント又は他のデータを使用した呼び出しスタイルを意味する場合がある。ファンクションコールは、関数を呼び出す（例えば、コールする）呼び出しスタイルを意味する場合がある。タスクという用語は、例えば時間ベースのような周期性、あるいは例えば初期化、リセット、及び終了のような非周期性を有する、アクセスポイントの属性を意味する場合がある。暗黙的タスクは、アクセスポイントが、次に高い階層レベルの各実行可能エンティティに公開されるべきではないこと、及び、アクセスポイントが、代わりに、同じタスク属性を有する別の実行可能エンティティのアクセスポイントと暗黙的に結合される場合があることを意味する場合がある。他の呼び出しスタイルが表示されてもよいものと理解すべきである。

【００６８】

エントリ８０２は、初期化実行可能エンティティのアクセスポイントを公開するとともに、それをエッジトリガ呼び出しスタイルに関連付けるように構成されている。ダイアログ８００は、リセット実行可能エンティティのエントリ８０８をさらに含む場合があり、エントリ８０８は、ドロップダウン矢印８１２を有するテキストボックス８１０を含む場合がある。エントリ８０８は、リセット実行可能エンティティのアクセスポイントを公開するとともに、それをエッジトリガ呼び出しスタイルに関連付けるように構成されている。ダイアログ８００は、終了実行可能エンティティのエントリ８１４をさらに含む場合があり、エントリ８１４は、ドロップダウン矢印８１８を有するテキストボックス８１６を含む。エントリ８１４は、終了実行可能エンティティのアクセスポイントを公開するとともに、それをエッジトリガ呼び出しスタイルに関連付けるように構成されている。ダイアログ８００は、Ｓｔｅｐ＿１ｓ実行可能エンティティのエントリ８２０をさらに含み、エントリ８２０は、ドロップダウン矢印８２４を有するテキストボックス８２２を含む場合がある。エントリ８２２は、Ｓｔｅｐ＿１ｓ実行可能エンティティのアクセスポイントを公開するとともに、それをファンクションコール呼び出しスタイルに関連付けるように構成されている。ダイアログ８００は、Ｓｔｅｐ＿２ｓ実行可能エンティティのエントリ８２６を含む場合があり、エントリ８２６は、ドロップダウン矢印８３０を有するテキストボックス８２８を含む。エントリ８２６は、Ｓｔｅｐ＿２ｓ実行可能エンティティのアクセスポイントを公開しないように構成され、代わりに、このアクセスポイントを、親モデ

ル又は親モデルコンポーネントの１以上のアクセスポイントと集約するように構成されている。

【００６９】

親モデルにおいてエントリーポイントが、例えば暗黙的にエクスポートされた場合、例えばアダプテーション層では、実行頻度及びデータ依存関係情報を分析することによって、実行シーケンスが、自動的に決定される場合がある。例えば、実行可能エンティティは、親モデルにおけるものと同じ実行頻度を有するタスク内で実行される場合がある。タスクの内部で、実行可能エンティティは、データ又は制御の依存関係に基づいて実行される場合があり、例えば、実行可能エンティティは、その実行可能エンティティにデータを提供するモデル要素の後、及び、その実行可能エンティティによって計算された結果を使用するモデル要素の前に実行される場合がある。

【００７０】

一部の実施形態では、実行可能エンティティは、ファンクションコール又はメッセージのような１以上の明示的イベント、あるいは、トリガ信号又は周期的タイマーのような１以上の暗黙的イベントによってトリガー駆動される場合がある。

【００７１】

図７Ａに戻り、アダプテーション層ビルダー４２４は、ステップ７０４に示されるように、親モデル又は親モデルコンポーネントに含まれるモデルコンポーネントのためのアダプテーション層インターフェースを構築することができる。アダプテーションインターフェースは、例えば初期化実行可能エンティティ、リセット実行可能エンティティ、終了実行可能エンティティ、及びＳｔｅｐ＿１ｓ実行可能エンティティのためのアクセスポイント（図８参照）のような選択されたアクセスポイントを、公開することができる。集約エンジン４２８は、ステップ７０６に示されるように、集約に備えてマークされた種々のアクセスポイントを集約することができる。例えば、これらのアクセスポイントは、モデル階層のより高いレベルのアクセスポイントと結合され、より高い階層レベルで公開される場合がある。

【００７２】

モデルアダプタ４１２は、ステップ７０８に示されるように、親モデル又は親モデルコンポーネントからアダプテーションインターフェースの公開されたアクセスポイントへの接続が行われたことを検出することができる。接続が検出された場合、マッピングエンジン４２６は、ステップ７１０に示されるように、親モデル又は親モデルコンポーネント（ユーザが指定してもよい）で使用される機能及び呼び出しスタイルを、各実行可能エンティティのエントリポイントにバインドすることができる。ループバック矢印７１２で示されるように、接続の検出、並びに、機能及び呼び出しスタイルのバインドは、他の公開されたアクセスポイントに接続が行われたときに繰り返される場合がある。モデルアダプタ４１２は、例えば親モデル又は親モデルコンポーネントの種々のモデル要素によって表されるような、親モデル又は親モデルコンポーネントに含まれるロジックを分析し、モデルコンポーネントの実行可能エンティティの実行を呼び出すことができる。

【００７３】

アダプテーション層の調節可能な属性は、公開された調節可能な属性に接続されたスケジューリングロジックその他のロジック以外の方法で設定又は構成されてもよいし、あるいは、そのようなロジックを含む他の方法で設定又は構成されてもよい。

【００７４】

一部の実施形態では、アダプテーション層ビルダー４２４（又はＩＲビルダー４２０）は、モデルコンポーネントを含む親モデル又は親モデルコンポーネントのためのコールグラフを構築することができる。コールグラフは、親モデル又は親モデルコンポーネントとモデルコンポーネントとの間の呼び出し関係を表すことができる。コールグラフは、親モデル又は親モデルコンポーネントがモデルコンポーネントの実行可能エンティティを呼び出す態様を示している。コールグラフは、親モデル又は親モデルコンポーネントがモデルコンポーネントの実行可能エンティティを呼び出すために使用する特定の呼び出しスタイ

ルを示す場合がある。例えば、コールグラフは、親モデル又は親モデルコンポーネントが、ファンクションコールを使用して第１の実行可能エンティティを呼び出し、トリガーを使用して第２の実行可能エンティティを呼び出すことを示す場合がある。

【００７５】

モデルアダプタ４１２は、ステップ７１６（図７Ｂ）に示されるように、親モデル又は親モデルコンポーネントのロジックによる実行可能エンティティのスケジューリングと、モデルコンポーネントの実行可能エンティティの実行のために決定された関係、制約、又は制限との間に、矛盾が存在するか否かを判定することができる。矛盾が検出された場合、モデルアダプタ４１２は、ステップ７１８に示されるように、エラー又は警告を発行することができる。例えば、ＵＩエンジン４０２は、モデルエディタウィンドウ１００上に、エラー又は警告メッセージを表示する場合がある。

【００７６】

アダプテーション層は、ステップ７２０に示されるように、モデルコンポーネントの種々の実行可能エンティティを実行するためのスケジュールを生成することができる。スケジュールを決定するために、アダプテーション層は、

１．親モデル又は親モデルコンポーネントによって実施される呼び出しスタイル（複数可）、及びモデルコンポーネントの実行可能エンティティの呼び出しスタイル（複数可）；及び

２．モデルコンポーネントの実行可能エンティティの実行を呼び出す、親モデル又は親モデルコンポーネントに含まれるスケジューリングロジック

を考慮することができる。

【００７７】

アダプテーション層によって生成されたスケジュールは、モデルコンポーネントの呼び出しスタイルとは対照的に、親モデル又は親モデルコンポーネントの呼び出しスタイルに適用される場合がある。例えば、親モデルがファンクションコール呼び出しスタイルを実施し、モデルコンポーネントが速度ベースのマルチタスク型呼び出しスタイルを実施している場合、アダプテーション層によって生成されたスケジュールは、種々のファンクションコールをスケジューリングすることができる。アダプテーション層は、親モデルのファンクションコールに応答して、速度ベースのマルチタスク型呼び出しスタイル用に当初定義されたモデルコンポーネントの実行可能エンティティの呼び出しも処理する。

【００７８】

アダプテーション層は、ステップ７２２に示されるように、実行可能エンティティによって使用されるサービスが、例えば現在の階層レベルから、コンポーネントに提供されるか否かを判定することができる。例えば、モデルコンポーネントが、実行中に計時サービスを使用すると仮定する。さらに、モデルコンポーネントは、モデルコンポーネントがリアルタイムプラットフォーム又は環境上でどのように実行されるかをシミュレートする形で実行されているものと仮定する。ユーザは、リアルタイムプラットフォーム又は環境によって計時サービスをモデル化するロジックを有する場合がある。例えば、ユーザは、リアルタイムプラットフォーム又はリアルタイムの計時サービスをシミュレートするように構成されたカウンタ又はタイマブロックを、現在の階層レベルに有する場合がある。アダプテーション層は、カウンタ又はタイマブロックが、実行可能エンティティによって使用される計時サービスをシミュレートするものと判定することができる。

【００７９】

実行可能エンティティによって使用される１以上のサービスが、例えば現在の階層レベルに又は別の階層レベル若しくはモデル位置に、明示的に定義され又は提供されていない限り、アダプテーション層は、ステップ７２４に示されるように、実行可能エンティティのための種々のサービスを提供することができる（図７Ｃ）。一部の実施形態では、アダプテーション層は、モデリング環境４００と協働して、種々のサービスを提供することができる。例えば、シミュレーションエンジン４１０は、計時サービス、デルタタイムサービス、及びエラー処理サービスを含む場合がある。アダプテーション層は、それらのサー

ビスを使用する実行可能エンティティへのそれらのサービスの提供を調整することができる。一部の実施形態では、アダプテーション層は、ステップ７２６に示されるように、明示的に定義され、又は暗黙的に生成されたサービスのスタイルを、実行可能エンティティによって必要とされるスタイルに変換することができる。

【００８０】

ステップ７２８に示されるように、データスケジューリング調整エンジン４３２は、親モデル又は親モデルコンポーネントとモデルコンポーネントとの間、あるいは、モデルコンポーネントの種々の実行可能エンティティ間でデータを交換するためのスケジュールを決定することができ、スケジュールは、モデルコンポーネントの実行可能エンティティを実行するためのスケジュールとの間で調整される。データスケジューリング調整エンジン４３２は、アダプテーション層から待ち時間スキームを読み出し、それを、実行可能エンティティの実行を呼び出す親モデル又はモデルコンポーネントのロジックに対して評価することができる。データスケジューリング調整エンジン４３２は、データ交換が、実行可能エンティティの実行との間で調整され、指定された待ち時間スキームを満たすようにするために、入力データをモデルコンポーネントにいつ提供する必要があるか、及び出力データをモデルコンポーネントからいつ読み出す必要があるかを、決定することができる。

【００８１】

一部の実施形態では、モデルコンポーネントによって使用される入力データ、及び、モデルコンポーネントによって生成された出力データを記憶する種々のモデルコンポーネントのために、データバッファが確立される場合がある。モデルコンポーネントがデータ転送をサービスとして要求した場合、モデルコンポーネントそれ自体は、データバッファを確立できない場合がある。代わりに、種々の要求データ転送サービスが、説明したように、現在の階層レベルに提供される場合がある。

【００８２】

モデルコンポーネントの実行可能エンティティは、種々のアクセスメソッドを使用して、バッファから入力データを獲得し、出力データをバッファに記憶することができる。データスケジューリング調整エンジン４３２は、そのようなバッファとの間の読み出し及び書き込みのスケジュールを確立することができる。スケジュールは、モデルコンポーネントの実行可能エンティティの実行スケジュールと同期させることができ、モデルコンポーネントの入力データ転送及び出力データ転送のために確立された待ち時間スキームを満たすことができる。モデルコンポーネントとの間のデータ転送のスケジューリングは、モデルコンポーネントのモデル要素間のデータ依存関係の関数であってもよい。例えば要求されたデータ転送サービスを提供するために、データ転送スケジュールは、アダプテーション層に提供され、アダプテーション層は、データ転送スケジュールにしたがってアクセスメソッドを呼び出す場合がある。

【００８３】

全てのデータ転送情報が使用可能である場合、コンパイル中にモデルコンパイラ４１６によって、データバッファが割り当てられる場合がある。モデルコンパイラ４１６がコンパイル中にデータバッファを割り当てるための十分な情報を欠いている場合（例えば、データ転送の挙動が不明であることから）、例えば初期化フェイズ中は、割当てが、延期される場合がある。一部の実施形態では、データバッファの割り当ては、モデルの実行中に実施される場合がある。さらに、アダプテーション層は、データバッファにアクセスするためのインターフェースを提供する場合がある。

【００８４】

シミュレーションエンジン４１０は、ステップ７３０に示されるように、モデルコンポーネントを含む親モデル又は親モデルコンポーネントを実行することができる。

【００８５】

モデル実行

シミュレーションエンジン４１０は、モデルコンポーネントが当初設計されたときの呼び出しスタイルとは異なる呼び出しスタイルを使用してそのモデルコンポーネントを呼び

出す、モデルについての種々の実行命令を生成することができる。実行命令の生成及びモデルの実行は、初期化フェイズ、シミュレーション（又は実行）フェイズ、０、１若しくは２以上のリセットフェイズ、及び終了フェイズのような複数のフェイズを含む場合がある。モデルの実行には、入力データを受け取り、処理すること、及び出力データを生成することが含まれる場合がある。一部の実施形態では、モデルの実行は、例えばシミュレーション時間のようなある期間にわたって実行される場合があり、この期間は、ユーザ指定であってもよいし、機械指定であってもよい。シミュレーション時間は、論理的実行時間であり、シミュレーション開始時刻から始まり、シミュレーション終了時刻に終了する場合がある。シミュレーション開始時刻とシミュレーション終了時刻との間の一連の時点（これらの時点は、シミュレーション時間ステップと呼ばれることがある）で、モデルの種々のモデル要素の状態、入力及び出力が計算される場合がある。時間ステップのサイズは、固定されていても変化されてもよく、モデルの実行の際に選択され、使用される特定のソルバー４１８によって決定される場合がある。シミュレーション時間の全体にわたって、実行命令は、シミュレーションエンジン４１０によって生成される場合がある。

【００８６】

初期化フェイズ

初期化フェイズは、モデルの実行の開始を特徴づけることができ、種々の命令をコンパイルし、リンクすることを含む場合がある。初期化フェイズは、データ構造を準備し、パラメータを評価すること、ブロック特性を構成し、伝搬すること、ブロック接続を決定すること、及び、ブロック削減やブロック挿入を実施することを含む場合がある。データ構造の準備及びパラメータの評価の結果として、初期化フェイズで使用するための１以上のデータ構造の作成及び初期化がもたらされる場合がある。モデルの種々のブロックについて、あるメソッドは、それらのブロックに、ブロックのすべてのパラメータを強制的に評価させることができ、このメソッドは、モデルの各ブロックについて呼び出される場合がある。未解決のパラメータがある場合、実行エラー及び／又はコンパイルエラーが投じられる場合がある。ブロック及びポート／信号特性の構成及び伝搬中に、各ブロック及び／又はポート／信号のコンパイル済属性（データ次元、データ型、複雑度、サンプルモード、及びサンプル時間など）が、対応する挙動、並びに、所与のブロック及び／又はポート／信号に接続されたブロック及び／又はポート／信号の属性に基づいて、設定される場合がある。この属性設定は、あるプロセスを通して実施される場合があり、当該プロセスの中で、あるブロックから次に続く信号、データフロー、制御、状態、物理的、メッセージ又は他の接続性へと、ブロックの挙動は、モデル全体に「波及」する。

【００８７】

このプロセスは、推論と呼ばれることがあり、その一実施形態が、伝搬である。ブロック（又はポート）挙動が明示的に指定されたブロックの場合、伝搬は、ブロック（又はポート）の属性が、それに接続されたブロック（又はポート）の属性と互換性を有することを確保するのに役立つ。もし互換性が無ければ、シミュレーションエンジン１３１０によって警告又はエラーが発行され、ユーザーインターフェースエンジン４０２によってユーザに対して表示される。多くのブロック（又はポート）は、幅広い属性と互換性を有するように実施されることができる。そのようなブロック（又はポート）は、自分自身の挙動を、自分自身に接続されたブロック（又はポート）の属性にしたがって、適合させることができる。ブロックの正確な実施形態は、そのブロックが配置されるモデルに基づいて選択される場合がある。このステップには、すべての速度遷移が決定論的結果をもたらすこと、及び、適当な速度遷移ブロック及び／又は遅延ブロックが使用されていることを検証すること、のような他の態様が含まれ得る。

【００８８】

初期化フェイズは、実際のブロック接続を決定することもできる。例えば、仮想ブロックは、モデルの実行において意味役割を何も演じない場合がある。このステップでは、仮想ブロックは、最適化によって消去され、例えば除去される場合があり、残りの非仮想ブロックが、互いに適当に再接続される場合がある。実際のブロック接続を有するモデルの

このコンパイル済バージョンは、この時点から、実行プロセスで使用することができる。モデル内でブロックを相互接続する仕方が必ずしも、ブロックメソッドが実行される順序を定義するとは限らない。一部の実施形態では、ブロックメソッドには、出力方程式、微分方程式及び更新方程式が含まれる場合がある。

【００８９】

初期化フェイズ中に、ブロックサンプル時間が決定される場合もある。ブロックのサンプル時間は、例えばそのブロックのサンプル時間パラメータを設定することによって、明示的に設定されてもよいし、あるいは、そのブロックのタイプ又はモデル内のそのブロックのコンテキストに基づいて、暗黙的に決定されてもよい。サンプル時間パラメータは、ベクトル［T_S、T_O］であってもよい。ここで、T_Sはサンプリング周期であり、T_Oは初期時間オフセットである。初期化中に、モデルコンパイラ４１６は、種々のモデル要素のサンプル時間を、明示的サンプル時間を有するモデル要素については、モデル要素のサンプル時間パラメータから決定し、暗黙的サンプル時間を有するモデル要素については、モデル要素タイプから決定する場合があり、あるいは、それらのサンプル時間を継承するモデル要素については、モデルコンテンツから決定する場合がある。コンパイルされたサンプル時間は、シミュレーションの際に、モデル要素のサンプル速度を決定する場合がある。

【００９０】

選択されたソルバー４１８は、モデルのシミュレーションのためのシミュレーション時間ステップのサイズを決定することができ、これらのシミュレーション時間ステップは、モデルのブロックのサンプル時間に対応するように選択される場合がある。シミュレーション時間ステップが、あるブロックのサンプル時間に一致すると、サンプル時間ヒットが発生し、そのブロックは、そのシミュレーションステップ中の実行に備えてスケジューリングされる。

【００９１】

ソルバー４１８ａ～４１８ｃは、モデルの種々のブロックについて実行リストを生成することができる１以上のスケジューラを含み、又は、そのような１以上のスケジューラを取得することができる。具体的には、スケジューラは、ブロックソート済順序リスト及びメソッド実行リストを生成することができる。スケジューラは、ブロックソート済順序リスト及びメソッド実行リストを生成するために、１以上のアルゴリズムを使用する場合がある。例えば、静的スケジューラは、レートモノトニックスケジューリング（ＲＭＳ：Rate Monotonic Scheduling）、ＥＤＦ（EarliestDeadline First）スケジューリング、静的循環スケジューリング、又はデッドラインモノトニックスケジューリング（ＤＭＳ：Deadline Monotonic Scheduling）を使用する場合がある。動的スケジューラは、イベントカレンダを使用して、タイムアウトやアラームのような将来のイベントをスケジューリングする場合がある。

【００９２】

一部の実施形態では、初期化フェイズは、初期化フェイズ中に実行することができるユーザ指定動作をさらに含む場合がある。例えば、初期化イベントに応答して条件付きで実行されるサブシステム中に指定された種々の動作は、初期化フェイズ中に実行される場合がある。

【００９３】

初期化フェイズは、１回発生する場合がある。

【００９４】

実行フェイズ

実行フェイズ又はシミュレーションフェイズは、初期化フェイズの後に続く場合がある。実行フェイズ中に、シミュレーションエンジン４１０は、シミュレーションループ又は反復ループステージに入り、その間、シミュレーションエンジン４１０は、入力を連続的に獲得して、シミュレーション開始時間からシミュレーション終了時間まで、特定のシミュレーション時間ステップでモデル要素の状態及び出力を計算する。各ステップにおいて

、モデル要素の入力、状態、及び出力の新しい値が計算される場合があり、モデル要素は、計算された値を反映するように更新される場合がある。ステップ間の時間の長さは、ステップサイズと呼ばれる。ステップサイズは、モデルを実行するために選択されたソルバー４１８のタイプ、モデルの基本サンプル時間、及びモデルの連続状態がゼロ交差のような不連続性を有するか否かに依存する。シミュレーションループ又は反復ループのステップは、シミュレーション終了時刻に達するまで繰り返される場合がある。シミュレーション時間の最後に、モデル要素の入力、状態、及び出力の最終値が、モデルに反映される。一部の実施形態では、モデルは、予め設定されたシミュレーション終了時刻なしに実行される場合がある。

【００９５】

リセットフェイズ（複数可）

一部の実施形態では、実行フェイズは、０、１又は２以上のリセットフェイズを含む場合がある。リセットフェイズは、システム生成動作を実行することができる暗黙的部分を含む場合がある。リセットフェイズは、ユーザ指定の（明示的）リセット動作を実行することができる明示的部分を含む場合がある。例えば、ユーザ指定の種々の動作が、リセットイベントに応答して条件付きで実行されるサブシステム中に指定される場合がある。一部の実施形態では、リセットフェイズは、明示的リセット動作のみを含む場合がある。

【００９６】

実行フェイズ中に、モデルのためのコードが生成されてもよいし、生成されなくてもよい。コードが生成された場合、モデルは、加速実行モードで実行される場合がある。加速実行モードでは、モデル又はその一部が、ソフトウェアモジュール又はハードウェア記述のいずれかに変換される。このステージが実行された場合、その後に続く種々のステージは、モデルの実行の際に、生成されたコードを使用することができる。コードが生成されなかった場合、モデルは、インタープリティブモードで実行される場合がある。インタープリティブモードでは、シミュレーション時間にわたってモデルを実行するために、モデルのコンパイルされ、リンクされたバージョンが、シミュレーションエンジン４１０によって直接使用される場合がある。モデルのためのコードが生成された場合、必ずしもモデルが実行される必要はない。モデリング環境４００内でモデルを実行する代わりに、生成されたコード４３４がコンパイルされ、コントローラやテストハードウェアのようなターゲット装置上に配備される場合がある。

【００９７】

終了フェイズ

終了フェイズは、実行フェイズの後に続く場合がある。終了フェイズ中に、シミュレーションエンジン４１０は、クリーンアップ動作を実施することができる。例えば、シミュレーションエンジン４１０は、割り当てられたメモリを解放する場合がある。終了フェイズ中に実行される他の動作としては、内部メモリを不揮発性ランダムアクセスメモリ（ＮＶＲＡＭ）のような不揮発性メモリに保存すること、ＡＤＣ及びＤＡＣの電源をオフにすること、アクチュエータをシャットダウンすること、終了フラグを発行すること、データをデータファイルに書き出すこと、及びデータファイルを閉じることが挙げられる。

【００９８】

終了フェイズは、終了フェイズ中に実行することができるユーザ指定の（明示的）終了動作をさらに含む場合がある。例えば、ユーザ指定の種々の動作が、終了イベントに応答して条件付きで実行されるサブシステムに含まれる場合がある。終了フェイズは、１回発生する場合がある。

【００９９】

モデルは、初期化、実行、リセット、及び終了の他に、追加及び／又は他のフェイズを含むことができるものと理解すべきである。例えば、モデルは、とりわけ、ウォームスタートフェイズ、コールドスタートフェイズ、部分リセットフェイズ、ソフトリセットフェイズ、フルリセットフェイズ、ハードリセットフェイズ、正常終了フェイズ、及び緊急終了フェイズを含む場合がある。異なる初期化フェイズ、リセットフェイズ、又は終了フェ

イズ中に、異なる動作が実行される場合がある。例えば、緊急終了フェイズ中に実行することができる唯一の動作は、例えば、電源が失われる前に重要データを保存することである。

【０１００】

コード発生器４０６は、ステップ７３２に示されるように、モデルコンポーネントを含む親モデル又は親モデルコンポーネントのためのコードを生成することができる。生成されたコードは、モデルコンポーネントのアダプテーション層にリンクされたユーザ指定の機能に基づいて、モデルコンポーネントの種々の実行可能エンティティを呼び出すためのスケジュールを実施することができる。生成されたコードは、モデルとモデルコンポーネントとの間のデータの交換によってモデルコンポーネントの実行可能エンティティの呼び出しを調整するコードを含む場合がある。生成されたコードは、異なる呼び出しスタイル、すなわち、親モデル又は親モデルコンポーネントで使用される呼び出しスタイルとモデルコンポーネントで使用される呼び出しスタイルのバインディングのためのコードをさらに含む場合がある。

【０１０１】

コード生成は、コードが実行されることになる環境によって提供される種々のサービスを呼び出すコードを生成することをさらに含む場合がある。例えばモデリング環境のようなシミュレーション環境では、これらのサービスは、シミュレーションエンジン４１０によって提供される場合がある。このようなサービスの例には、データ転送、タイマー、経過時間、及びエラー処理が挙げられる。例えば、リアルタイム環境における配備のために、これらのサービスは、リアルタイムオペレーティングシステム（ＲＴＯＳ）によって、及び／又はハードウェアクロックのようなリアルタイム環境の１以上のハードウェア要素によって、提供される場合がある。

【０１０２】

図９は、一実施形態による、例示的再利用環境９００を示す概略図である。再利用環境９００は、モデルコンポーネント９０２を含む場合があり、モデルコンポーネント９０２は、モデリング環境４００を操作するユーザによって作成されたサブシステム又はサブモデルのようなグラフィカルモデルコンポーネントである場合がある。モデルコンポーネント９０２は、速度ベースのマルチタスク型呼び出しスタイルを使用するように当初作成されたものである場合がある。モデルコンポーネント９０２は、検証テストを受ける場合があり、モデルコンポーネント９０２の検証要件を満たすことが証明されている場合がある。モデルコンポーネント９０２は、モデリング環境４００によって使用されるライブラリに保存される場合がある。

【０１０３】

参照符号９０２’で示されるモデルコンポーネント９０２のインスタンスは、破線矢印９０６で示されるように、モデルコンポーネント９０４（モデルコンポーネント＿２）内に含まれる場合がある。インスタンス９０２’は、モデルコンポーネント９０２のコピーであってもよい。あるいは、インスタンス９０２’は、ライブラリに保存されることがあるモデルコンポーネント９０２へのライブラリリンクのようなリンクを表す場合がある。ライブラリに保存されたリンク先のモデルコンポーネント９０２への変更は、ライブラリリンクを介して、インスタンス９０２’のようなモデルコンポーネントのすべてのインスタンスによって継承されることができる。ライブラリに保存されたモデルコンポーネント９０２の構造及び動作は、完全に定義され、固定されることができ、コンポーネント９０２のあらゆるインスタンスが、その固定された構造及び動作を実施することができる。

【０１０４】

モデルコンポーネント９０４は、モデリング環境４００においてユーザによって作成されたグラフィカルモデルコンポーネントであってもよい。ただし、モデルコンポーネント９０４は、ファンクションコール型呼び出しスタイルを使用するように作成されたものである場合がある。ファンクションコールベースのモデルコンポーネント９０４において速度ベースのマルチタスクベースのモデルコンポーネントインスタンス９０２’を使用する

ために、モデルアダプタ４１２は、モデルコンポーネントインスタンス９０２’のためのアダプテーション層インターフェイス９０８を構築することができる。さらに、モデルコンポーネント９０４は、モデルコンポーネントインスタンス９０２’の種々の実行可能エンティティを呼び出すために、アダプテーション層インターフェース９０８に接続されたモデル要素９１０～９１２で示される機能又はロジックを含む場合がある。アダプテーション層は、モデルコンポーネント９０４のファンクションコールに基づく呼び出しを、モデルコンポーネントインスタンス９０２’の速度ベースのマルチタスク型呼び出しスタイルにバインドすることができる。重要なことに、モデルコンポーネントインスタンス９０２’が、モデルコンポーネント９０４内で実行されるように修正される必要はない。例えば、速度ベースのマルチタスク型モデルコンポーネントインスタンス９０２’をファンクションコールモデルコンポーネント９０４内で実行するために、モデルコンポーネントインスタンス９０２’のモデル要素を追加、削除、置換、又は変更する必要はない。モデルコンポーネントインスタンス９０２’に変更を加える必要がないため、モデルコンポーネントインスタンス９０２’を再検証する必要はない。

【０１０５】

モデルコンポーネント９０４は、破線矢印９１６で示されるように、モデル９１４（モデル＿１）に含まれる場合がある。モデル９１４は、モデリング環境４００においてユーザによって作成されたグラフィカルモデルであってもよい。モデル９１４は、速度ベースのマルチタスク型呼び出しスタイルを使用するように作成される場合がある。ファンクションコールベースのモデルコンポーネント９０４をモデル９１４で使用するために、モデルアダプタ４１２は、モデルコンポーネント９０４のためのアダプテーション層インターフェース９１８を構築することができる。さらに、モデル９１４は、モデルコンポーネント９０４の種々の実行可能エンティティを呼び出すために、アダプテーション層インターフェース９１８に接続されたモデル要素９２０及び９２２で示されるロジックを含む場合があり、モデルコンポーネント９０４は、同様に、モデルコンポーネント９０２’の種々の実行可能エンティティを呼び出すことができる。

【０１０６】

参照符号９０２”で示されるモデルコンポーネント９０２の別のインスタンスが、破線矢印９２６で示されるように、別のモデル９２４（モデル＿２）に含まれる場合がある。モデル９２４は、モデリング環境４００においてユーザによって作成されたグラフィカルモデルであってもよい。ただし、モデル９２４は、エッジトリガー型呼び出しスタイルを使用するように作成されたものである場合がある。エッジトリガベースのモデル９２４において速度ベースのマルチタスクベースのモデルコンポーネントインスタンス９０２”を使用するために、モデルアダプタ４１２は、モデルコンポーネントインスタンス９０２”のためのアダプテーション層及びアダプテーション層インターフェイス９２８を構築することができる。一部の実施形態では、以前に作成されたアダプテーション層が、モデルコンポーネントインスタンス９０２”のために使用される場合がある。さらに、モデル９２４は、モデルコンポーネントインスタンス９０２”の種々の実行可能エンティティを呼び出すために、アダプテーション層インターフェース９２８に接続されたモデル要素９３０～９３２で示されるロジックを含む場合がある。アダプテーション層は、モデル９２４のエッジトリガー型呼び出しを、モデルコンポーネントインスタンス９０２”の速度ベースのマルチタスク型呼び出しスタイルにバインドすることができる。この場合も、モデルコンポーネントインスタンス９０２”が、モデル９２４内で実行されるように修正される必要はない。したがって、モデルコンポーネントインスタンス９０２”を再検証する必要はない。

【０１０７】

図９は、モデルコンポーネント９０２のために生成されたコードの再利用も示している。例えば、コード発生器４０６は、モデルコンポーネント９０２のためのコード９３４を生成することができる。コード発生器４０６はさらに、モデルコンポーネントインスタンス９０２’を含むモデルコンポーネント９０４のためのコード９３６を生成することがで

きる。モデルコンポーネント９０４に含まれるモデルコンポーネントインスタンス９０２’のための新しいコードを生成する代わりに、コード発生器４０６は、モデルコンポーネント９０２のために生成されたコード９３４を、モデルコンポーネント９０４のために生成されたコード９３６内で再利用することができる。コード発生器４０６は、アダプテーション層のためのコード９３８を生成することができ、このコード９３８をモデルコンポーネント９０４のためのコード９３６に含めることができる。コード発生器４０６はさらに、矢印９４０及び９４２で示されるように、アダプテーション層のために生成されたコード９３８に対する種々の呼び出し及び戻りのためのコードを生成することができる。

【０１０８】

コード発生器４０６は、モデル９１４のためのコード９４４を生成することができる。モデル９１４に含まれるモデルコンポーネント９０４のための新しいコードを生成する代わりに、コード発生器４０６は、モデルコンポーネントインスタンス９０２’のために生成されたコード９３４を含む、モデルコンポーネント９０４のために生成されたコード９３６を再利用することができる。コード発生器４０６は、モデルコンポーネント９０４に向けて、アダプテーション層のためのコード９４６を生成することができる。コード発生器４０６はさらに、矢印９４８及び９５０で示されるように、アダプテーション層のために生成されたコード９４６に対する種々の呼び出し及び戻りのためのコードを生成することができる。

【０１０９】

コード発生器４０６は、モデルコンポーネントインスタンス９０２“を含むモデル９２４のためのコード９５２を生成することができる。モデルコンポーネントインスタンス９０２“のための新しいコードを生成する代わりに、コード発生器４０６は、モデルコンポーネント９０２のために生成されたコード９３４を再利用することができる。コード発生器４０６は、モデルコンポーネントインスタンス９０２”のために作成されたアダプテーション層のためのコード９５４を生成することができる。コード発生器４０６はさらに、矢印９５６及び９５８で示されるように、アダプテーション層のために生成されたコード９５４に対する種々の呼び出し及び戻りのためのコードを生成することができる。

【０１１０】

あるモデルコンポーネントの種々の実行可能エンティティへの分割は、ユーザ入力によって案内されてもよい。例えば、モデルコンポーネントの分割を制御するために、１以上の設定が使用可能である場合がある。１以上の設定は、ユーザが設定可能であってもよい。例えば、設定は、同じサンプル速度で実行される実行可能エンティティに対して、又は同じファンクションコールに応答して、別々のパーティションを作成すべきか否かを決定する場合がある。

【０１１１】

図１０は、例示的モデルコンポーネント１０００を示す概略図である。モデルコンポーネント１０００は、初期化サブシステム１００２、リセットサブシステム１００４（ラベル「マイリセット」が付されている）、終了サブシステム１００６、及びアルゴリズムモデル部分１００８を含む場合がある。アルゴリズムモデル部分１００８は、複数の相互接続されたモデル要素を含む場合がある。アルゴリズムモデル部分１００８は、３つの独立した経路、例えば未接続の経路１０１０、１０１２、及び１０１４を含む場合がある。経路１０１０、１０１２、及び１０１４は、Ｉｎｐｏｒｔブロック１０１６～１０１８、利得ブロック１０２０～１０２２、合算ブロック１０２４～１０２６、単位遅延ブロック１０２８～１０３０、及びＯｕｔｐｏｒｔブロック１０３２～１０３４を、それぞれ含む場合がある。

【０１１２】

アルゴリズムモデル部分１００８は、マルチレート動的システムを表す場合があり、速度ベースのマルチタスク型モデリングスタイルを使用して実施される場合がある。具体的には、経路１０１０、１０１２、及び１０１４は、異なる速度で実行される場合がある。例えば、経路１０１０及び１０１４は、１秒の周期的速度で実行される場合がある一方、

経路１０１２は、２秒の周期的速度で実行される場合がある。

【０１１３】

初期化サブシステム１００２、マイリセットサブシステム１００４、及び終了サブシステム１００６は、初期化イベント、リセットイベント及び終了イベントのような種々のイベントをそれぞれ監視するイベントトリガー型サブシステムである場合がある。サブシステム１００２、１００４及び１００６は、それぞれのイベントの発生に応答して実行される例えば明示的動作のようなユーザ指定の機能を含む場合がある。

【０１１４】

分割エンジン４２２は、モデルコンポーネント１０００を分析し、モデルコンポーネント１０００を形成する種々の実行可能エンティティを識別することができる。一部の実施形態では、分割エンジン４２２は、経路１０１０と経路１０１４を単一のパーティションの一部であるものと考える場合がある。なぜなら、これらは両方とも、同じ周期的速度、例えば１秒で動作するからである。ただし、ユーザは、２つの経路１０１０と経路１０１４の実行を明示的に別々にスケジューリングすること、例えば、これらを明示的に互いに独立してスケジュールすることを望む場合がある。したがって、ユーザは、経路１０１０及び経路１０１４を別々のパーティションとして識別するように分割エンジン４２２を案内する場合がある。

【０１１５】

図１１は、一実施形態による、図１０のモデルコンポーネント１０００のための例示的ウィンドウを示す概略図であり、ウィンドウは、パーティション凡例ダイアログ１１００の形を有する場合がある。パーティション凡例ダイアログ１１００は、複数の行１１０２ａ～１１０２ｆを含む場合があり、各行が、分割エンジン４２２によって識別された実行可能エンティティに対応している。また、パーティション凡例ダイアログ１１００は、色列１１０４、注釈列１１０６、説明列１１０８、及び値列１１１０を含む場合がある。パーティションは、色分け及び注釈を使用して、モデルコンポーネント１０００上に表される場合がある。様々な実行パーティションについて、モデルコンポーネント１０００上で使用される特定の色及び注釈が、パーティション凡例ダイアログ１１００の色列１１０４及び注釈列１１０６に含まれる場合がある。説明列１１０８は、実行可能エンティティの呼び出しスタイルを識別する情報を含む場合があり、値列１１１０は、実行エンティティの呼び出しスタイルの値を示す場合がある。

【０１１６】

例えば、行１１０２ａは、経路１０１０が赤色に着色され、ラベル「Ｄ１」で注釈されており、呼び出しスタイルが離散サンプル時間であり、その値が１秒であることを示している。行１１０２ｂは、経路１０１２が緑色に着色され、ラベル「Ｄ２」で注釈されており、呼び出しスタイルが離散サンプル時間であり、その値が２秒であることを示している。行１１０２ｃは、経路１０１４も赤色に着色され、ラベル「Ｄ３」で注釈されており、経路１０１０と同様に、呼び出しスタイルが離散サンプル時間であり、その値が１秒であることを示している。行１１０２ｄは、初期化サブシステム１００２が青色に着色され、ラベル「Ｉｎｉｔ」で注釈されており（初期化サブシステム１００２が開かれていた場合）、呼び出しスタイルがトリガー型であり、トリガーの値が［Ｉｎｉｔ］であることを示している。行１１０２ｅは、終了サブシステム１００６がオレンジ色に着色され、ラベル「Ｔｅｒｍ」で注釈されており（終端サブシステム１００６が開かれていた場合）、呼び出しスタイルがトリガー型であり、トリガーの値が［Ｔｅｒｍ］であることを示している。行１１０２ｆは、マイリセットサブシステム１００４が紫色に着色され、ラベル「リセット」で注釈されており（マイリセットサブシステム１００４が開かれていた場合）、呼び出しスタイルがトリガー型であり、トリガーの値が［Ｒｅｓｅｔ］であることを示している。

【０１１７】

図１２は、一実施形態による、モデルエディタウィンドウ１００上に開かれることがある例示的モデル１２００を示す概略図である。モデル１２００は、モデル参照ブロック１

２０２によってモデル１２００内に表されたモデルコンポーネント１０００を含む。モデル参照ブロック１２０２は、アダプテーション層インターフェース１２０４を含む場合があり、アダプテーション層インターフェース１２０４は、モデルコンポーネント１０００のために決定された種々のエントリポイントに対応するアクセスポイントを含む。例えば、アダプテーション層インターフェース１２０４は、初期化アクセスポイント１２０６、リセットアクセスポイント１２０８、終了アクセスポイント１２１０、２つの１秒アクセスポイント１２１２（ラベル「１ｓｅｃ」が付されている）及び１２１４（ラベル「１ｓｅｃ２」が付されている）、２秒アクセスポイント１２１６（ラベル「２ｓｅｃ１」が付されている）、並びに３つのデータ入力アクセスポイント１２１８～１２２０を含む。１秒アクセスポイント１２１２は、モデルコンポーネント１０００の経路１０１０に対応する一方、１秒アクセスポイント１２１４は、モデルコンポーネント１０００の経路１０１４に対応する場合がある。

【０１１８】

モデル１２００は、ファンクションコールベースのモデリングスタイルを実施することができる。例えば、モデル１２００は、ファンクションコールを発行するモデル要素を含む場合があり、これらのモデル要素の少なくとも一部を使用して、モデルコンポーネント１０００の種々の実行エンティティの実行を明示的にスケジュールすることができる。例えば、モデル１２００は、モデル１２００の実行中に電力信号１２２４及びリセット信号１２２６を供給する信号ビルダーブロック１２２２と、信号ビルダーブロック１２２２から電力信号１２２４及びリセット信号１２２６を受信する状態図（「スケジューラ」とラベル付けされている）１２２８とを含む。スケジューラ状態図１２２８は、ＳｔａｒｔＵｐ（）ファンクションコール１２３０、Ｒｅｓｅｔ（）ファンクションコール１２３２、ＳｈｕｔＤｏｗｎ（）ファンクションコール１２３４、及びＲｕｎ（）ファンクションコール１２３６を発行することができる。また、モデル１２００は、２つの関数発生器ブロック１２３８及び１２４０を含む場合があり、これらは、Ｒｕｎ１（）ファンクションコール１２４２及びＲｕｎ２（）ファンクションコール１２４４を発行することができる。

【０１１９】

ＳｔａｒｔＵｐ（）ファンクションコール１２３０は、アダプテーション層インターフェース１２０４の初期化アクセスポイント１２０６に接続される場合がある。これに応答して、マッピングエンジン４２６は、ＳｔａｒｔＵｐ（）ファンクションコール１２３０を、初期化イベントにバインドすることができる。Ｒｅｓｅｔ（）ファンクションコール１２３２は、アダプテーション層インターフェース１２０４のＲｅｓｅｔアクセスポイント１２０８に接続される場合がある。これに応答して、マッピングエンジン４２６は、Ｒｅｓｅｔ（）ファンクションコール１２３２を、リセットイベントにバインドすることができる。ＳｈｕｔＤｏｗｎ（）ファンクションコール１２３４は、アダプテーション層インターフェース１２０４の終了アクセスポイント１２１０に接続される場合がある。これに応答して、マッピングエンジン４２６は、ＳｈｕｔＤｏｗｎ（）ファンクションコール１２３６を、終了イベントにバインドすることができる。スケジューラ状態図１２２８のＲｕｎ（）ファンクションコール１２３６は、アダプテーション層インターフェース１２０４の１秒アクセスポイント１２１２に接続される場合がある。これに応答して、マッピングエンジン４２６は、Ｒｕｎ（）ファンクションコール１２３６を、アルゴリズムモデル部分１００８の経路１０１０の１秒の離散サンプル時間にバインドすることができる。

【０１２０】

関数発生器ブロック１２３８のＲｕｎ１（）ファンクションコール１２４２は、アダプテーション層インターフェース１２０４の２ｓｅｃ１アクセスポイント１２１６に接続される場合がある。これに応答して、マッピングエンジン４２６は、Ｒｕｎｌ（）ファンクションコール１２４２を、アルゴリズムモデル部分１００８の経路１０１２の２秒の離散サンプル時間にバインドすることができる。関数発生器ブロック１２４０のＲｕｎ２（）ファンクションコール１２４４は、アダプテーション層インターフェース１２０４の１ｓｅｃ２アクセスポイント１２１４に接続される場合がある。これに応答して、マッピング

エンジン４２６は、Ｒｕｎ２（）ファンクションコール１２４４を、アルゴリズムモデル部分１００８の経路１０１４の１秒の離散サンプル時間にバインドすることができる。

【０１２１】

モデルコンポーネント１１００の２つの経路１０１０及び１０１４について別々のアクセスポイントを設けることにより、たとえそれらの経路が両方とも同じ速度で実行される場合であっても、ユーザは、異なるロジックを使用することで、例えば信号ビルダーブロック１２２２及びスケジューラ状態図１２２８を一方のインスタンスで使用し、関数発生器ブロック１２４０を第２のインスタンスで使用することで、２つの経路１０１０及び１０１４に対応する実行可能エンティティの実行をスケジューリングするためのファンクションコールを生成することができる。

【０１２２】

説明したように、分割エンジン４２２によれば、例えば暗黙的分割のような自動的分割と、例えば明示的分割のようなユーザ指定の分割との両方が可能となる。エントリポイントは、各パーティションについて確立されることができ、暗黙的パーティションと明示的パーティションの両方が、アクセスポイントを介してインターフェース上に表現される場合がある。

【０１２３】

一部の実施形態では、アダプテーション層インターフェイスに接続されたユーザ指定のロジック又は機能を、例えばテンプレートとして、ライブラリに保存することができる。図３を参照すると、信号ビルダーブロック３１６（コマンド１）、状態図３２２（スケジューラ）、及び関数発生器ブロック３３２（周期関数発生器）は、モデルコンポーネント３０２の分割された実行可能エンティティをスケジューリングするためのユーザ指定のスケジューリングロジックを表している。所望の実行スケジュールを実施するこのスケジューリングロジックは、テンプレートとして保存され、他のモデルコンポーネントとともに再利用される場合がある。例えば、保存されたスケジューリングロジックテンプレートのインスタンスは、別のモデルに含められ、別のモデルコンポーネントを実行するために使用される場合がある。別のモデルコンポーネントとしては、モデルコンポーネント３０２のアダプテーション層インターフェース３０４の初期化アクセスポイント３０６、マイリセットアクセスポイント３０８、終了アクセスポイント３１０、１ｓ＿Ｒｕｎアクセスポイント３１２、及び２ｓ＿Ｒｕｎアクセスポイント３１４のような、初期化アクセスポイント、リセットアクセスポイント、終了アクセスポイント、１ｓ＿Ｒｕｎアクセスポイント、及び２ｓ＿Ｒｕｎアクセスポイントが挙げられる。

【０１２４】

さらに、モデル階層の複数のレベルに定義されたスケジューリングロジックが、集約されてもよい。集約されたスケジューリングロジックは、テンプレートとして保存される場合がある。例えば、集約されたスケジューリングロジックは、ライブラリに保存される場合がある。集約されたスケジューリングロジックは、例えば別のモデルで、及び／又は同じモデルの他の部分で、再利用されることができる。一部の実施形態では、所与のモデルについて、スケジュールパラメータ及び属性の値が指定される場合があり、これらの値も、例えばスケジューリングロジックと一緒に、ライブラリに保存される場合がある。例えば、ユーザは、初期化コンポーネントが特定の態様で集約され、例えば最上位レベルを除くすべての階層レベルで集約され、あるいは、イネーブルサブシステムの部分階層で集約されるようにする等のために、初期化コンポーネントの属性を設定することができる。この属性の値は、保存され、１以上の他のモデルで再利用され、例えば採用される場合がある。

【０１２５】

図１３は、本発明の実施形態を実施するための例示的コンピュータシステム又はデータ処理システム１３００を示す概略図である。コンピュータシステム１３００は、システムバス１３１２によって相互接続されたプロセッサ１３０２のような１以上の処理要素、メインメモリ１３０４、ユーザ入出力（Ｉ／Ｏ）１３０６、ディスクドライブ１３０８のよ

うな不揮発性データ記憶装置、及び、リムーバブル媒体ドライブを含む場合がある。コンピュータシステム１３００は、ネットワークインターフェースカード（ＮＩＣ）１３１４のような通信ユニットをさらに含む場合がある。ユーザＩ／Ｏ１３０６には、キーボード１３１６、マウス１３１８のようなポインティングデバイス、及びディスプレイ１３２０が含まれ得る。他のユーザＩ／Ｏ１３０６コンポーネントとしては、音声（”voice” or “speech”）コマンドシステム、タッチパッド及びタッチスクリーン、プリンタ、プロジェクタなどが挙げられる。プロセッサの例としては、シングル若しくはマルチコアの中央演算処理装置（ＣＰＵ）、グラフィック処理装置（ＧＰＵ）、フィールドプログラマブルゲートアレイ（ＦＰＧＡ）、特定用途向け集積回路（ＡＳＩＣ）、マイクロプロセッサ、マイクロコントローラなどが挙げられる。

【０１２６】

メインメモリ１３０４は、ランダムアクセスメモリ（ＲＡＭ）である場合があり、オペレーティングシステム１３２２のような複数のプログラムライブラリ又はモジュールと、モデリング環境１３００のようなオペレーティングシステム１３２２と接続する１以上のアプリケーションプログラムとを記憶している場合がある。

【０１２７】

リムーバブル媒体ドライブ１３１０は、ＣＤ、ＤＶＤ、フロッピー（登録商標）ディスク、ＳＳＤ（ソリッドステートドライブ）、テープ、フラッシュメモリ、又は他の非一時的媒体のようなコンピュータ読み取り可能媒体１３２６を受け入れ、読み取ることができる。リムーバブル媒体ドライブ１３１０は、コンピュータ読み取り可能媒体１３２４に書き込むこともできる。

【０１２８】

適当なコンピュータシステムとしては、パーソナルコンピュータ（ＰＣ）、ワークステーション、サーバ、ラップトップ、タブレット、パームコンピュータ、スマートフォン、電子リーダー、及び他のポータブルコンピューティングデバイスなどが挙げられる。ただし、当業者であれば、図１３のコンピュータシステム１３００が単なる例示のためのものであり、本発明が他のコンピュータシステム若しくは装置、データ処理システム若しくは装置、又は計算システム若しくは装置と共に使用されてもよいことを理解するであろう。また、本発明は、コンピュータネットワークにおいて使用されてもよく、例えば、クライアント－サーバ、アーキテクチャ、あるいは、パブリック及び／又はプライベートクラウドコンピューティング装置において使用されてもよい。例えば、モデリング環境１３００は、１以上のクラウドサーバ又はデバイス上に置かれ、マイクロソフト社のリモートデスクトップ接続ツールのようなウェブポータルシステム又はアプリケーションホスティングシステムを通して、リモートクライアントによって取得される場合がある。

【０１２９】

適当なオペレーティングシステム１３２２としては、とりわけ、ワシントン州レドモンドのマイクロソフト社のＷｉｎｄｏｗｓ（登録商標）シリーズのオペレーティングシステム；カリフォルニア州マウンテンビューのＧｏｏｇｌｅ社のＡｎｄｒｏｉｄ（登録商標）及びＣｈｒｏｍｅ（登録商標）　ＯＳオペレーティングシステム；Ｌｉｎｕｘオペレーティングシステム；カリフォルニア州クパチーノのアップル社のＭＡＣ　ＯＳ（登録商標）シリーズのオペレーティングシステム；及びＵＮＩＸ（登録商標）シリーズのオペレーティングシステムなどが挙げられる。オペレーティングシステム１３２２は、種々のアプリケーション又はモジュールのために、メモリを割り当て、ファイルシステムにしたがってデータオブジェクトやファイルを編成し、要求に優先順位を付け、Ｉ／Ｏを管理するといった、種々のサービス又は機能を提供することができる。オペレーティングシステム１３２２は、データ処理システム１３００によって提供されることがある仮想マシン上で実行される場合がある。

【０１３０】

上述のように、エンジニア、科学者、プログラマー、開発者等のユーザは、キーボード１３１６、マウス１３１８、及びディスプレイ１３２０のような１以上の入力デバイスを

使用して、モデリング環境１３００を動作させ、１以上のモデルを構築し、改訂することができる。説明したように、モデルは計算的であり、種々の実行可能なセマンティクスを有することができる。具体的には、モデルは、シミュレートされてもよいし、実行されてもよい。具体的には、モデルは、時間ベース、イベントベース、状態ベース、メッセージベース、頻度ベース、制御フローベース、及びデータフローベースの実行セマンティクスのうちの１以上を提供することができる。モデルの実行により、設計中又は評価中のシステムの動作がシミュレートされる。グラフィカルモデルという用語は、グラフィカルプログラムを含むことを意図している。

【０１３１】

モデリング環境の例としては、マスワークス社のＭＡＴＬＡＢ（登録商標）アルゴリズム開発環境及びＳｉｍｕｌｉｎｋ（登録商標）モデルベースの設計環境；同じくマスワークス社のＳｉｍｓｃａｐｅ（商標）物理モデリングシステム及びＳｔａｔｅｆｌｏｗ（登録商標）状態図ツール；カナダ国オンタリオ州ウォータールーのウォータールーメイプル社のＭａｐｌｅＳｉｍ（登録商標）物理モデリング及びシミュレーションツール；テキサス州オースティンのナショナルインスツルメンツ社のＬＡＢＶＩＥＷ（登録商標）仮想機器プログラミングシステム、及び、ＮＩ　ＭａｔｒｉｘＸモデルベースの設計製品；カリフォルニア州サンタクララのアジレントテクノロジー社の仮想エンジニアリング環境（ＶＥＥ）製品；カリフォルニア州マウンテンデューのシノプシス社のＳｙｓｔｅｍ　Ｓｔｕｄｉｏモデルベースの信号処理アルゴリズム設計分析ツール、及びＳＰＷ信号処理アルゴリズムツール；カリフォルニア州サンノゼのザイリンクス社の統一モデリング言語（ＵＭＬ）システム、システムモデリング言語（ＳｙｓＭＬ）、及びＳｙｓｔｅｍ　Ｇｅｎｅｒａｔｏｒシステム；並びに、ニューヨーク州ソマーズのＩＢＭ社のＲａｔｉｏｎａｌ（登録商標）Ｒｈａｐｓｏｄｙ（登録商標）設計マネージャソフトウェアが挙げられる。高度なモデリング環境で作成された種々のモデルは、あまり多くの実施詳細を含まない場合があり、したがって、Ｃ、Ｃ＋＋、Ｃ＃、ＳｙｓｔｅｍＣプログラミング言語のような特定のプログラミング言語よりも高いレベルで動作する場合がある。

【０１３２】

当業者は、ＭＡＴＬＡＢ（登録商標）アルゴリズム開発環境が、とりわけ、デジタル信号処理（ＤＳＰ）設計のための数学指向のテキストプログラミング環境であることを理解するであろう。Ｓｉｍｕｌｉｎｋ（登録商標）モデルベースの設計環境は、とりわけ、動的システムその他のシステムのモデリング及びシミュレーションのためのモデリングツールである。ＭＡＴＬＡＢ（登録商標）環境及びＳｉｍｕｌｉｎｋ（登録商標）環境によれば、アルゴリズムの開発と探索を容易にし、モデルベースの設計を支援する多数の高度な機能が得られる。高度な機能の例としては、とりわけ、動的タイピング、配列ベースの動作、データ型の推論、サンプル時間の推論、及び実行順序の推論が挙げられる。

【０１３３】

一部の実施形態では、とりわけ、Ｃ、Ｃ＋＋、Ｃ＃、及びＳｙｓｔｅｍＣプログラミング言語のような高レベルなモデリング環境４００に比べて低レベルなプログラミング言語を使用して、１以上のモデルを作成する場合がある。

【０１３４】

モデリング環境４００内に構築されるモデルは、テキストモデル、ブロック図のようなグラフィカルモデル、状態ベースのモデル、及びそれらの組み合わせを含む場合がある。所与のモデルは、システムの動作をシミュレートすることができ、例えば、システムの動作を近似することができる。

【０１３５】

コード発生器の例としては、限定はしないが、マサチューセッツ州ナティックのマスワークス社のＳｉｍｕｌｉｎｋ（登録商標）Ｃｏｄｅｒ（商標）、Ｅｍｂｅｄｄｅｄ　Ｃｏｄｅｒ（登録商標）、及びＨＤＬ　Ｃｏｄｅｒ（商標）製品、並びに、ドイツ国パーダーボルンのｄＳｐａｃｅ社のＴａｒｇｅｔＬｉｎｋ（登録商標）製品が挙げられる。

【０１３６】

図１４は、本明細書に記載された種々のシステム及び／又は方法を実施することができる例示的分散コンピューティング環境１４００を示す概略図である。環境１４００は、ネットワーク１４１０のような１以上のネットワークによって相互接続された２つのサーバ１４０２、１４０４及び３つのクライアント１４０６～１４０８のような、クライアント装置とサーバ装置を含む場合がある。サーバ１４０２及び１４０４は、クライアント１４０６～１４０８によってアクセス可能なアプリケーション又はプロセスを含む場合がある。例えば、サーバ１４０２は、テクニカルコンピューティング環境（ＴＣＥ）１４１２を含む場合があり、ＴＣＥ１４１２は、モデリング環境１３００のようなモデリング環境を含み、又はそのようなモデリング環境へのアクセスを有する場合がある。サーバ１４０４は、コード発生器１３０６のようなコード発生器を含む場合がある。環境１４００の種々の装置は、有線接続によって相互接続されてもよいし、無線接続によって相互接続されてもよく、あるいは、有線接続と無線接続の組み合わせによって相互接続されてもよい。

【０１３７】

サーバ１４０２及び１４０４は、情報を受信し、生成し、記憶し、処理し、実行し、及び／又は提供することができる１以上の装置を含む場合がある。例えば、サーバ１４０２及び１４０４は、サーバ、デスクトップコンピュータ、ラップトップコンピュータ、タブレットコンピュータ、ハンドヘルドコンピュータ、又は同様の装置のような計算装置を含む場合がある。一部の実施形態では、サーバ１４０２及び１４０４は、ＴＣＥ１４１２、モデリング環境１３００、及び／又はコード発生器１３０６を含む場合がある。

【０１３８】

クライアント１４０６～１４０８は、情報を受信し、生成し、記憶し、処理し、実行し、及び／又は提供することができる。情報は、例えば１以上のネットワーク及び／又は１以上の装置における使用に適合する実質的に任意のフォーマットを有する任意のタイプの機械読み取り可能情報を含む場合がある。情報は、デジタル情報及び／又はアナログ情報を含む場合がある。情報はさらに、パケット化され、及び／又は、非パケット化される場合がある。一実施形態では、クライアント１４０６～１４０８は、ネットワーク１４１０を介してサーバ１４０２及び１４０４から、データ及び／又はコードをダウンロードすることができる。一部の実施形態では、クライアント１４０６～１４０８は、デスクトップコンピュータ、ワークステーション、ラップトップコンピュータ、タブレットコンピュータ、ハンドヘルドコンピュータ、携帯電話（例えば、スマートフォンや無線電話など）、電子リーダ、又は同様の装置である場合がある。一部の実施形態では、クライアント１４０６～１４０８は、サーバ１４０２、１４０４から情報を受信することができ、及び／又は、サーバ１４０２及び１４０４に情報を送信することができる。

【０１３９】

ネットワーク１４１０は、１以上の有線ネットワーク及び／又は無線ネットワークを含む場合がある。例えば、ネットワーク１４１０は、セルラーネットワーク、公衆陸上移動体ネットワーク（「ＰＬＭＮ」）、ローカルエリアネットワーク（「ＬＡＮ」）、ワイドエリアネットワーク（「ＷＡＮ」）、メトロポリタンエリアネットワーク（「ＭＡＮ（ＰＳＴＮ））、アドホックネットワーク、イントラネット、インターネット、光ファイバベースのネットワーク、及び／又は、これらのタイプ若しくは他のタイプのネットワークの組み合わせを含む場合がある。任意のネットワークプロトコルを使用してネットワーク装置間で情報を交換することができ、限定はしないが、例えば、インターネットプロトコル（ＩＰ）、非同期転送モード（ＡＴＭ）、同期光ネットワーク（ＳＯＮＥＴ）、ユーザデータグラムプロトコル（ＵＤＰ）、ＩＥＥＥ（Ｉｎｓｔｉｔｕｔｅ ｏｆ Ｅｌｅｃｔｒｉｃａｌ ａｎｄ Ｅｌｅｃｔｒｏｎｉｃｓ Ｅｎｇｉｎｅｅｒｓ）８０２．１１などのネットワークプロトコルが使用される場合がある。

【０１４０】

図１４に示した装置及び／又はネットワークの数は、例として提供したものである。実際には、さらに別の装置及び／又はネットワークがあってもよいし、もっと少ない数の装置及び／又はネットワークしかなくてもよく、異なる装置及び／又はネットワークがあっ

てもよく、あるいは、図１４に示したものとは異なる配置の装置及び／又はネットワークがあってもよい。また、図１４に示した２以上の装置が、単一の装置内で実施されてもよいし、図１４に示した単一の装置が、複数の分散装置として実施されてもよい。さらに、分散コンピューティング環境１４００の装置のうちの１以上が、環境１４００の別の１以上の装置によって実行されるものとして説明された１以上の機能を実施してもよい。

【０１４１】

下記の例は、本開示の方法及び／又はシステムの１以上の態様を実施する。これらの例は、非限定的例である。実施形態によっては、異なる例の特徴が、組み合わせられる場合がある。実施形態によっては、各例の種々の特徴は、変更され、又は削除される場合がある。

【０１４２】

システム又は方法は、モデル要素と前記モデル要素間のリンクとを含むモデルについて、

前記モデル要素及び前記リンクの少なくとも一部が、宣言型言語によって定義され、

前記モデル要素及び前記リンクの前記少なくとも一部の挙動が、前記宣言型言語によって暗黙的に定義された計算上の実施形態を含み、

前記モデルが、第１の表示形式を有する場合に、

１以上のプロセッサにより、前記モデル及び前記計算上の実施形態を分析し、

前記１以上のプロセッサにより、前記分析に基づいて、前記計算上の実施形態の構造を変更することなく調節することができる前記モデルの調節可能な属性を自動的に決定し、前記調節可能な属性が、前記宣言型言語によって暗黙的に定義され、又は前記モデル要素若しくは前記リンクによって明示的に定義されており、

前記第１の表示形式とは異なる第２の表示形式の前記モデルと共に表示するために、前記１以上のプロセッサにより、前記調節可能な属性のうちの少なくとも１つを抽出すること

を含み、

前記調節可能な属性のうちの前記少なくとも１つが、ユーザーインターフェースに明示的に表示され、前記調節可能な属性のうちの前記少なくとも１つを調節することを可能にする。

【０１４３】

システム及び方法は、モデル要素と前記モデル要素間のリンクとを含むモデルについて、

前記モデル要素及び前記リンクの少なくとも一部が、宣言型言語によって定義され、

前記モデル要素及び前記リンクの前記少なくとも一部の挙動が、前記宣言型言語によって暗黙的に定義された計算上の実施形態を含み、

前記モデルが、第１の表示形式を有する場合に、

１以上のプロセッサにより、前記モデル及び前記計算上の実施形態を分析し、

前記１以上のプロセッサにより、前記分析に基づいて、前記計算上の実施形態の構造を変更することなく調節することができる前記モデルの調節可能な属性を自動的に決定し、前記調節可能な属性が、所定の特徴を有し、当該所定の特徴に基づいて、前記モデルを実行することができ、

前記第１の表示形式とは異なる第２の表示形式の前記モデルと共に表示するために、前記１以上のプロセッサにより、前記調節可能な属性のうちの少なくとも１つを抽出すること

を含み、

前記調節可能な属性のうちの前記少なくとも１つが、

対応する所定の特徴を有し、

ユーザーインターフェースに明示的に表示され、前記調節可能な属性のうちの前記少

なくとも１つを、前記対応する所定の特徴とは異なる特徴を有するように調節することを可能にする。

【０１４４】

一部の実施形態では、前記調節可能な属性は、前記モデル内の１以上の分割された実行可能エンティティの１以上のインターフェースを含み、各実行可能エンティティが、１以上のモデル要素及び／又はリンクを含む。

【０１４５】

一部の実施形態では、前記モデル及び前記計算上の実施形態を分析することは、前記モデルを分割し、前記１以上の分割された実行可能エンティティを決定することを含む。

【０１４６】

一部の実施形態では、前記１以上のインターフェースは、１以上の明示的イベント又は１以上の暗黙的イベントによってトリガー駆動されるように調節されることができる。

【０１４７】

一部の実施形態では、前記１以上の明示的イベントは、ファンクションコール又はメッセージであり、前記１以上の暗黙的イベントは、トリガ信号又は周期的タイマーである。

【０１４８】

一部の実施形態では、第１の分割された実行可能エンティティは、前記モデル内の第１の階層レベルにあり、第２の分割された実行可能エンティティは、前記モデル内の第２の異なる階層レベルにあり、前記第１の分割された実行可能エンティティの第１のインターフェース及び前記第２の分割された実行可能エンティティの第２のインターフェースは、同じメカニズム又は異なるメカニズムによってトリガー駆動されるように調節されることができる。

【０１４９】

一部の実施形態では、前記調節可能な属性は、１以上のモデル要素の１以上の実行速度を含む。

【０１５０】

一部の実施形態では、前記システム又は方法は、前記調節可能な属性のうちの前記少なくとも１つを前記第２の表示形式の前記モデル内に表示することを含む。

【０１５１】

一部の実施形態では、前記モデルは、前記第１の表示形式及び前記第２の表示形式で実行可能である。

【０１５２】

一部の実施形態では、前記システム又は方法は、前記モデルのためのコードを生成することを含み、前記コードは、前記調節可能な属性のうちの前記抽出された少なくとも１つに対応する部分を含む。

【０１５３】

システム及び方法は、メモリから複数の実行可能エンティティを含むモデルコンポーネントを取得し、

前記モデルコンポーネントの前記複数の実行可能エンティティを分割し、

前記複数の実行可能エンティティの調節可能な属性を識別し、

前記モデルコンポーネントについて、前記複数の実行可能エンティティの前記調節可能な属性を含むアダプテーション層を確立し、

前記調節可能な属性を、前記モデルコンポーネントに関連するグラフィカルアフォーダンス内に公開し、

前記モデルコンポーネントに関連する前記グラフィカルアフォーダンス内に公開された前記調節可能な属性にリンクされたロジックをバインドすること、及び

前記モデルコンポーネントの実行中に前記調節可能な属性にリンクされた前記ロジックを使用すること、若しくは

前記モデルコンポーネントのためのコードを生成することのうちの少なくとも一方

を含み、

前記生成されたコードが、前記調節可能な属性にリンクされた前記ロジックを実施する１以上のコード部分を含む。

【０１５４】

一部の実施形態では、前記システム及び方法は、前記モデルコンポーネントの前記複数の実行可能エンティティのための呼び出しスケジュールを決定することをさらに含み、前記呼び出しスケジュールは、前記調節可能な属性にリンクされた前記ロジックの少なくとも一部によって示される。

【０１５５】

一部の実施形態では、前記システム及び方法は、前記モデルコンポーネントとのデータの交換により、前記呼び出しスケジュールを調整することをさらに含む。

【０１５６】

一部の実施形態では、前記調節可能な属性にリンクされた前記ロジックが、ユーザ指定である。

【０１５７】

一部の実施形態では、前記モデルコンポーネントの前記複数の実行可能エンティティは、第１の呼び出しスタイルを使用して呼び出され、前記ユーザ指定のロジックは、前記第１の呼び出しスタイルとは異なる第２の呼び出しスタイルを実施し、前記バインドすることは、前記第２の呼び出しスタイルを前記第１の呼び出しスタイルにマッピングすることを含む。

【０１５８】

一部の実施形態では、前記第１の呼び出しスタイルは、速度ベース型、イベントベース型、ファンクションコール型、又はトリガー型のうちの何れか１つである。

【０１５９】

一部の実施形態では、前記ユーザ指定のロジックは、前記モデルコンポーネントを含む親モデルに含まれ、前記ユーザ指定のロジックは、前記グラフィカルアフォーダンスに含まれる前記公開された調節可能な属性に関連するアクセスポイントにグラフィカルに接続される。

【０１６０】

一部の実施形態では、前記システム及び方法は、前記親モデルと前記モデルコンポーネントとの間のデータの交換を調整することをさらに含む。

【０１６１】

一部の実施形態では、　前記モデルコンポーネントが、階層レベルを有するモデルに含まれ、前記モデルコンポーネントが、より高いレベルのモデルコンポーネントに含まれ、前記システム及び方法は、前記モデルコンポーネントの所与の調節可能な属性を、前記モデルコンポーネントを含むより高いレベルのモデルコンポーネントについての調節可能な属性と集約することをさらに含む。

【０１６２】

一部の実施形態では、前記ユーザ指定のロジックは、複数のモデルコンポーネントを通して実施される。

【０１６３】

システム及び方法は、モデル要素と前記モデル要素間のリンクとを含むモデルについて、

前記モデル要素及びリンクの少なくとも一部が、宣言型言語によって定義され、前記モデル要素及び前記リンクの前記少なくとも一部の挙動が、前記宣言型言語によって暗黙的に定義された計算上の実施形態を含み、

前記モデルの調節可能な属性が、前記計算上の実施形態の構造を変更することなく調節のために抽出され、前記調節可能な属性が、所定の特徴を有し、当該所定の特徴に基づいて、前記モデルを実行することができ、

前記調節可能な属性のうちの少なくとも１つが、対応する所定の特徴を有し、前記対応する所定の特徴とは異なる特徴を有するように調節可能である場合に、

前記モデルのためのコードを生成すること
を含み、
前記コードが、前記調節可能な属性に対応する１以上のコード部分と、前記計算上の実施形態に対応する他のコード部分とを含み、
前記調節可能な属性の調節によって、前記他のコード部分に影響を及ぼすことなく、前記１以上のコード部分の調節がもたらされる。
【０１６４】
システム及び方法は、複数の実行可能エンティティを含むモデルコンポーネントについて、
前記モデルコンポーネントの前記複数の実行可能エンティティが分割され、
前記複数の実行可能エンティティのアクセスポイントが識別され、
前記複数の実行可能エンティティの前記アクセスポイントを含むアダプテーション層が確立され、
前記調節可能な属性に選択可能なロジックがリンクされている場合に、
少なくとも１つのプロセッサにより、前記モデルのためのコードを生成すること
を含み、
前記コードは、前記モデルコンポーネントの実行可能エンティティの前記アクセスポイントのうちの少なくとも１つに対するアクセスメソッドを実施する、前記選択可能なロジックを実施する１以上のコード部分を含む。
【０１６５】
態様１.
好ましくはメモリから複数の実行可能エンティティを含むモデルコンポーネントを取得し、
前記モデルコンポーネントの前記複数の実行可能エンティティを分割し、
前記複数の実行可能エンティティの調節可能な属性を識別し、
前記モデルコンポーネントについて、前記複数の実行可能エンティティの前記調節可能な属性を含むアダプテーション層を確立し、
前記調節可能な属性を、前記モデルコンポーネントに関連するグラフィカルアフォーダンス内に公開し、
前記モデルコンポーネントに関連する前記グラフィカルアフォーダンス内に公開された前記調節可能な属性にリンクされたロジックをバインドすること、及び
前記モデルコンポーネントの実行中に前記調節可能な属性にリンクされた前記ロジックを使用すること又は前記モデルコンポーネントのためのコードを生成することのうちの少なくとも一方
を含み、
前記生成されたコードが、前記調節可能な属性にリンクされた前記ロジックを実施する１以上のコード部分を含む、方法。
【０１６６】
態様２.
前記モデルコンポーネントの前記複数の実行可能エンティティのための呼び出しスケジュールを決定することをさらに含み、
前記呼び出しスケジュールは、前記調節可能な属性にリンクされた前記ロジックの少なくとも一部によって示される、態様１に記載の方法。
【０１６７】
態様３.
前記モデルコンポーネントとのデータの交換により、前記呼び出しスケジュールを調整することをさらに含む、先行する態様の何れか、特に態様２に記載の方法。
【０１６８】
態様４.
前記調節可能な属性にリンクされた前記ロジックは、ユーザ指定である、先行する態様

の何れか、特に態様１に記載の方法。
【０１６９】
態様５．
前記モデルコンポーネントの前記複数の実行可能エンティティは、第１の呼び出しスタイルを使用して呼び出され、
前記ユーザ指定のロジックは、前記第１の呼び出しスタイルとは異なる第２の呼び出しスタイルを実施し、
前記バインドすることは、前記第２の呼び出しスタイルを前記第１の呼び出しスタイルにマッピングすることを含む、先行する態様の何れか、特に態様４に記載の方法。
【０１７０】
態様６．
前記第１の呼び出しスタイルは、速度ベース型、イベントベース型、ファンクションコール型、又はトリガー型のうちの何れか１つである、先行する態様の何れか、特に態様５に記載の方法。
【０１７１】
態様７．
前記ユーザ指定のロジックは、前記モデルコンポーネントを含む親モデルに含まれ、
前記ユーザ指定のロジックは、前記グラフィカルアフォーダンスに含まれる前記公開された調節可能な属性に関連するアクセスポイントにグラフィカルに接続される、先行する態様の何れか、特に態様４に記載の方法。
【０１７２】
態様８．
前記親モデルと前記モデルコンポーネントとの間のデータの交換を調整することをさらに含む、先行する態様の何れか、特に態様７に記載の方法。
【０１７３】
態様９．
前記モデルコンポーネントが、階層レベルを有するモデルに含まれ、
前記モデルコンポーネントが、より高いレベルのモデルコンポーネントに含まれ、
前記方法は、
前記モデルコンポーネントの所与の調節可能な属性を、前記モデルコンポーネントを含むより高いレベルのモデルコンポーネントについての調節可能な属性と集約すること
をさらに含む、先行する態様の何れか、特に態様１に記載の方法。
【０１７４】
態様１０．
前記ユーザ指定のロジックは、複数のモデルコンポーネントを通して実施される、先行する態様の何れか、特に態様４に記載の方法。
【０１７５】
態様１１．
前記モデルコンポーネントが、少なくとも１つの動作及び／又は少なくとも１つの手順を含み、
前記分割は、
　前記モデルコンポーネントの依存関係グラフを作成することを
を含み、
前記依存関係グラフは、前記モデルコンポーネントに含まれる前記少なくとも１つの動作及び／又は前記少なくとも１つの手順、並びに／あるいは、前記少なくとも１つの動作及び／又は前記少なくとも１つの手順の間のデータ及び／又は制御の依存関係に関する情報を含む、先行する態様の何れかに記載の方法。
【０１７６】
態様１２．
前記依存関係グラフは、複数の部分グラフを含み、

前記依存関係グラフは、部分グラフによって受信され、及び／又は生成されたデータを表す、先行する態様の何れか、特に態様１１に記載の方法。

【０１７７】

態様１３．

前記依存関係グラフを検査すること

をさらに含み、

前記検査は、何れの他の部分グラフからも独立した少なくとも１つの部分グラフを、実行可能なエンティティとして識別することを含む、先行する態様の何れか、特に態様１１または態様１２に記載の方法。

【０１７８】

態様１４．

前記調節可能な属性を識別することは、

少なくとも１つの実行可能エンティティのエントリポイント、前記少なくとも１つの実行可能エンティティによって使用されるサービス、及び／又は、前記少なくとも１つの実行可能エンティティによって処理される入力データを、調節可能な属性として識別すること、並びに／あるいは、

エントリポイントに関連する呼び出しスタイルを調節可能な属性として識別することを含む、先行する態様の何れか、特に態様１１～１３の何れかに記載の方法。

【０１７９】

態様１５．

少なくとも１つのプロセッサによって実行されたときに、前記少なくとも１つのプロセッサに、先行する態様の何れかに記載の方法を実施させる命令を記憶しているコンピュータ読み取り可能媒体。

【０１８０】

前述の実施形態の説明は、例示及び説明を提供することを意図するものであり、網羅的であることを意図するものでも、開示を開示した正確な形態に限定することを意図するものでもない。種々の変更及び変形が、上述の教示に照らして可能であり、あるいは、本開示の実施（practice）から得られる場合がある。例えば、一連のアクションが、フロー図に関連して説明されているが、実施形態によっては、アクションの順序は、変更されてもよい。また、アクション、動作、及びステップは、追加の又は他のモジュール又はエンティティによって実行されてもよく、それらは、他のモジュール又はエンティティを形成するように結合又は分離されていてもよい。さらに、依存性を持たない複数のアクションは、並行に実施されてもよい。また、本明細書で使用される「ユーザ」という用語は、特に断りがない限り、例えば、コンピュータシステム若しくはデータ処理システム、又はコンピュータシステム若しくはデータ処理システムの人間のユーザを含むように、広く解釈されることが意図されている。

【０１８１】

さらに、本開示の特定の実施形態は、１以上の機能を実施するロジックとして実施される場合がある。このロジックは、ハードウェアベースであっても、ソフトウェアベースであってもよく、あるいは、ハードウェアベースとソフトウェアベースの組み合わせであってもよい。ロジックの一部又は全部は、１以上の有形の非一時的コンピュータ読み取り可能記憶媒体に記憶される場合があり、コンピュータ１３００のようなコンピュータシステム又はデータ処理システムによって実行することができる種々のコンピュータ実行可能命令を含む場合がある。コンピュータ実行可能命令は、本開示の１以上の実施形態を実施する種々の命令を含む場合がある。有形の非一時的コンピュータ読み取り可能記憶媒体は、揮発性であっても、不揮発性であってもよく、例えば、フラッシュメモリ、ダイナミックメモリ、リムーバブルディスク、及び非リムーバブルディスクを含む場合がある。

【０１８２】

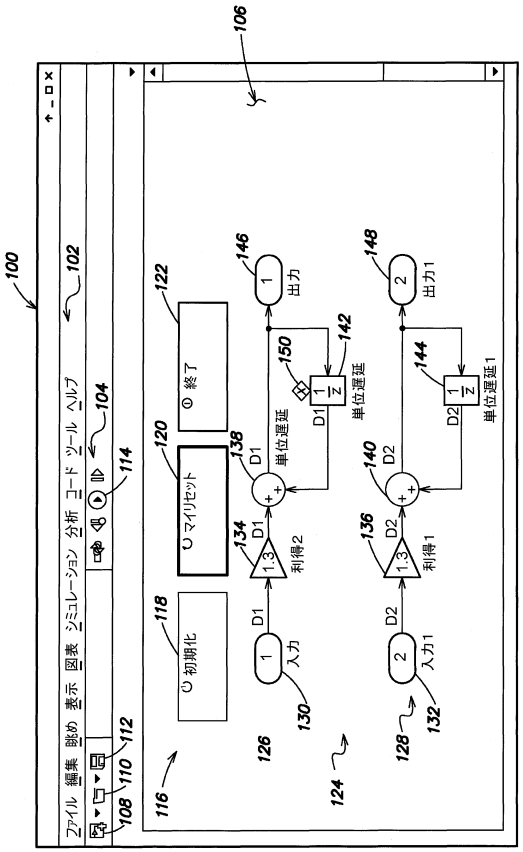
特に明示的断りがない限り、本明細書で使用される種々の要素、アクション、又は命令を、本開示にとって重要又は必須であると解釈してはならない。ただ一つの品目を意図す

る場合、「１つ」又は類似の語が使用される。また、特に明示的断りがない限り、「基づく」という語は、「少なくとも部分的に基づく」ことを意味することを意図している。

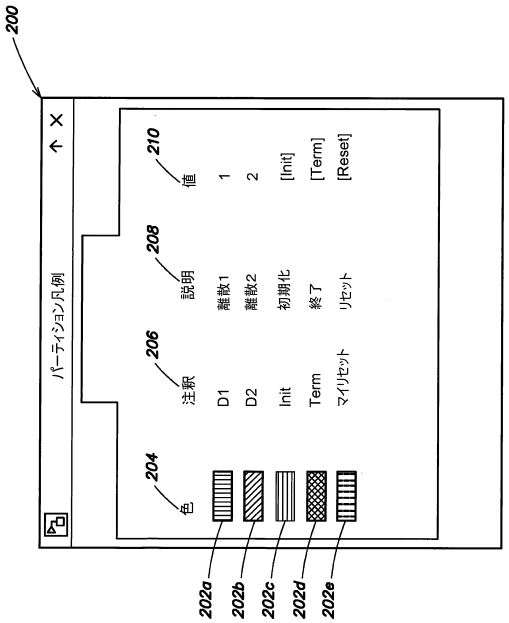
【０１８３】

前述の説明は、本開示の特定の実施形態を対象としている。しかしながら、それらの利点の一部又は全部を達成しつつ、他の変形または修正を、記載した実施形態に変更を加えてもよいことは、明らかである。例えば、モデルの機能は、アダプテーション層インターフェースの公開されたアクセスポイントにグラフィカルにリンクされるものとして例示されているが、実施形態によっては、モデルの機能は、テキストのような他の形でリンクされてもよい。したがって、添付の特許請求の範囲の目的は、そのような変形や修正をすべて、本開示の真の思想及び範囲の中に入るように包含することである。

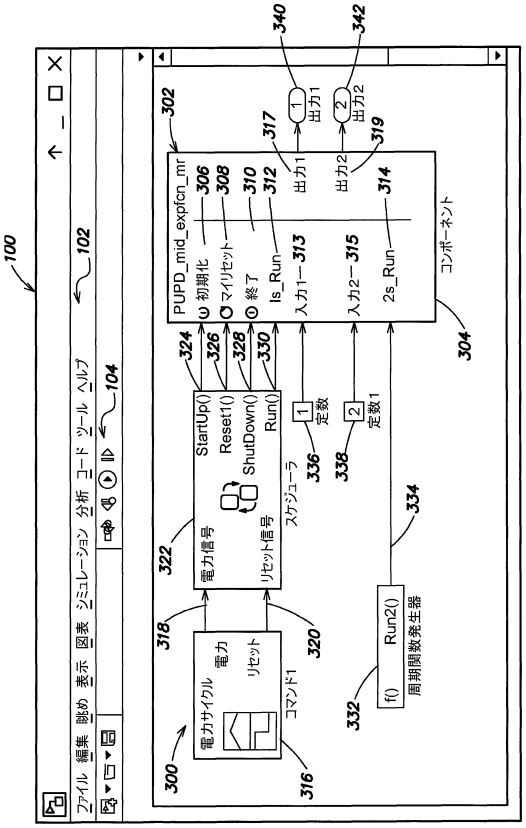
【図１】



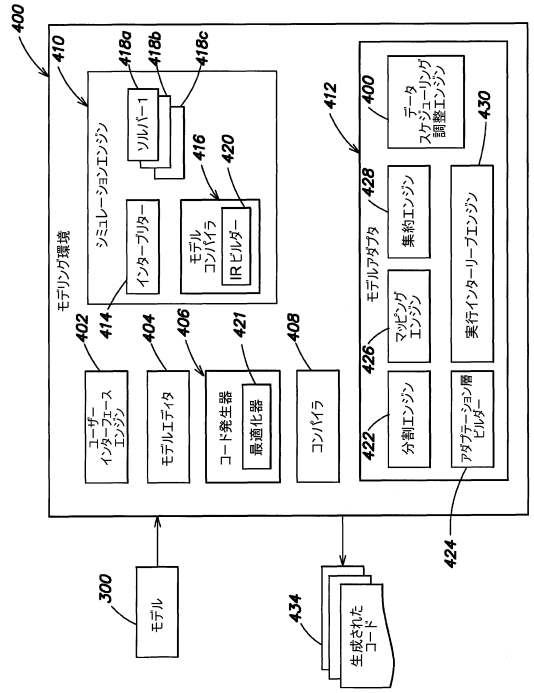
【図２】



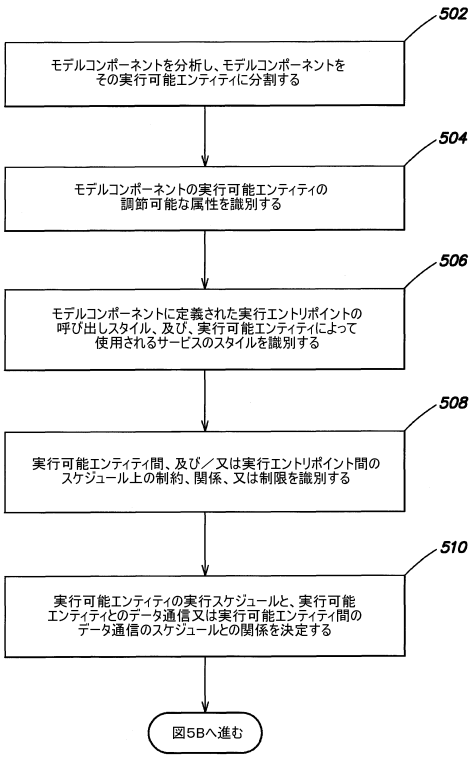
【図３】


100
102
104
ファイル 編集 眺め 表示 図表 シミュレーション 分析 コード ツール ヘルプ
300
302
304
コンポーネント
PUPD_mid_expfcn_mr
306 初期化
308 マイリセット
310 終了
312 Is_Run
313 入力1
314 2s_Run
315 入力2
317 出力1
319 出力2
340
342
322 電力信号
リセット信号
スケジューラ
StartUp()
Reset1()
ShutDown()
Run()
324
326
328
330
336 定数
338 定数1
334
318 電力
320 リセット
電力サイクル
コマンド1
316
332 f()
Run2()
周期関数発生器

【図４】


400
410
モデリング環境
シミュレーションエンジン
418a ソルバー1
418b
418c
インタープリター
モデル コンパイラ
IR ビルダー
416
420
412
モデルアダプタ
データ スケジューリング 調整エンジン
集約エンジン
マッピング エンジン
分割エンジン
実行インターリーブエンジン
アダプテーション層 ビルダー
428
426
422
430
402 ユーザー インターフェース エンジン
414
404 モデルエディタ
406
421 コード発生器
最適化器
408 コンパイラ
300 モデル
434 生成された コード
424

【図５Ａ】


502
モデルコンポーネントを分析し、モデルコンポーネントを その実行可能エンティティに分割する
504
モデルコンポーネントの実行可能エンティティの 調節可能な属性を識別する
506
モデルコンポーネントに定義された実行エントリポイントの 呼び出しスタイル、及び、実行可能エンティティによって 使用されるサービスのスタイルを識別する
508
実行可能エンティティ間、及び／又は実行エントリポイント間の スケジュール上の制約、関係、又は制限を識別する
510
実行可能エンティティの実行スケジュールと、実行可能 エンティティとのデータ通信又は実行可能エンティティ間の データ通信のスケジュールとの関係を決定する
図5Bへ進む

【図５Ｂ】

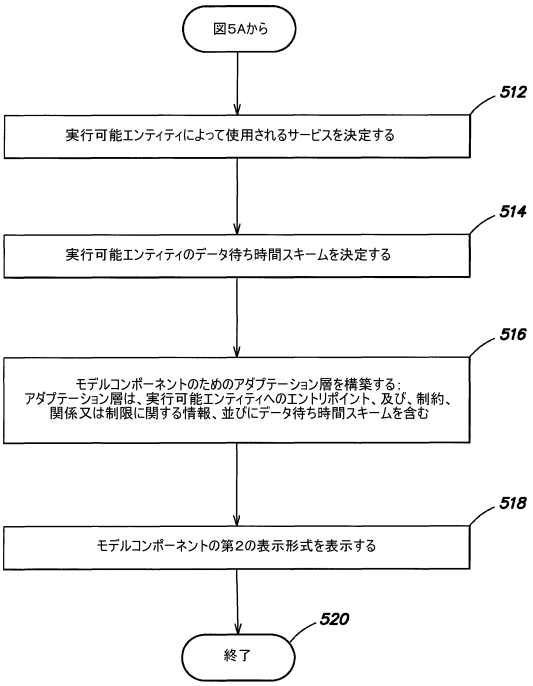
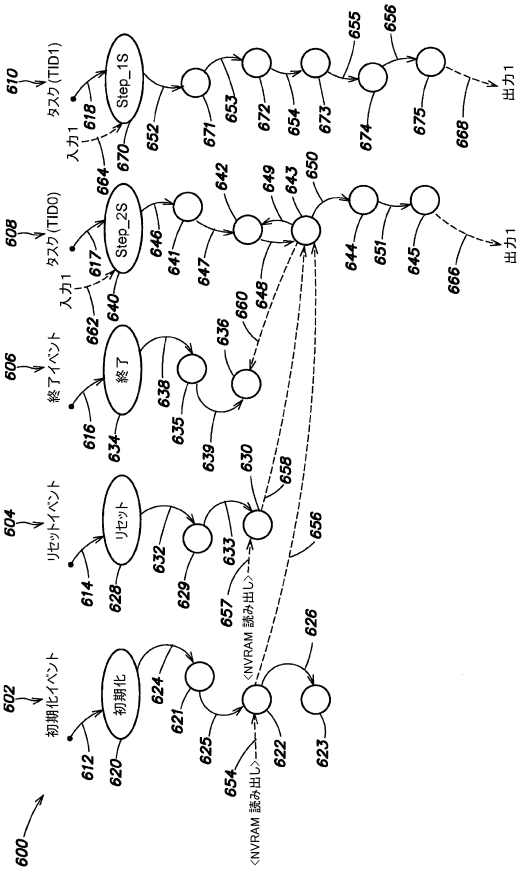
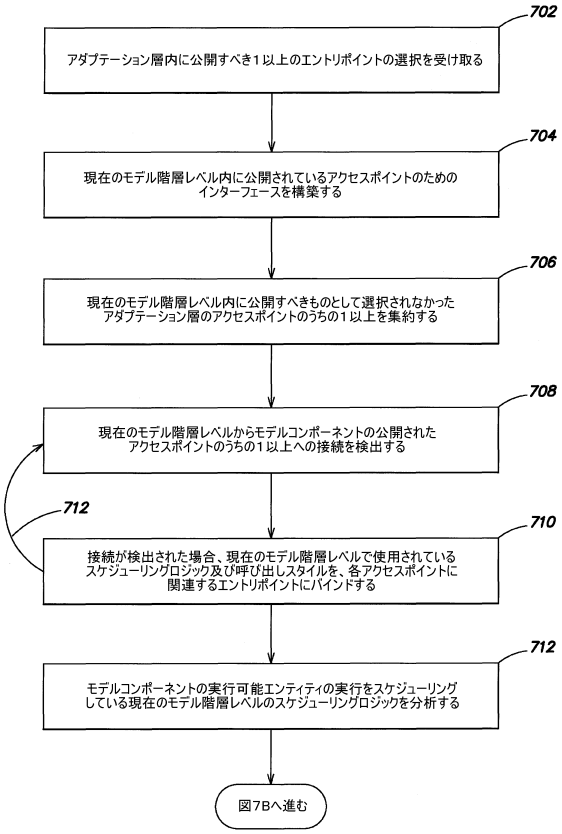


図5Aから
512
実行可能エンティティによって使用されるサービスを決定する
514
実行可能エンティティのデータ待ち時間スキームを決定する
516
モデルコンポーネントのためのアダプテーション層を構築する； アダプテーション層は、実行可能エンティティへのエントリポイント、及び、制約、 関係又は制限に関する情報、並びにデータ待ち時間スキームを含む
518
モデルコンポーネントの第2の表示形式を表示する
520
終了

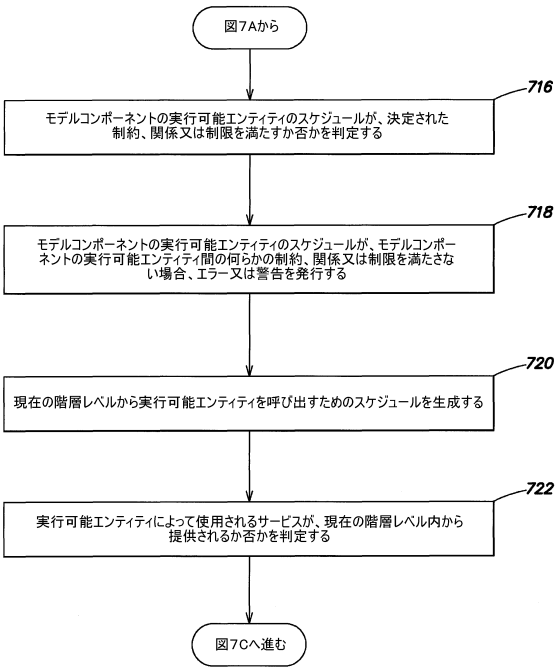
【図６】



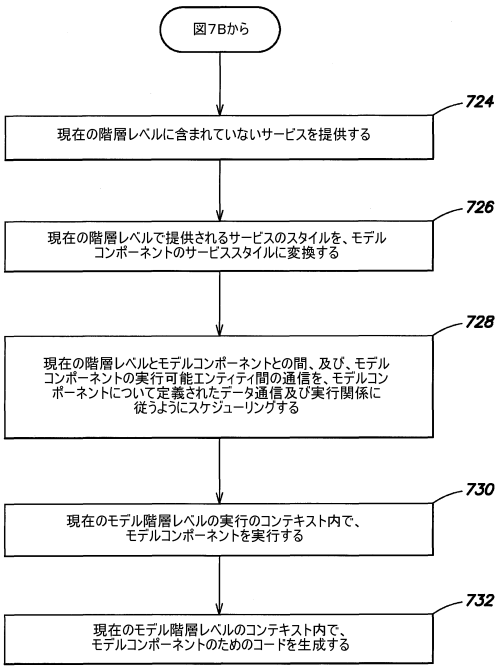
【図７Ａ】



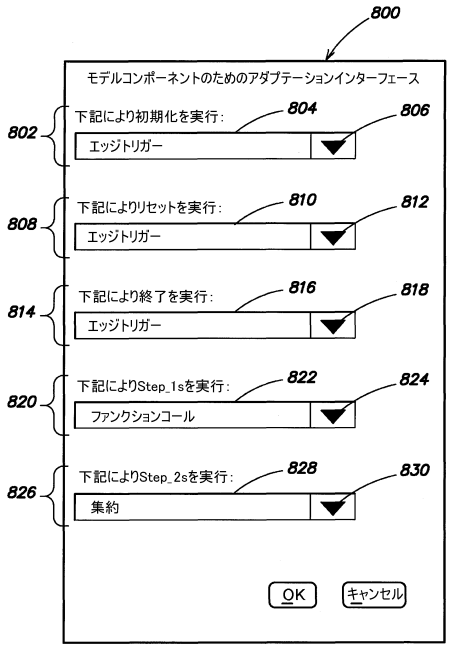
【図７Ｂ】



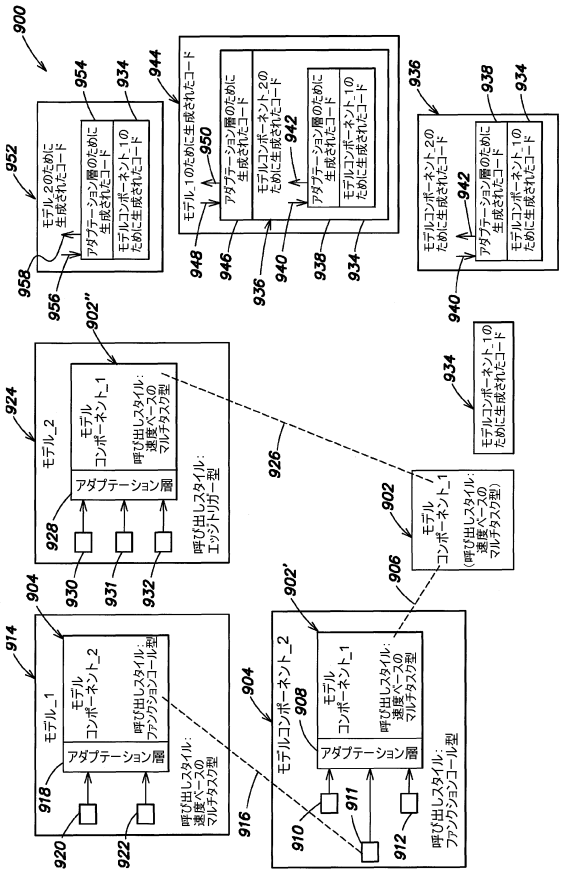
【図７Ｃ】



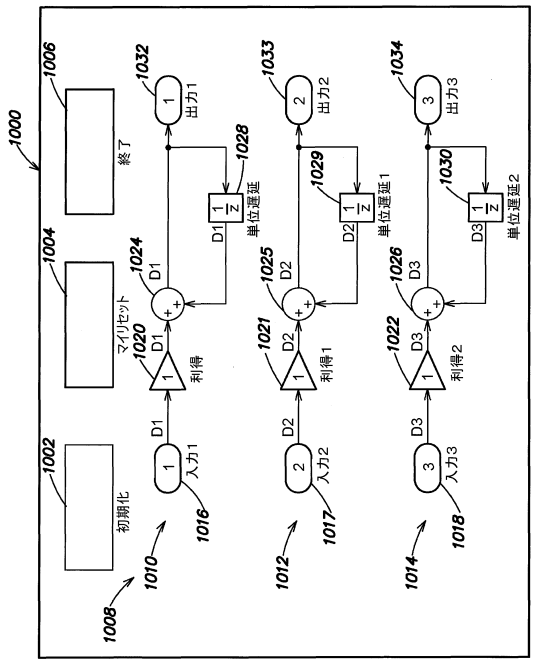
【図８】


800
モデルコンポーネントのためのアダプテーションインターフェース
802
下記により初期化を実行:
804
806
エッジトリガー
808
下記によりリセットを実行:
810
812
エッジトリガー
814
下記により終了を実行:
816
818
エッジトリガー
820
下記によりStep_1sを実行:
822
824
ファンクションコール
826
下記によりStep_2sを実行:
828
830
集約
OK
キャンセル

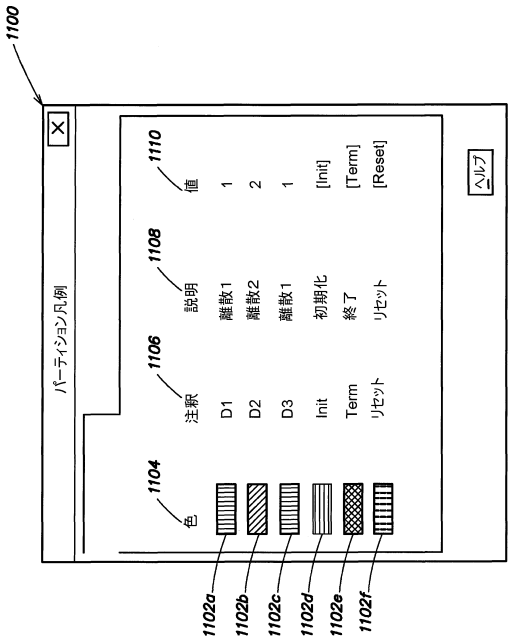
【図９】


900
モデル_2のために生成されたコード
アダプテーション層のために生成されたコード
モデルコンポーネント_1のために生成されたコード
モデル_1のために生成されたコード
モデルコンポーネント_2のために生成されたコード
902''
902'
902
モデル_2
モデル_1
モデルコンポーネント_2
モデルコンポーネント_1
アダプテーション層
呼び出しスタイル:速度ベースのマルチタスク型
呼び出しスタイル:エッジトリガー型
呼び出しスタイル:ファンクションコール型
926

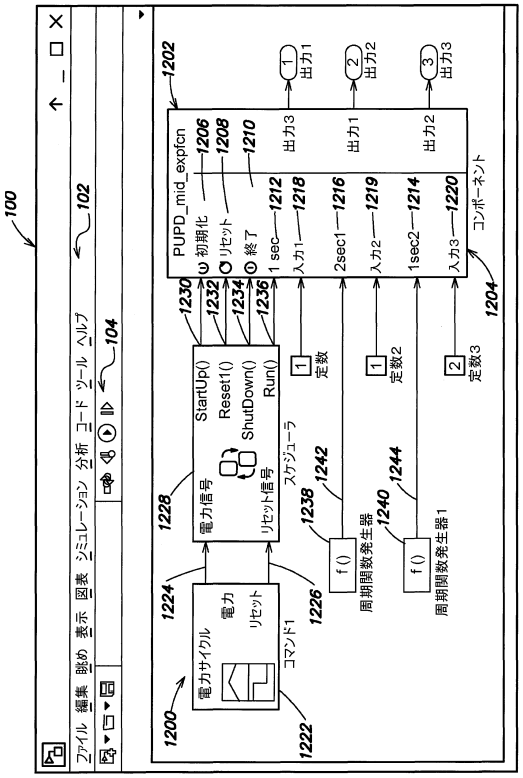
【図１０】


1000
1002
初期化
1004
マイリセット
1006
終了
1008
1010
1012
1014
1016
入力1
1017
入力2
1018
入力3
1020
1021
1022
利得
利得1
利得2
1024
1025
1026
1028
単位遅延
1029
単位遅延1
1030
単位遅延2
1032
出力1
1033
出力2
1034
出力3

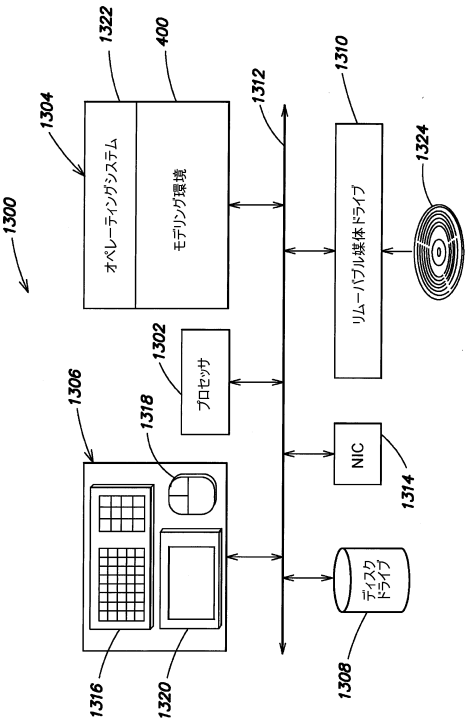
【図１１】


1100
パーティション凡例
1102a
1102b
1102c
1102d
1102e
1102f
1104
色
1106
注釈
D1
D2
D3
Init
Term
リセット
1108
説明
離散1
離散2
離散1
初期化
終了
リセット
1110
値
1
2
1
[Init]
[Term]
[Reset]
ヘルプ

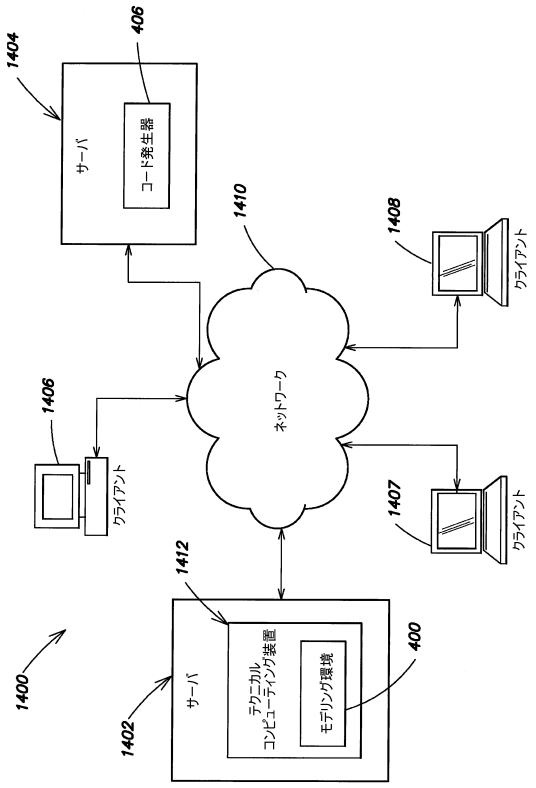
【図１２】


100
102
104
ファイル 編集 眺め 表示 図表 シミュレーション 分析 コード ツール ヘルプ
1200
電力サイクル
電力
リセット
コマンド1
1222
1224
1226
1228
電力信号
リセット信号
StartUp()
Reset1()
ShutDown()
Run()
スケジューラ
1230
1232
1234
1236
1238
1240
1242
1244
f ()
周期関数発生器
f ()
周期関数発生器1
1202
PUPD_mid_expfcn
初期化
リセット
終了
1 sec
2sec1
1sec2
入力1
入力2
入力3
1206
1208
1210
1212
1214
1216
1218
1219
1220
出力3
出力1
出力2
コンポーネント
1204
1
定数
1
定数2
2
定数3
1
出力1
2
出力2
3
出力3

【図１３】


1300
1304
オペレーティングシステム
1322
400
モデリング環境
1312
1310
1324
リムーバブル媒体ドライブ
1302
プロセッサ
1306
1318
NIC
1314
ディスクドライブ
1308
1316
1320

【図１４】


1400
1404
サーバ
406
コード発生器
1410
ネットワーク
1408
クライアント
1406
クライアント
1407
クライアント
1412
サーバ
テクニカルコンピューティング装置
モデリング環境
400
1402

フロントページの続き

(74)代理人 100195693
弁理士 細井 玲
(72)発明者 シュパーク，ピーター，エス
アメリカ合衆国マサチューセッツ州０２４６５，ニュートン，ソマーセット・ロード，４１
(72)発明者 ユ，ビアオ
アメリカ合衆国マサチューセッツ州０２０６７，シャロン，アサートン・レーン・１０
(72)発明者 チュティナン，アロングクリト
アメリカ合衆国マサチューセッツ州０２０３０，ドーバー，ティスデール・ドライブ・サークル・２

審査官 石川 亮

(56)参考文献 米国特許出願公開第２０１５／００５８７７２（ＵＳ，Ａ１）
特開平１１－３５３１６８（ＪＰ，Ａ）
特開２００９－２３８２２９（ＪＰ，Ａ）
特表２０１６－５０６５５０（ＪＰ，Ａ）
米国特許出願公開第２０１４／０３５９５６０（ＵＳ，Ａ１）
米国特許出願公開第２００４／０２１０５９２（ＵＳ，Ａ１）
特開２０１５－０５６１４０（ＪＰ，Ａ）
Robert H. Bishop，ＬａｂＶＩＥＷ ２０１０ プログラミングガイド，第1版，株式会社アスキー・メディアワークス，2011年03月07日，p.10-11,74-83
佐藤 征宏，外3名，ＵＭＬを用いたモデル駆動開発の実践例，Ｄｅｓｉｇｎ Ｗａｖｅ ＭＡＧＡＺＩＮＥ，ＣＱ出版株式会社，2007年02月01日，第１２巻，第２号，p.141-149

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／４４－９／４５５
Ｇ０６Ｆ ８／３５