OpenKasugai Controller

v1.1.0

用語定義①

用語	説明
ファンクションチェイン	データ処理フローを構成する機能と機能間の接続関係の定義。テンプレートとして用意されている場合もある。Function Chain、FCとも記載する
ファンクション	ファンクションチェインに定義される1つの機能のことであり、データ処理モジュールのこと
データフロー	ファンクションチェインを物理リソースにデプロイするものであり、デプロイしたもの。Data Flow、DFとも記載する
ユーザ	本システムの利用者。3種類のユーザがいる ① DF実行管理者（DFの配備を要求する人＋DFへの入力とDFからの出力を受け取るアプリを管理する人） ② 本システムの運用者（オペレータ。DF実行管理者にDFを提供する人） ③ 開発者（DFに必要な機能(ファンクションなど)を開発する人）
GW	DF実行管理者に見える入口・出口（KafkaやRabbitMQ、NetScalerのようなイメージ。クラウドのGWサービスになる場合もあるだろう）
StartPoint(SP)、 EndPoint(EP)	データフローの送信元・宛先のこと。現在はカメラ・配信サーバのこと。 将来的には入口GW・出口GWのことなどに拡張し、セッションの切り張りなどの機能を持つ予定
切り替え	データフローの付け替え処理
スケジューリング条件	データフローを配備する際の配備先に関する指定（例：配備先ノード/デバイスの直接指定(や候補からの削除)やフィルタ実行戦略） データフロー配備要求時に定義済のスケジューリング条件を選択することで、当該条件に沿って配備先が決定される
フィルタ実行戦略	スケジューリング時に適用するフィルタに関する指定（適用するフィルター覧やその順番、フィルタを通過した配備先候補の上位何個を採用するか等）

用語定義②

用語	説明
親bs (親bitstream)	FPGAの基本となる回路。以下の子bsを書込んで初めて使用できる
子bs (子bitstream)	親bsの上に書込む実際のFPGAの回路（例：リサイズ回路）
子bs書込み	下記の3パターンの子bsの書込みの総称
自動書込み	子bs未書込み状態のFPGAが配備先に選択された場合、動的に子bsを書込んでからDFのファンクションを配備すること
手動書込み	運用中にコントローラを停止させることなく、子bs未書込み状態のFPGAに対して子bsを書込み、コントローラに新たな配備先候補として認識させること
上書き	子bs書込み済のFPGAに子bsを書込むこと
リセット	下記の2パターンのFPGAのリセットの総称
FPGAのリセット	子bs未書込み状態のFPGAに戻すこと
子bsのリセット	今書込まれている子bsを書込み直後の初期状態に戻すこと。DF配備時に設定した接続情報なども初期化する
FPGAリソースの解放	DF配備に伴い消費していたFPGAのリソースを解放して未使用状態に戻すこと。K8sリソースやコントローラとは関係ないFPGA側の操作である
分岐	下記の2パターンの分岐の総称。ただし、文脈によっては「コピー分岐」を略している場合もある
コピー分岐	1つのデータを複数の宛先にコピーして送る
条件分岐	1つのデータを特定(複数の場合もあり)の宛先に送る
統合	複数の処理結果を1つにして流す
Glue	入出力の接続種別が異なるファンクションで、接続種別の変換を行うファンクション（例：TCP→DMA） 接続種別が異なるファンクションの間に配備することで、(本来接続できない)それらのファンクションを繋いだDFを配備できる

補足）本コントローラでのリセットの実装について

- FPGAや子bsのリセット処理に関して、本コントローラの実装での想定処理は以下の通りである
  - 子bsを上書きすることでDF配備時に設定された情報を初期化する
    - FPGAのリセット処理として、ダミーの子bsで上書きする
    - 子bsのリセット処理として、書込み済の子bsと同じ子bsで上書きして同じパラメータを再設定する
- 想定するリセット処理後の状態や動作は以下の通りである

		FPGAのリセット	子bsのリセット
FPGA側	書込まれている子bs	ダミーの子bs	元々書込まれていた子bsと同じ子bs
	設定されている子bsのパラメータ	未設定	元々設定されていたパラメータと同じ値
コントローラ側	FPGAリソースの情報例	FPGAの基本情報	FPGAの基本情報 書込み済の子bsリソース名
	子bsリソースの情報例	存在しない	書込み済の子bsの基本情報 設定済のパラメータ値 上記により決まる「ファンクション名」（例：デコード） チャネルの使用状況（全チャネルが未使用）
	配備先候補としての認識	子bs未書込みのFPGA	子bs書込み済のFPGA
	DF配備時のスケジューラ動作	配備したいファンクションの対応デバイスに一致するかを確認	配備したいファンクション名に一致するかを確認
	DF配備時のコントローラ動作	配備したいファンクションに応じた子bsを自動書込みし、 その後、配備処理を実施	配備処理を実施

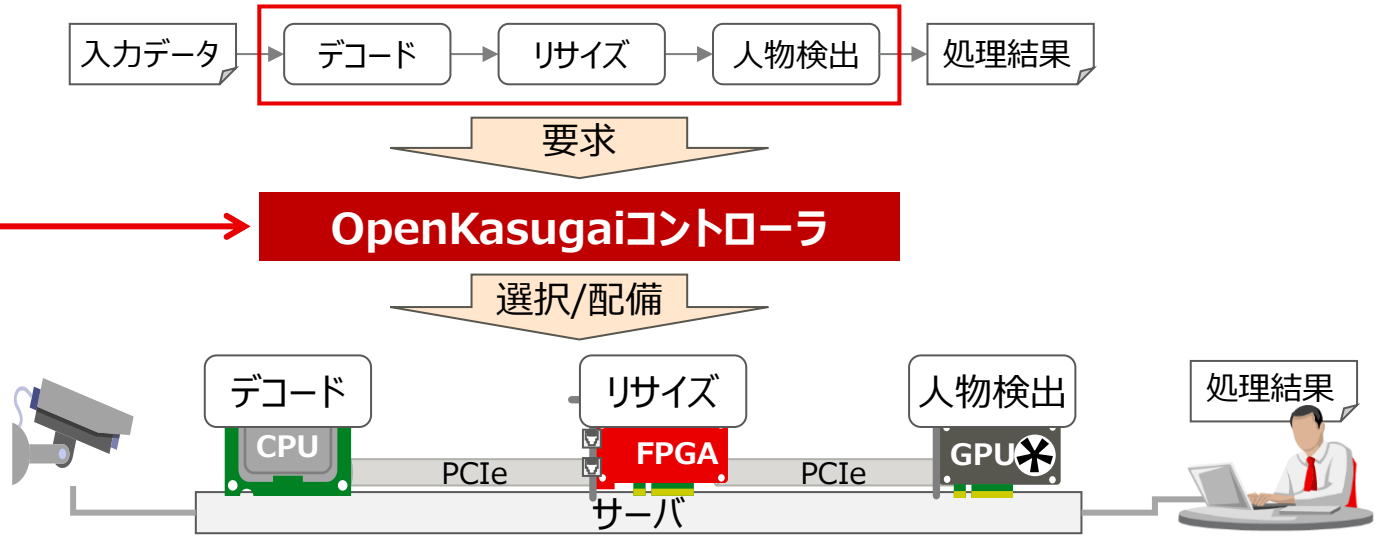
目次

1. はじめに

目的とアプローチ

- アクセラレータを活用したデータ処理を容易に使えるようにする
  - 実績のあるアクセラレータ処理を抽象的な機能ブロックとしてカタログ化
  - カタログ化した処理を繋げて必要なデータ処理をアクセラレータを活用して実現
  - 適切なアクセラレータ・接続種別を選択して配備/接続

処理カタログ		
機能	処理内容	アクセラレータ
映像前処理	デコード	CPU
	リサイズ	FPGA
		CPU
	平滑化	FPGA
	グレースケール	GPU
推論処理	人物検出	GPU
	物体検出	GPU
…	…	…



目指している提供価値

- データフローを自動配備できる
  - ユーザが意識しなくてもリソース容量超過による通信失敗・性能劣化を避ける
    - = DFを配備する配備先のリソース管理を行う
  - ユーザが意識しなくても性能や消費電力が良いDFを配備する
    - = 性能や消費電力を考慮した、より良い使用デバイスや経路の選択を行う
  - (ユーザが望むならば) DFごとに異なるスケジュール戦略で配備できる
    - = 色々なスケジューリング条件を事前に用意。DF定義内でユーザが選択した条件に沿って配備先を選択する

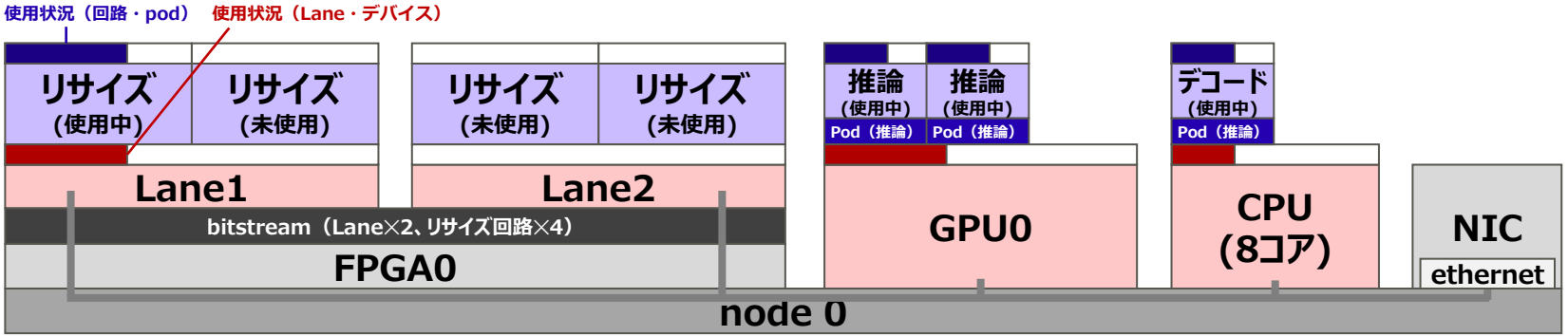
- 環境変更時のユーザの手作業による変更量を削減する
  - 環境に依存する情報をインフラから自動収集・リソース登録することで、ユーザの手作業を回避する
  - 環境やDFに依存しない情報を共通情報化して環境やDFに応じた自動補完を行うことで、ユーザの作業量を削減する

2. OpenKasugaiコントローラの概要

対象とするハードウェア構成

デバイス	ハードウェアの構成	データフロー配備時の構成
FPGA	• 1つのFPGAを2つのLane(領域)に分割し、Lane上で複数の回路を動かす • ハード側の制約により複数Lane・複数回路が一括で書込まれる	• 1回路に対して複数データフローが割り当てられる • Lane単位やLane上の回路単位で配備先を選択する
GPU	• 1つのGPU上に複数のpodを配備する	• 1podに対して1データフローが割り当てられる • スケジューリングはGPU単位で配備先を選択する
CPU	• サーバ上の全CPUコアをまとめて管理する	• 1podに対して1データフローが割り当てられる • スケジューリングは全CPUコアを一括で選択する • 実際にどのコアを使用するかはK8sに任せる

- 参考）ファンクション定義
  - ファンクションを登録する際、配備先の領域(Lane/GPU機種/CPU)を意識して、どの領域に配備可能かや必要資源量・性能を登録しておく
  - ファンクションごとに使用するI/F(IO)種類(TCP/DMA)が決まっており、どれを使用するのかも登録しておく。そのIOが使用できない配備先は選択しない



配備できるデータフロー

- 使用したいファンクションをユーザが指定して配備できる。以下のファンクションを想定する
  - デコード：CPU版が存在
  - リサイズ：FPGA版とCPU版の2種類が存在。後段の推論に合わせて入出力サイズを設定可能
  - 推論：GPU版の高度推論(1280x1280,A100使用)と軽量推論(416x416,T4使用)の2種類存在。
    人物検出と車両検出の2種類存在（計4種類）
- ファンクション間の接続種別としてTCPやDMAが選択できる
- 各ファンクションの配備先を具体的に指定しなくても配備できる
  - DF全体で最適な配備先を選択(スケジューリング)
    - 接続種別や空き容量を考慮して適切な配備先の組合せを選択
    - 複数の接続種別に対応したファンクションを配備する場合、カタログ情報をもとにスケジューラが接続種別自体も適切に選択
    - DF定義の中で適用したいスケジューラ条件名を指定可能。指定した場合は条件に沿った配備先に配備可能
- データフローは複数本の配備を連続要求できる

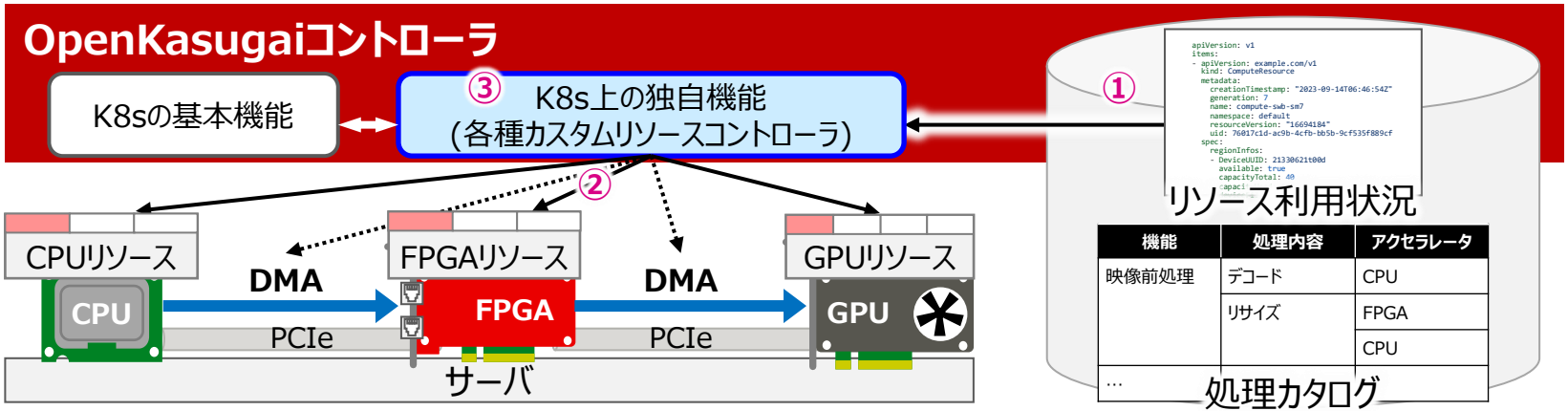
処理カタログ		
機能	処理内容	アクセラレータ
映像前処理	デコード	CPU
	リサイズ	FPGA
		CPU
推論処理	人物検出(高度)	GPU
	人物検出(軽量)	GPU
	車両検出(高度)	GPU
	車両検出(軽量)	GPU

基本コンセプト①

① アクセラレータのリソース利用状況を管理・活用
- アクセラレータのリソース利用状況を管理。利用状況に合わせて適切なアクセラレータ種別や接続種別を選択

② 各種アクセラレータへの処理配備やアクセラレータ間の接続をソフトウェアで制御
- 選択されたアクセラレータ種別や接続種別に合わせた操作を実施

③ Kubernetes(K8s)の機能拡張の仕組み(カスタムリソース)を利用して実現
- データ処理パイプラインやリソース使用状況、各種アクセラレータやアクセラレータ間接続をカスタムリソースで管理



基本コンセプト②

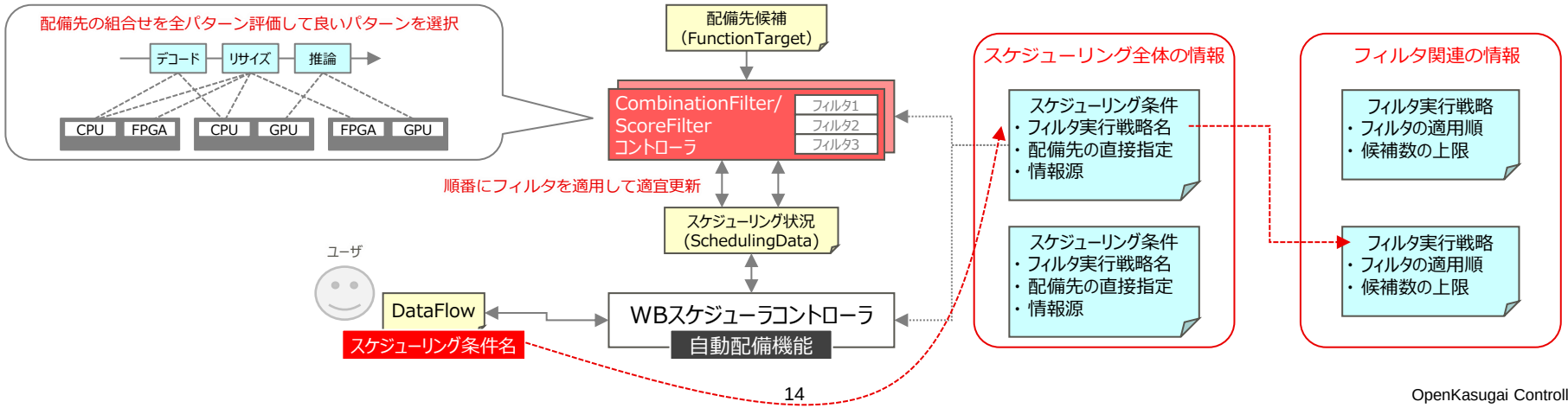
- 宣言値ベースでスケジューリング
  - ノードやアクセラレータの最大処理性能(fps)を管理する
  - DF情報に想定負荷量(fps)が含まれており、スケジューラで搭載可否を判断する
    - 配備先のアクセラレータの空き状況（最大処理性能と現在負荷量の差）をチェックして想定負荷量の収容可否をチェック

- 宣言値ベースでインフラの利用状況を管理
  - DFを配備すると、想定負荷量分だけ、配備先のアクセラレータの現在負荷量を増加する
  - DFを削除すると、反対に想定負荷量分だけそれぞれの現在負荷量を減少する

- 搭載可能なノード・アクセラレータの中から、可能な限り使用中のノード・アクセラレータに片寄するように配備先を選択
  - より詳細な、または、別の配備先選択方法に関しては、次ページのスケジューリング条件での指定も可能である

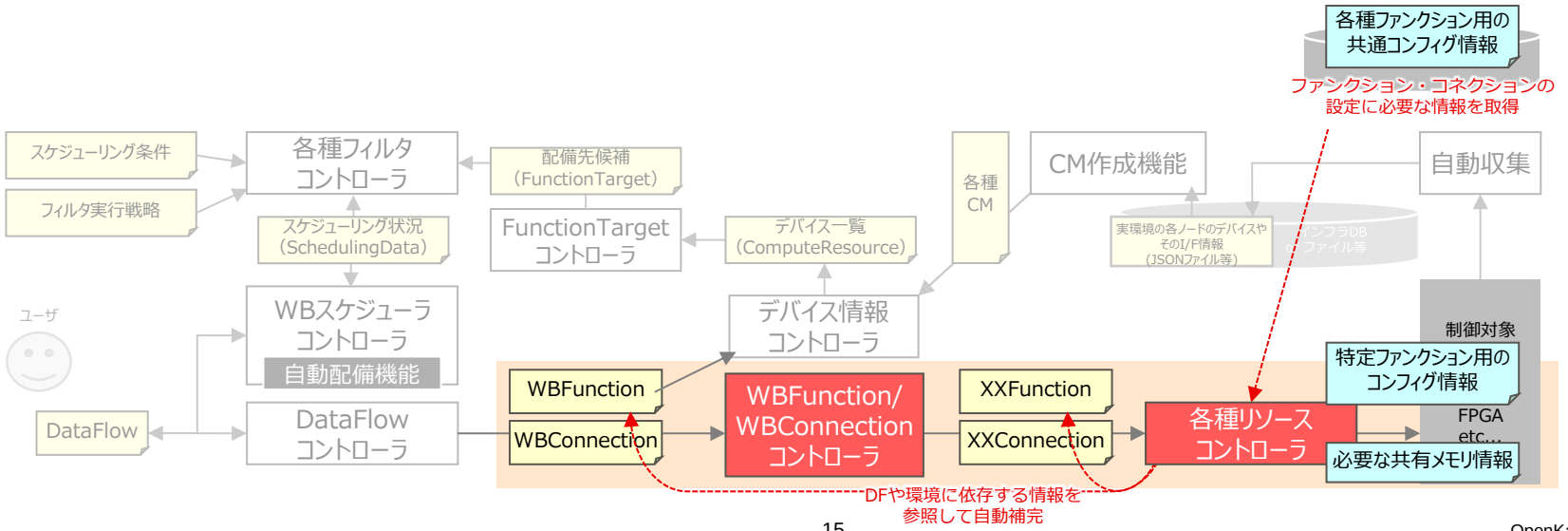
基本コンセプト③

- DF配備要求時にユーザが選択したスケジューリング条件に沿ってスケジューラが配備先を決定
  - スケジューリング条件の例
    - 適用するフィルタリング/スコアリングのプラグインの組合せ
    - フィルタリング/スコアリングの適用後の配備先候補数（上位何個を残すか）
    - 配備先ノードやデバイスの直接指定（許容条件、または、除外条件）
    - 配備先を選択するにあたり参照する情報源の指定（配備先候補の情報が複数管理されている場合）
  - DF定義とは別のリソースにすることで複数DF間でスケジューリング条件を使い回し可能に
    - 将来的には、電力重視、性能重視など様々なスケジューリング条件を用意してユーザが選択可能にする



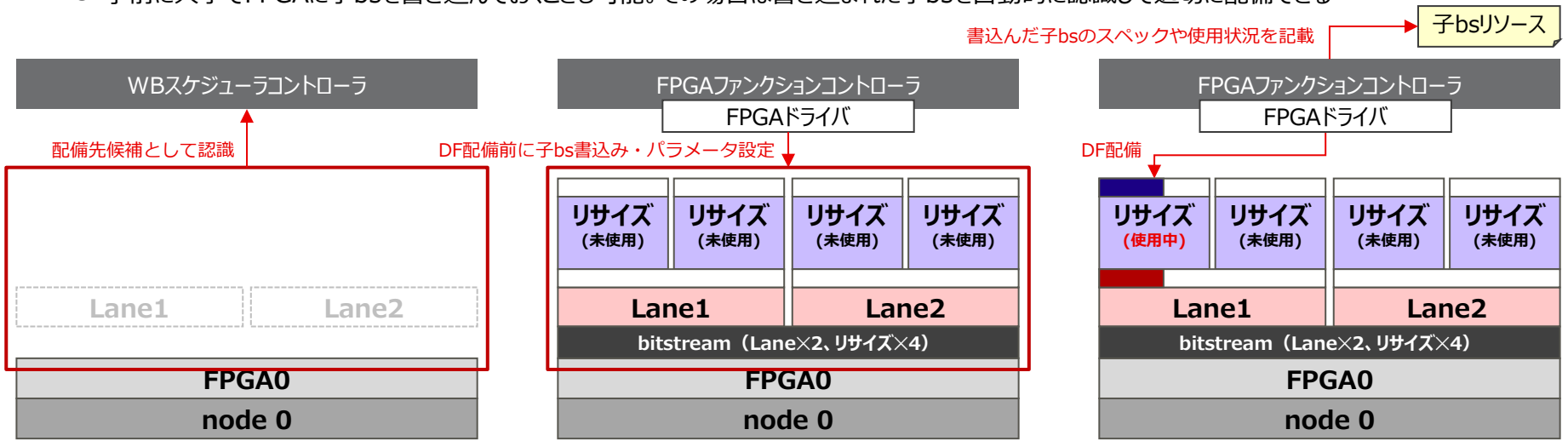
基本コンセプト④

- 各種ファンクション(FPGA・GPU・CPUファンクション)の配備に必要な情報を自動補完
  - 事前用意する情報を環境やDFに依存しない情報にすることで共通化して使い回せるようにする
    - 各種ファンクション用の共通コンフィグ情報
  - 実際の配備に必要な環境やDFに依存する情報は自動補完する
    - 当該ファンクションの前後のファンクションの配備先デバイスの情報や当該ファンクションが属するDFの情報など
    - PCIe接続に必要な共有メモリ情報をDF情報やファンクションのconfigファイル等から自動補完する（共有メモリ名やサイズ）



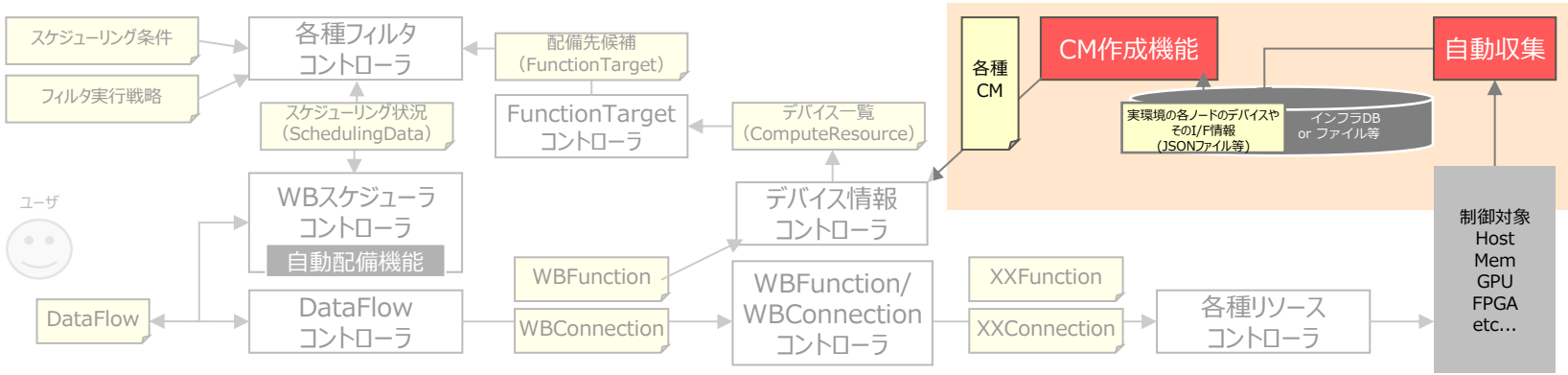
基本コンセプト⑤

- 事前に人手でFPGAに子bsを書き込んでおく必要がなく、運用中に必要になった子bsを動的に書込む
  - 子bs未書込みのFPGAも配備先候補として認識し、スケジューラは子bs書込み/未書込みのFPGAから配備先を選択する
  - 子bs未書込みのFPGAが配備先として選択された場合、配備の過程で動的に子bsを書込み、パラメータ値を設定する
    - ハードの制限により、同一FPGA上の複数Laneは一括書込みが必要なため、動的に子bsを書込む際は全Laneに同一の回路リソースを作成
    - 全Laneに対して、要求されたファンクション用のパラメータ値を一括設定
  - 子bs書込みを行った場合、書込んだ子bsの情報を記載したk8sリソースを作成・管理する
  - 事前に人手でFPGAに子bsを書き込んでおくことも可能。その場合は書き込まれた子bsを自動的に認識して適切に配備できる



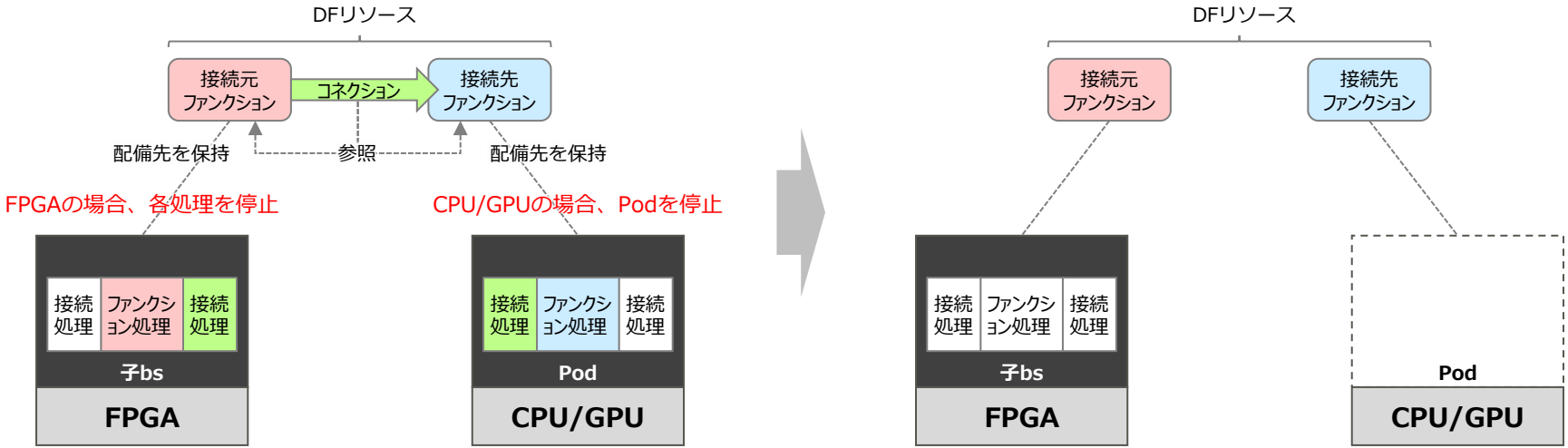
基本コンセプト⑥

- 従来人手で用意していた入力情報を自動収集し、各種コントローラに必要なK8sリソースを作成
  - 環境に依存しない情報の一部を共通化して使い回せるようにする
    - アクセラレータや回路・Podのカタログ情報
  - 環境に依存する情報の一部を自動収集する
    - ノード上のアクセラレーター覧(FPGA/GPU/CPU)。FPGA上に書込まれている領域の構成
  - 上記の情報は共にインフラDBに格納しておき、自動でK8sリソース化する



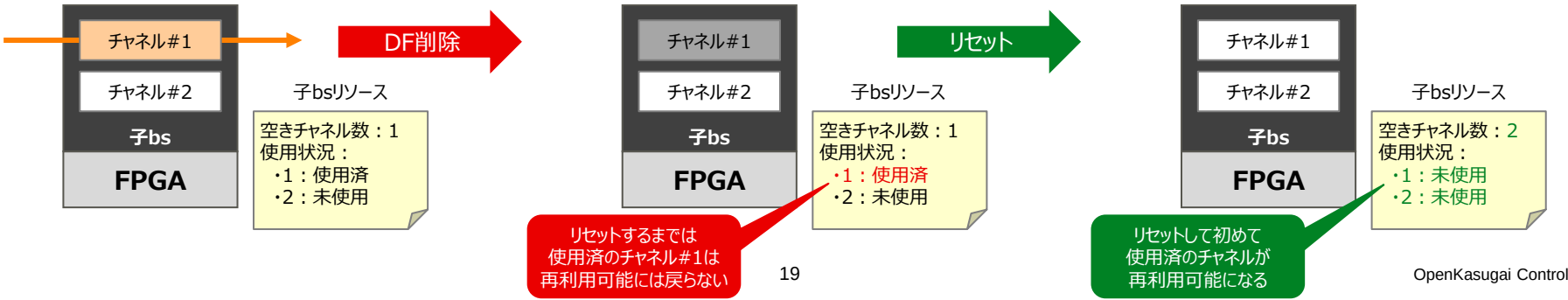
基本コンセプト⑦

- DF削除時に適切にデバイス処理が停止されるように、DF削除に伴う各種リソースの削除順序を管理
  - DFリソースを削除すると、(配備時に作成した)DFを構成するファンクションリソースとコネクションリソースが削除される
  - 適切なデバイス処理の停止のため、接続処理の停止の一環としてFPGAのファンクション処理やCPU/GPU上のPodを停止させる
    - コネクションリソースの削除時に、接続先と接続元のファンクションリソースの情報を参照して、停止させる処理を特定する
  - ファンクションリソースが先に削除された状態だと、(特にFPGAにおいて)どのデバイスのどの処理を停止させるべきか特定できないため、コネクションリソースを先に削除し、削除が完了したのを確認してからファンクションリソースを削除する



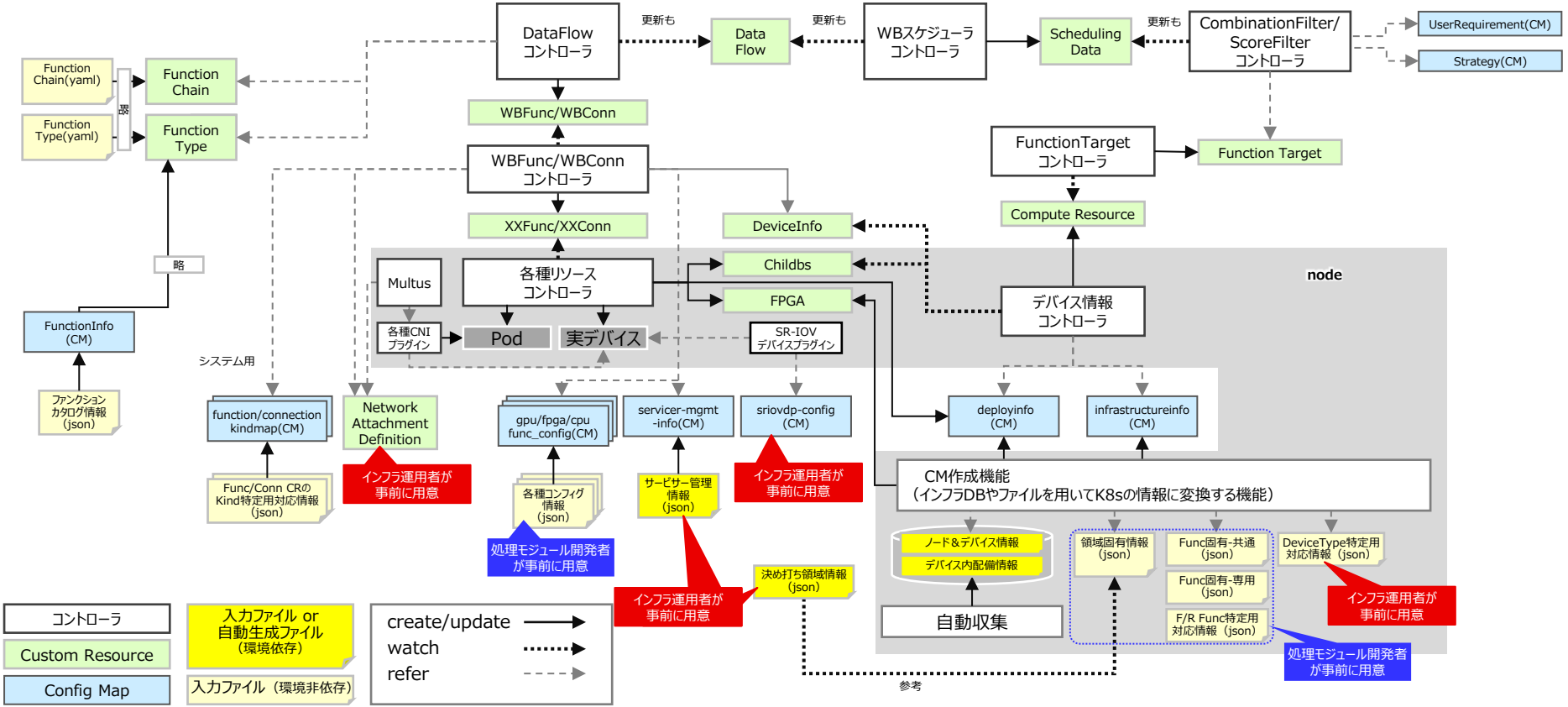
基本コンセプト⑧

- 問題：以前のDF配備時の状態がFPGAのチャネルに残っている場合、そのチャネルを使用してDF配備すると疎通に失敗
  - DF削除に伴い、子bsの使用をやめても、内部に状態が残っている可能性がある
  - FPGAや子bsをリセットすることも考えられるが、配備中の他DFの疎通に影響を与える可能性がある
- 解決策：未使用チャネルと使用済チャネルを区別することで、使用済チャネルが再び使用されることを回避
  1. **DFを削除する場合、使用していたチャネルを今後配備するDFが使わないように、CR上では使用できないチャネルとして管理する**
     - DFを削除する際は、子bsリソースのCR上では、使用していたチャネルは使用済のままとし、空きチャネル数も元に戻さない
     - DFを配備する際は、上記を考慮して、当該子bsの当該チャネルが配備先として選択されないようにする
  2. **チャネルが枯渇したタイミングなどで、FPGAや子bsをリセットし、CR上でも使用できるチャネルに戻す**
     - FPGAや子bsのリセットは、当該FPGAが使用されていない場合のみ実施可能である
     - FPGAや子bsをリセットした際は、当該子bsの全チャネルを未使用状態に戻し、空きチャネル数を初期値に戻す



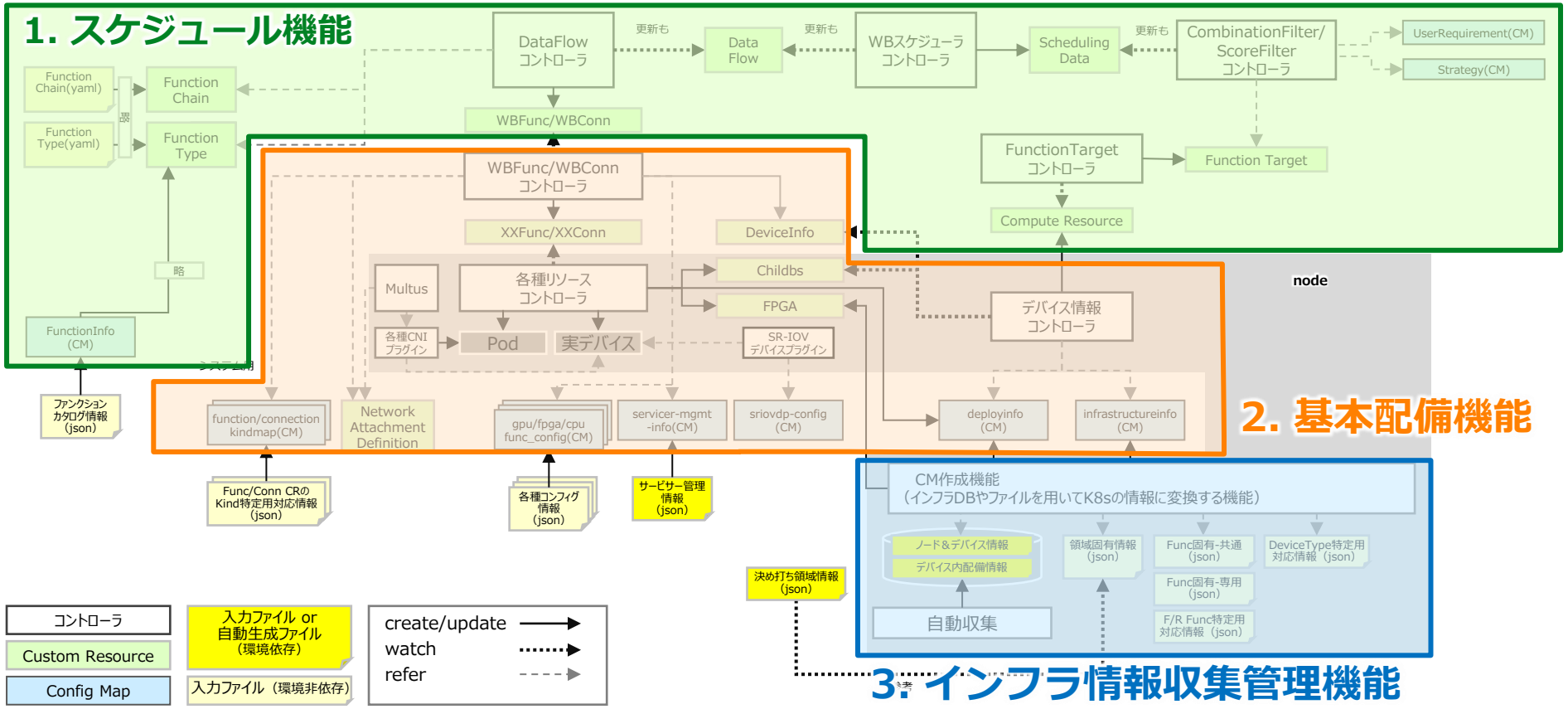
3. OpenKasugaiコントローラの全体構成

OpenKasugaiコントローラの全体構成



各種コントローラやリソースの説明は次頁以降を参照のこと

OpenKasugaiコントローラを構成する3機能



各種コントローラやリソースの説明は次頁以降を参照のこと

OpenKasugaiコントローラを構成する3機能

機能名	説明
スケジュール機能	• データフローの配備要求を受けて、ユーザが選択したスケジューリング条件や現在の使用状況などを考慮して、データフローの各構成要素の配備先を自動決定する • 基本配備機能にデータフローの各構成要素(WBFunction/WBConnection)の配備を要求したり、順序を考慮して削除を要求する • FCのテンプレートの登録や、それに伴うファンクションやコネクションの登録も行える
基本配備機能	• データフローの各構成要素の配備・削除要求を受けて、各種リソースコントローラを用いて実際に配備・削除の設定を行う • 配備先のFPGAに子bsが書込まれていなければ、動的に子bsを書込む • 各種FPGA回路やPodに必要な詳細パラメータはコントローラが自動補完する。例えば、チャネル番号やIPアドレス、ポート番号 • 配備時の情報をもとに各種リソースの削除の設定を行う。使用済リソースが再び使用されないように使用状況を管理する • 配備・削除した結果を各種リソースの使用状況としてスケジュール機能側に提供する • ユーザからの子bsの手動書込み要求や、FPGAや子bsのリセット要求を受けて、処理を実施し、結果を各種リソースの使用状況に反映する

インフラ情報収集管理機能	• インフラ情報を自動収集してK8sリソースを自動作成する機能を提供する
	  • インフラの構成情報や書込まれている領域情報など、環境依存情報の収集とk8s上での活用の自動化
	  • DFに依存する情報(DF毎に異なる情報)の自動補完

構成要素一覧：
コントローラ概要①

	項目	概要	関連機能
CRC	FunctionTypeコントローラ	ユーザの登録情報からFunctionTypeリソースを作成する（図では省略。NTTとの共同研究の検討結果をほぼそのまま使用）	スケジュール機能
	FunctionChainコントローラ	ユーザの登録情報からFunctionChainリソースを作成する（図では省略。NTTとの共同研究の検討結果をほぼそのまま使用）	スケジュール機能
	DataFlowコントローラ	データフローの要求を受けて、データフローを配備・削除する	スケジュール機能
	FunctionTargetコントローラ	ComputeResourceリソース(デバイス情報)から、スケジューラが使用する配備先対象情報(GPU/FPGA/CPU)を作成する	スケジュール機能
	WBスケジューラコントローラ	データフローの要求を受けた際、ユーザが配備先を直接指定していない場合、適切な配備先を自動選択する デバイスの空き容量をチェックし、複数の候補が存在する場合はスコア計算により適切な候補を選択する	スケジュール機能
	CombinationFilterコントローラ	SchedulingDataを参照し、ファンクション種別やリソース状況を考慮して配備先候補の組合せをフィルタリングして、SchedulingDataを更新する	スケジュール機能
	ScoreFilterコントローラ	SchedulingDataを参照し、デバイスのリソース状況を考慮して配備先候補の組合せをスコアリングして、SchedulingDataを更新する	スケジュール機能
	WBFunctionコントローラ	WBFunctionリソースから実体となる各種ファンクションリソース(GPUFunction/FPGAFunction/CPUFunction)を作成・削除する	基本配備機能
	WBConnectionコントローラ	WBConnectionリソースから実体となる各種コネクションリソース(EthernetConnection/PCIeConnection)を作成・削除する	基本配備機能
	各種リソースコントローラ	GPUFunction, FPGAFunction, CPUFunction の配備・削除を行う。 また、EthernetConnection, PCIeConnection に応じたデバイス間の接続用設定やその停止を行う。 FPGAFunctionコントローラに関しては、ユーザからの、子bsの手動書込み要求や、FPGAや子bsのリセット要求を受けて動作することもある	基本配備機能
	デバイス情報コントローラ	以下を実行する ・初期状態のComputeResourceリソース(デバイス情報)を作成する ・配備状況に応じてComputeResourceを更新する	基本配備機能

構成要素一覧：
コントローラ概要②

	項目	概要	関連機能
他	Multus	各種CNIプラグインを呼び出すメタプラグインとして機能する	基本配備機能
	各種CNIプラグイン	Pod用のNICおよびNWの設定を行う。CNIプラグインとして、PodのK8sのオーバレイNW用にCalico(既存)、2nd NICのNW用にSR-IOV CNIプラグイン(新規)を利用。また、2nd NIC用のCNI IPAMプラグインとしてstatic(新規)を利用	基本配備機能
	SR-IOVデバイスプラグイン	各ノードのNICのVFを認識し、K8sのPodから利用可能なリソースとして公開する	基本配備機能
	CM作成機能	インフラDB(JSONファイル)の情報をもとに必要なConfigMapを自動作成・登録する ※1	インフラ情報収集管理機能
	自動収集	インフラから情報を収集してインフラDB(JSONファイル)に格納する ※1	インフラ情報収集管理機能

※1. 現在は１つのツール(InfoCollector)として実装

構成要素一覧：
リソース概要①

	名称	概要	関連機能	手動/自動
CR	DataFlow	配備したいデータフローの構成(ファンクションやコネクションの指定)や要件(想定負荷量)に関する情報を持つカスタムリソース	スケジュール機能	DF実行管理者 or 運用者が作成
	FunctionChain	データフローの構成を表すためのカスタムリソース	スケジュール機能	運用者が作成 or 開発者が作成
	FunctionType	ファンクションチェインで利用可能なFunctionを表すカスタムリソース	スケジュール機能	運用者が作成 or 開発者が作成
	SchedulingData	DataFlowのスケジューリング状況に関する情報を持つカスタムリソース。 スケジューリング中は、現在の配備先候補群やフィルタ適用順、適用済フィルタなどの情報を管理し、コントローラ間で状況を共有する	スケジュール機能	自動
	ComputeResource	各ノードのハード構成や容量管理に関する情報を持つカスタムリソース。ノード上のCPU・GPU・FPGAの情報を表す。 デバイスの容量管理に関する情報も持つ	スケジュール機能	自動
	FunctionTarget	ComputeResourceから作られるファンクションの配備先候補一覧に関する情報を持つカスタムリソース	スケジュール機能	自動
	WBFunction	配備するFunctionに関するカスタムリソース。配備の過程で下記のFJ版のXXXFunctionのいずれかに変換される	スケジュール機能	自動
	WBConnection	配備するConnectionに関するカスタムリソース。配備の過程で下記のFJ版のXXXConnectionのいずれかに変換される	スケジュール機能	自動
	GPUFunction	GPUに配備するFunction(GPU付きPodに配備)に関する情報を持つFJ版のカスタムリソース	基本配備機能	自動
	FPGAFunction	FJ版FPGA(phase3)に配備するFunctionに関する情報を持つカスタムリソース	基本配備機能	自動
	CPUFunction	CPUに配備するFunction(Podに配備)に関する情報を持つFJ版のカスタムリソース	基本配備機能	自動
	EthernetConnection	Ethernet接続に関する情報を持つFJ版のカスタムリソース	基本配備機能	自動
	PCIeConnection	共有メモリ経由のPCIe接続に関する情報を持つFJ版のカスタムリソース	基本配備機能	自動
	DeviceInfo	WBFunctionコントローラとDeviceInfoコントローラ間でやり取りする情報を持つカスタムリソース	基本配備機能	自動
	FPGA	FPGAデバイスに関する情報を持つカスタムリソース。利用状況や書込み・設定の状態管理の情報を持つ	基本配備機能	自動
	Childbs	FPGAに書込まれた子bsに関する情報を持つカスタムリソース。設定されたパラメータ値やDFの配備状況の情報を持つ	基本配備機能	自動
	FPGAReconfiguration	FPGAへの子bsの手動書込みやFPGA・子bsのリセットを要求するためのカスタムリソース。書込む子bsやパラメータ値の情報を持つ	基本配備機能	DF実行管理者 or 運用者が作成
	NetworkAttachmentDefinition	Multusが参照するPodの2nd NIC用の設定情報を持つカスタムリソース。各ノードの100GNIC毎に作成する運用とする	基本配備機能	運用者が作成

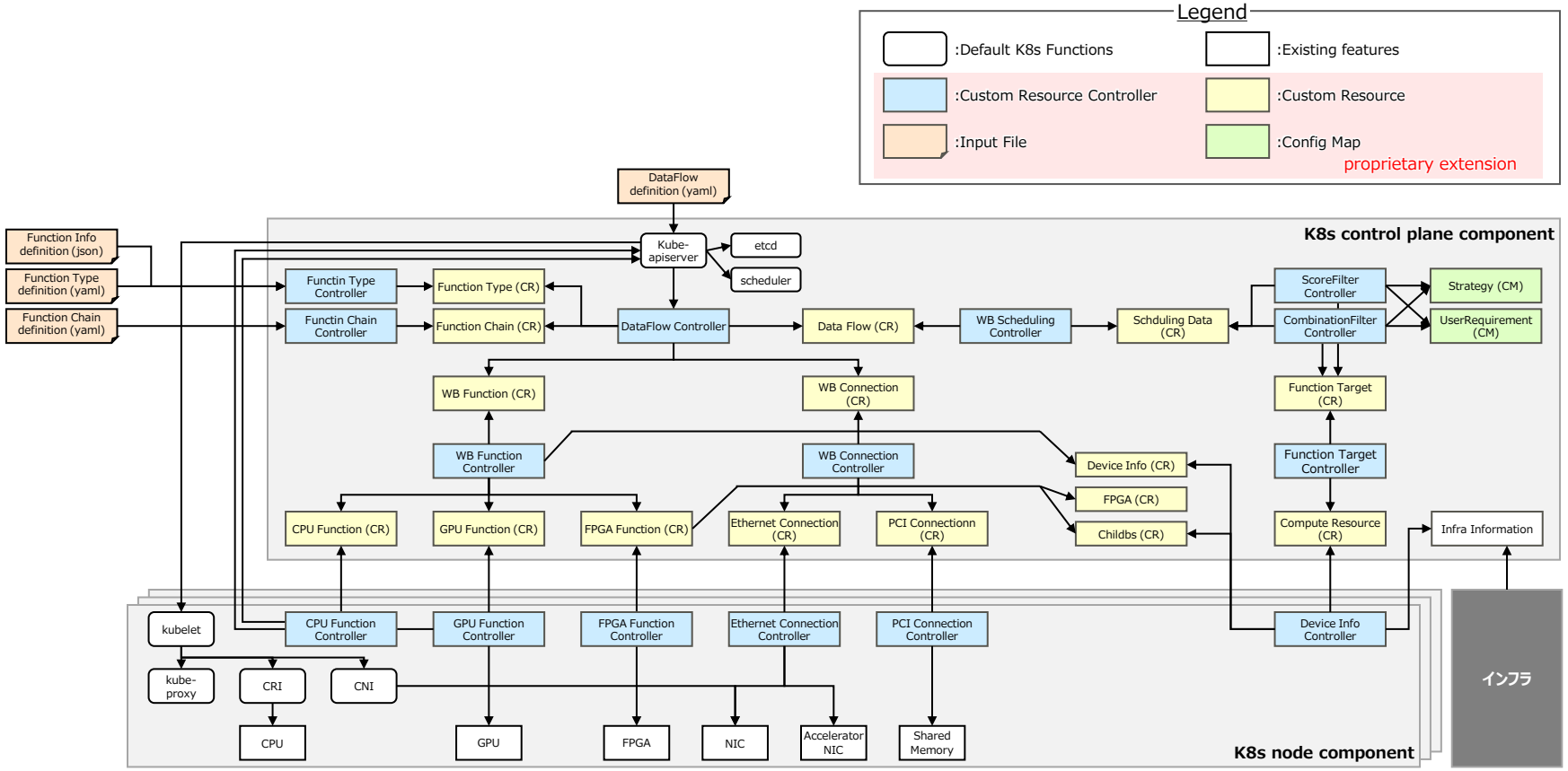
構成要素一覧：
リソース概要②

	名称	概要	関連機能	手動/自動
CM	UserRequirement	DataFlowのスケジューリングに利用するStrategyのコンフィグマップや、ファンクション/コネクションの配備先の絞り込み条件を指定する情報	スケジュール機能	DF実行管理者 or 運用者が作成
	Strategy	DataFlowのスケジューリングにおけるFilterの実行戦略を指定する情報	スケジュール機能	DF実行管理者 or 運用者が作成
	FunctionInfo	ファンクションのカタログ情報に相当する情報	スケジュール機能	運用者が作成 or 開発者が作成
	infrastructureinfo	各ノードのハード構成について定義するための情報。デバイス(GPU, FPGA, CPU)毎に配備領域情報を定義	基本配備機能	自動取集
	deployinfo	各ノードに用意された配備可能領域について定義するための情報。デバイス(GPU, FPGA, CPU)毎に配備領域情報を定義。 デバイスの最大容量に関する情報も持つ	基本配備機能	自動収集
	fpgacatalogmap	各FPGAFunctionに対して払い出すFPGAに関する情報	基本配備機能	自動収集
	functionkindmap	WBFunctionからどのFunction系CR(GPUFunction/FPGAFunction/CPUFunction)に変換すべきかを特定するための情報	基本配備機能	運用者が作成
	connectionkindmap	WBConnectionからどのConnection系CR(EthernetConnection/PCIeConnection)に変換すべきかを特定するための情報	基本配備機能	運用者が作成
	gpufunc_config	GPUFunction用のコンフィグ情報を記載したConfigMap。GPUFunction毎に作成	基本配備機能	運用者が作成
	fpgafunc_config	FPGAFunction用のコンフィグ情報を記載したConfigMap。FPGAFunction毎に作成	基本配備機能	運用者が作成
	cpufunc_config	CPUFunction用のコンフィグ情報を記載したConfigMap。CPUFunction毎に作成	基本配備機能	運用者が作成
	fpgalist-ph3	EthernetConnection起動時に各FPGAに設定する情報(Lane情報とネットワーク情報)を記載したConfigMap	基本配備機能	運用者が作成
	sriovdp-config	SR-IOVデバイスプラグインが管理するデバイスの情報を記載したConfigMap	基本配備機能	運用者が作成

構成要素一覧：
リソース概要③

	名称	関連CM	概要	関連機能	手動/自動
入力・自動生成ファイル (json)	ファンクションカタログ情報	FunctionInfo	各種ファンクションの基本情報（配備可能アクセラレータ、対応接続種別、容量など）	スケジュール機能	運用者が作成
	各種コンフィグ情報	XXXfunc_config	各種ファンクションの全DFで共通で使用可能なコンフィグ。 これに、コントローラ側でDFごとの専用の設定内容を補完することで、実際のコンフィグ情報となる （DFごとに作成する必要が無くなったため、共通のコンフィグは開発者が作成する想定）	基本配備機能	開発者が作成
	FPGA初期設定情報	fpgalist-ph3	FPGA(子bs)に設定するために使う情報。 現状はFPGAの通信部に設定する情報など（MAC, IP, Subnet, Gateway 等）	基本配備機能	運用者が作成
	Func/Conn CRの Kind特定用対応情報	functionkindmap, connectionkindmap	システム側で用意する、スケジュール機能が作成したWBファンクション・WBコネクションが、基本配備機能におけるどのファンクション種別・コネクション種別に対応するかを記載したマッピング情報	基本配備機能	システム固定
	ノード＆デバイス情報	infrastructureinfo	自動収集した実インフラのノード一覧やノードに搭載されているデバイス一覧の情報	インフラ情報収集管理機能	自動取集
	デバイスタイプ特定用対応情報		自動取得したデバイスのモデル名からDeviceTypeに変換するためのマッピング情報	インフラ情報収集管理機能	システム固定
	デバイス内配備情報	deployinfo	自動収集した各デバイスに書き込まれている領域の情報（FPGAの子bs）や各領域に配備中のファンクションの情報	インフラ情報収集管理機能	自動収集
	F/R Func特定用対応情報		FPGAのリサイズファンクションに関して、高度推論向けか軽量推論向けかを判別するためのマッピング情報	インフラ情報収集管理機能	システム固定
	領域固有情報		領域を書き込む回路(子bs)に固有の情報。領域の個数やサイズなど。GPUに関しても同等の情報を保持する	インフラ情報収集管理機能	運用者が作成
	Func固有-共通		回路やコンテナによって固定で決まる情報の内、共通情報（ID, Name, 搭載可能WBFunc数, 容量）	インフラ情報収集管理機能	開発者が作成
	Func固有-専用	fpgacatalogmap	回路やコンテナによって固定で決まる情報の内、デコードやリサイズの専用情報。回路の種類ごとに用意する	インフラ情報収集管理機能	開発者が作成

参考）OpenKasugaiコントローラとK8s機能の関係



4. OpenKasugaiコントローラの機能説明

OpenKasugaiコントローラを構成する3機能（再掲）

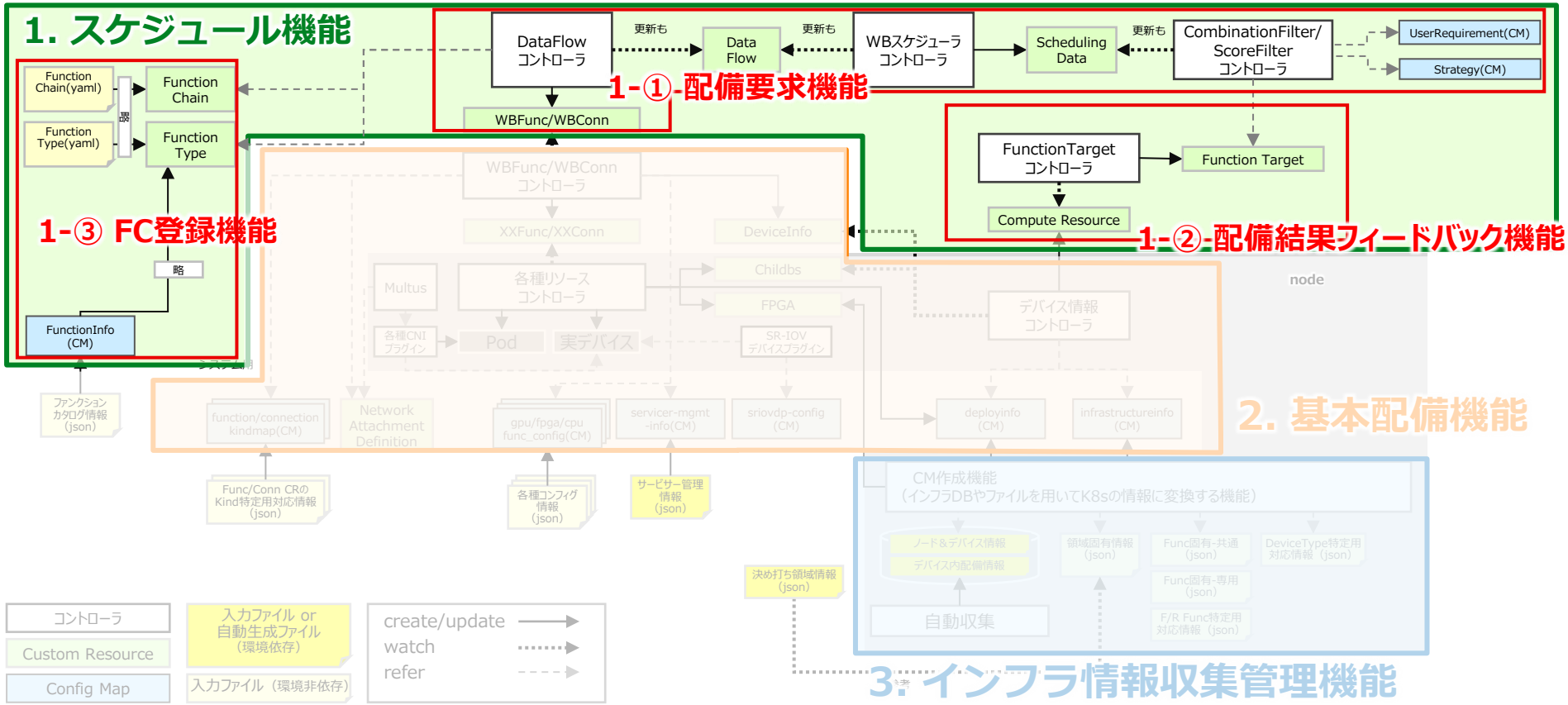
機能名	説明
スケジュール機能	• データフローの配備要求を受けて、ユーザが選択したスケジューリング条件や現在の使用状況などを考慮して、データフローの各構成要素の配備先を自動決定する • 基本配備機能にデータフローの各構成要素(WBFunction/WBConnection)の配備を要求したり、順序を考慮して削除を要求する • FCのテンプレートの登録や、それに伴うファンクションやコネクションの登録も行える
基本配備機能	• データフローの各構成要素の配備・削除要求を受けて、各種リソースコントローラを用いて実際に配備・削除の設定を行う • 配備先のFPGAに子bsが書込まれていなければ、動的に子bsを書込む • 各種FPGA回路やPodに必要な詳細パラメータはコントローラが自動補完する。例えば、チャネル番号やIPアドレス、ポート番号 • 配備時の情報をもとに各種リソースの削除の設定を行う。使用済リソースが再び使用されないように使用状況を管理する • 配備・削除した結果を各種リソースの使用状況としてスケジュール機能側に提供する • ユーザからの子bsの手動書込み要求や、FPGAや子bsのリセット要求を受けて、処理を実施し、結果を各種リソースの使用状況に反映する

インフラ情報収集管理機能	• インフラ情報を自動収集してK8sリソースを自動作成する機能を提供する
	  • インフラの構成情報や書込まれている領域情報など、環境依存情報の収集とk8s上での活用の自動化
	  • DFに依存する情報(DF毎に異なる情報)の自動補完

スケジュール機能

- データフローの配備要求を受けて、ユーザが選択したスケジューリング条件や現在の使用状況などを考慮して、データフローの各構成要素の配備先を自動決定する
  - スケジューリング処理の概要
    1. ファンクションの配備先候補から、DFを構成する各ファンクションの配備先の組合せパターンを作成する
       - 回路書込み済のFPGAがあり、収容可能なDF数に余裕があれば、その回路を使用する想定で配備先を選択する
       - 未書込み状態のFPGAがあり、子bsを書込み可能であれば、新規書込み想定で配備先を選択する（配備先として書込み後の回路を選択する）
       - 既存Podが動いていて収容可能なDF数に余裕があれば、そのpodを使用する想定で配備先を選択する
       - CPU/GPUに余裕があり、Podが作成可能であれば、新規pod作成想定で配備先を選択する
    2. ユーザが選択したスケジューリング条件や現在の使用状況などを考慮して、条件を満たした配備先の組合せパターンをフィルタリング・スコアリングする
       - 途中で候補が無くなればリトライ
    3. スコアに基づき配備先の組合せパターンを1つ選択し、DFを構成する各ファンクションの配備先を決定する
- 基本配備機能にデータフローの各構成要素(WBFunction/WBConnection)の配備・削除を要求する
  - 削除の際は、順序を考慮してWBConnectionをすべて削除してからWBFunctionを削除する
- FCのテンプレートの登録や、それに伴うファンクションやコネクションの登録も行える

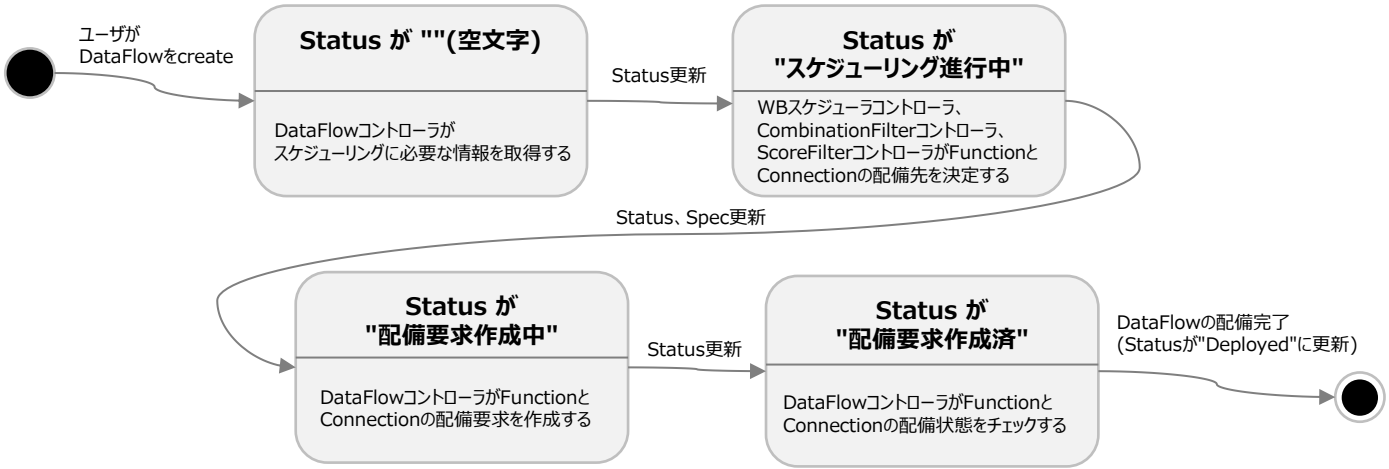
スケジュール機能:
機能構成



スケジュール機能:
配備要求機能（概要）

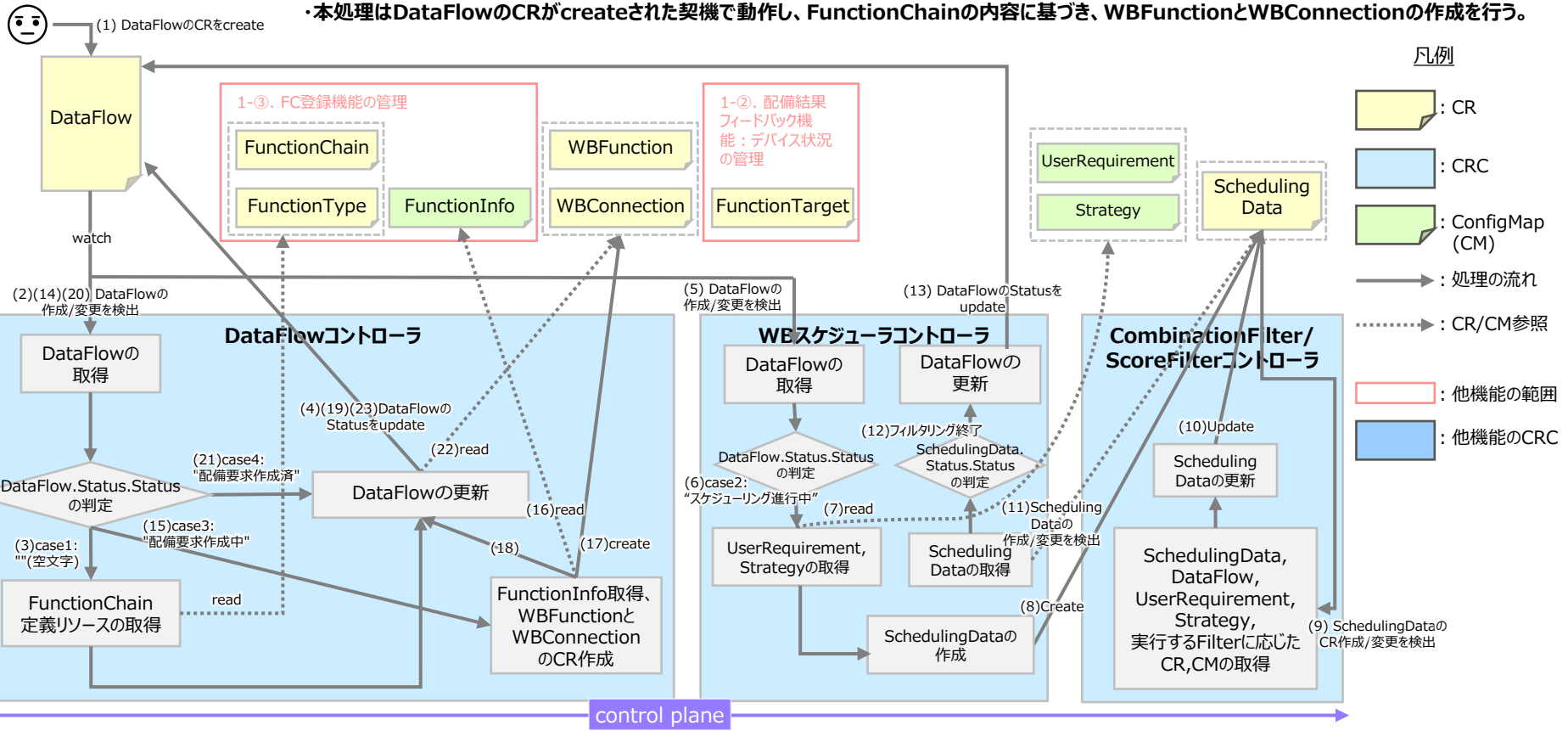
- カスタムリソースであるDataFlowの作成を契機に、配備先のスケジューリングをして「基本配備機能」に配備要求を出す
- 下図の状態遷移の通り、DataFlowの状態(*1)に応じてDataFlowコントローラ、WBスケジューラコントローラ、CombinationFilterコントローラ、ScoreFilterコントローラが連携動作する

(*1)DataFlowのStatus.Statusフィールドの文字列

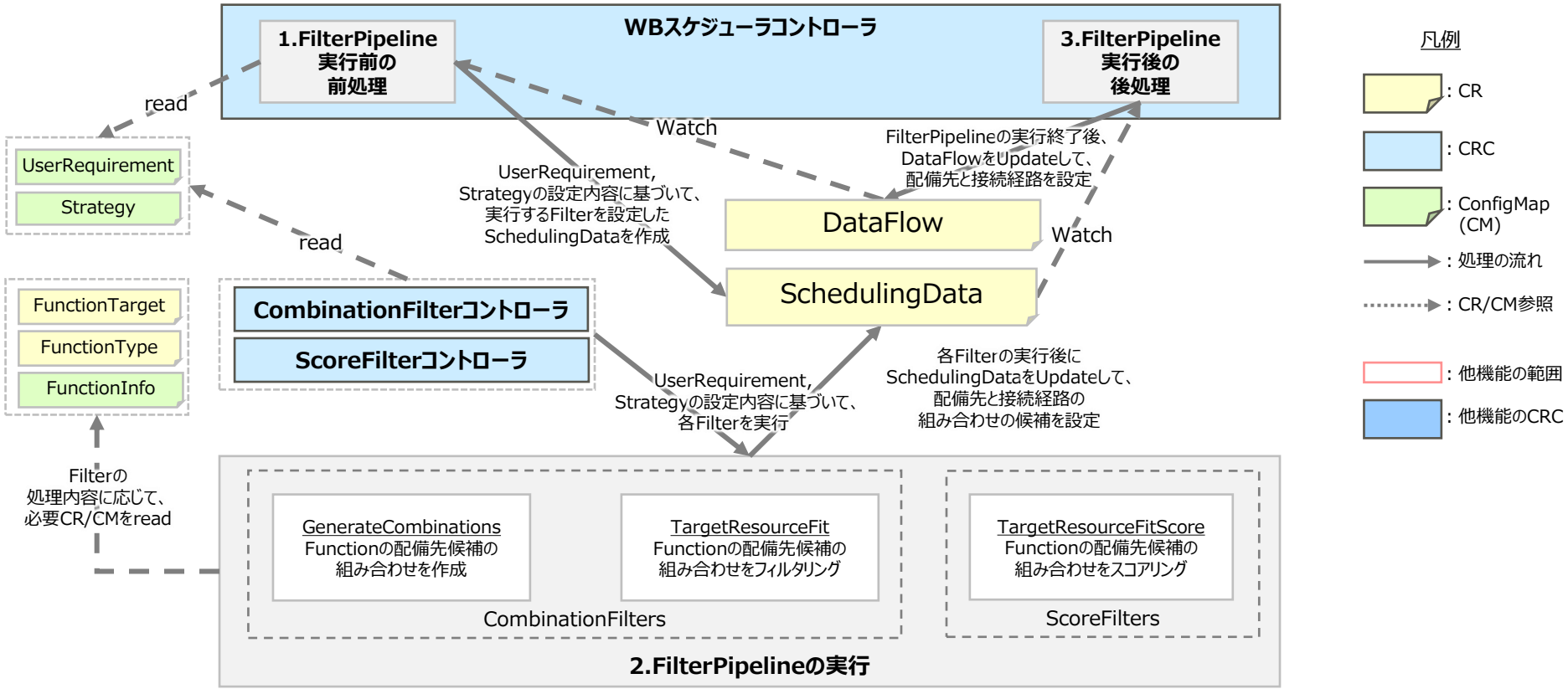


スケジュール機能:
配備要求機能（ブロック図）

・本処理はDataFlowのCRがcreateされた契機で動作し、FunctionChainの内容に基づき、WBFunctionとWBConnectionの作成を行う。



スケジュール機能:
配備要求機能（スケジューリング処理のブロック図）



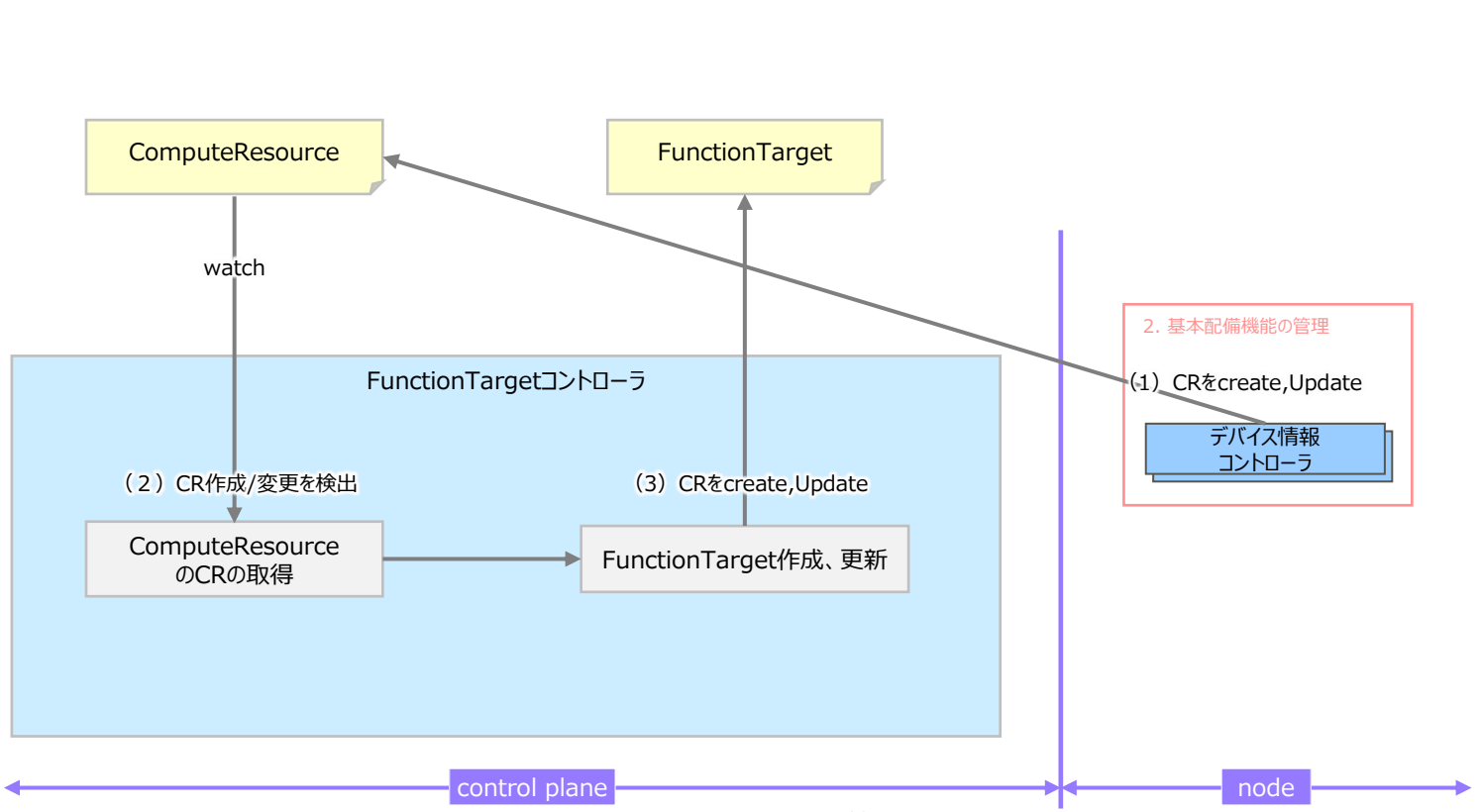
スケジュール機能:
配備結果フィードバック機能（概要）

- 「基本配備機能」によってWBFunctionが配備された結果のデバイスの利用状況を、スケジューラにフィードバックする
  - デバイス情報作成用のConfigMap(ComputeResource)の作成・更新・削除に合わせて、デバイス情報を作成・更新・削除する機能
  - 上記のデバイス情報を管理するためのカスタムコントローラとして、FunctionTargetコントローラを利用し、
    スケジューリング時に参照されるカスタムリソースであるFunctionTargetのStatusの情報としてデバイス情報を管理する

- 参考
  - FunctionTarget：Functionの配備先を表現するためのカスタムリソース
    - Functionを配備可能なデバイス上の領域（＝Lane。複数領域が無い場合はデバイス自体）を管理したもの
    - 複数チャネルや複数Podのように1領域に複数配備できる場合、それらの情報はFunctionTarget.functionsパラメータでリストとして管理される
    - 詳細はCR/CM仕様書を参考のこと

スケジュール機能:
配備結果フィードバック機能（ブロック図）

・本処理はCRのcreate, updateされた契機で動作し、ComputeResourceの内容に基づきFunctionTargetのcreatもしくはupdateを行う。



凡例

: CR

: CRC

: ConfigMap (CM)

: 処理の流れ

: CR/CM参照

: 他機能の範囲

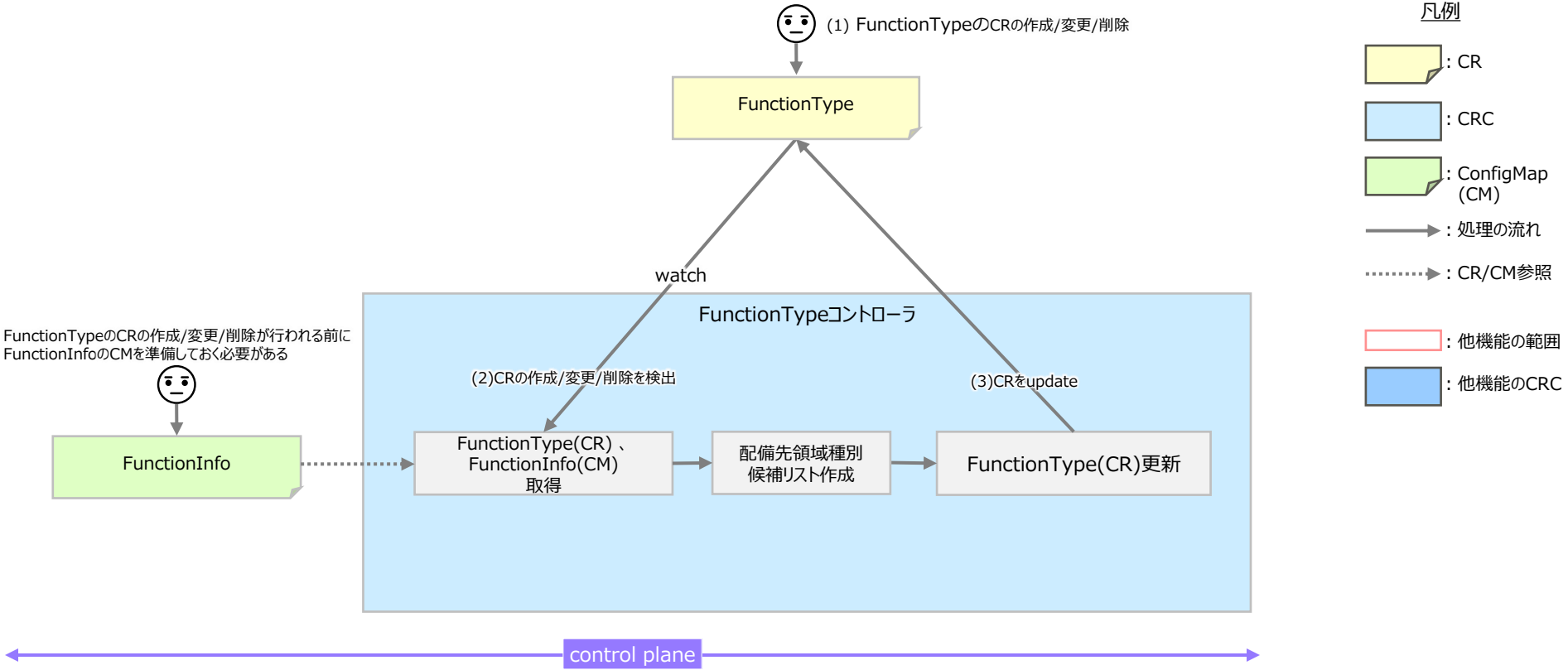
: 他機能のCRC

スケジュール機能:
FC登録機能

- FunctionChainのテンプレートの登録・管理に必要な処理を行う
  - ファンクションのカタログ（FunctionType CR）の登録・変更
    - FunctionInfoおよび自身のデータを取得し、deployableItmesのvalueをチェックしてFunctionTypeの配備先候補を編集する
  - ファンクションチェインの定義（FunctionChain CR）の登録・変更
    - FunctionInfo/FunctionTypeおよび自身のデータを取得し、FunctionChainに定義されたFunctionの登録（FunctionTypeがReady）をチェックする。
      Functionの接続数オーバー/接続Port重複エラーが無いかもチェックする

スケジュール機能:
FC登録機能（ブロック図）

・本処理はFunctionTypeのCRのcreate, update,deleteされた契機で動作し、FunctionInfoのConfigmapの内容に基づきFunctionTypeのCRのupdateを行う。



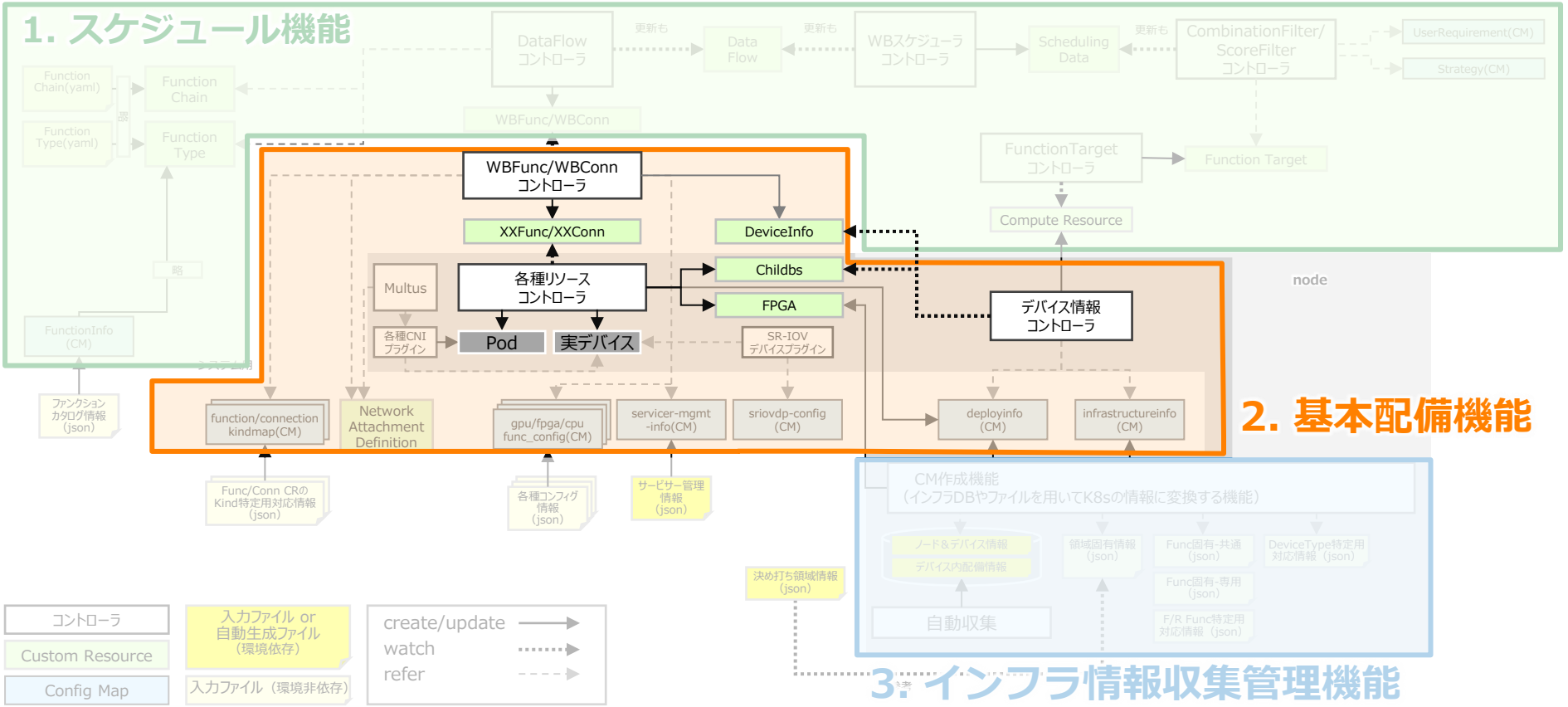
基本配備機能

- データフローの各構成要素の配備・削除要求を受けて、各種リソースコントローラを用いて実際に配備・削除の設定を行う
- 配備先のFPGAに子bsが書込まれていなければ、動的に子bsを書込む
- 各種FPGA回路やPodに必要な詳細パラメータはコントローラが自動補完する
  - 例えば、チャネル番号やIPアドレス、ポート番号
- 配備時の情報をもとに各種リソースの削除の設定を行う。使用済リソースが再び使用されないように使用状況を管理する
- 配備・削除した結果を各種リソースの使用状況としてスケジュール機能側に提供する
- ユーザからの子bsの手動書込み要求や、FPGAや子bsのリセット要求を受けて、処理を実施し、結果を各種リソースの使用状況に反映する

基本配備機能:
インフラやアプリの情報管理の基本方針

- 共通項目をコンフィグ情報で定義しておき各種ファンクション作成時にその情報を利用する
  - 「共通項目」とは、そのコンフィグ情報のファンクションとCR Kindであれば、どのDFでも同じ値となる項目
    - 例：ファンクションのファイル(コンテナイメージ名やbsファイル名など)やCPUFunction/GPUFunctionにおけるpodの"Template"配下の定型部など
- 個別項目に必要名情報はシステム側で管理しておき基本配備機能部で自動補完する
  - 「個別項目」とは、DF毎に値が異なる項目。またはインフラ構成に依存する項目
    - 例：配備先の環境(インフラ構成や採用する共有方式の違いなど)によって異なる値になる項目、環境に依存しないがDF毎に異なる値になる項目など
  - スケジュール機能部から渡されたDFの情報や、インフラ情報収集管理機能が収集したインフラの情報をもとに自動補完

基本配備機能:
機能構成

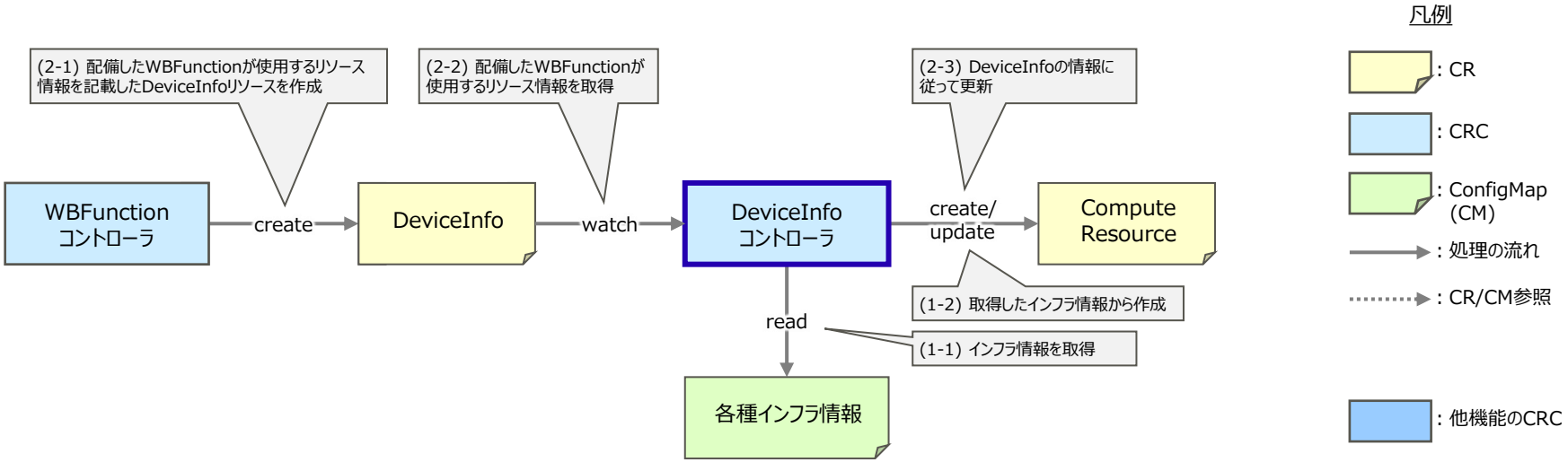

1. スケジュール機能
DataFlow コントローラ
更新も
Data Flow
更新も
WBスケジューラ コントローラ
Scheduling Data
更新も
CombinationFilter/ ScoreFilter コントローラ
UserRequirement(CM)
Strategy(CM)
Function Chain(yaml)
Function Chain
Function Type(yaml)
Function Type
WBFunc/WBConn
FunctionTarget コントローラ
Function Target
Compute Resource
WBFunc/WBConn コントローラ
XXFunc/XXConn
DeviceInfo
Childbs
FPGA
各種リソース コントローラ
Multus
各種CNI プラグイン
Pod
実デバイス
SR-IOV デバイスプラグイン
デバイス情報 コントローラ
node
FunctionInfo (CM)
ファンクション カタログ情報 (json)
function/connection kindmap(CM)
Network Attachment Definition
gpu/fpga/cpu func_config(CM)
servicer-mgmt -info(CM)
sriovdp-config (CM)
deployinfo (CM)
infrastructureinfo (CM)
2. 基本配備機能
Func/Conn CRの Kind特定用対応情報 (json)
各種コンフィグ情報 (json)
サービサー管理情報 (json)
CM作成機能
(インフラDBやファイルを用いてK8sの情報に変換する機能)
ノード&デバイス情報
デバイス内配備情報
領域固有情報 (json)
Func固有-共通 (json)
DeviceType特定用 対応情報 (json)
Func固有-専用 (json)
F/R Func特定用 対応情報 (json)
自動収集
決め打ち領域情報 (json)
3. インフラ情報収集管理機能
コントローラ
Custom Resource
Config Map
入力ファイル or 自動生成ファイル (環境依存)
入力ファイル (環境非依存)
create/update
watch
refer

基本配備機能:
機能構成（説明対象）

項目	概要
デバイス情報コントローラ (DeviceInfoコントローラ)	初期状態のComputeResourceリソース(デバイス情報)を作成する 配備状況に応じてComputeResourceを更新する
WBFunctionコントローラ	WBFunctionリソースから各Function系(CPUFunction/GPUFunction/FPGAFunction)のCRを作成する WBFunctionの配備に伴い、デバイス情報コントローラに配備に必要なデバイスに関する情報の払い出しと、配備状況の更新を要求する
CPUFunctionコントローラ	CPUを動作させるPodの配備・管理を行う
GPUFunctionコントローラ	GPUを動作させるPodの配備・管理を行う
FPGAFunctionコントローラ	FPGAアクセラレータ装置で実行する処理の配備・管理を行う FPGAや子bsの情報や状態を管理し、必要に応じてFPGA内リソースの割当やFPGAへの子bsの書込みを行う 配備時の情報をもとに各種リソースの削除の設定を行う。使用済リソースが再び使用されないように使用状況を管理する ユーザからの、子bsの手動書込み要求や、FPGAや子bsのリセット要求を受けて動作することもある
WBConnectionコントローラ	WBConnectionリソースから各Connection系(EthernetConnection/PCIeConnection)のCRを作成する
EthernetConnectionコントローラ	CPU, GPU上のpodおよび外部装置の間のEthernet接続を実現する
PCIConnectionコントローラ	アクセラレータ(CPU, GPU, FPGA)装置間のPCIe接続を実現する

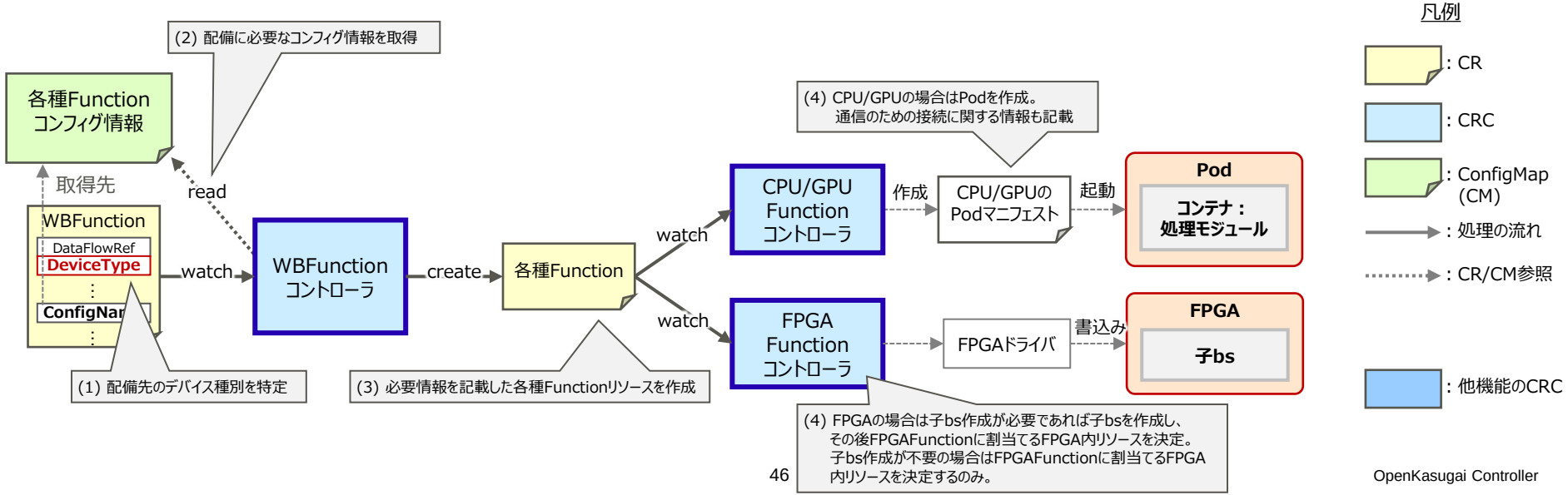
基本配備機能：
DeviceInfoコントローラ

- インフラリソース管理を実施するため、インフラ管理用の「DeviceInfoコントローラ(デバイス情報コントローラ)」を導入する
  - システム構築時に、各種デバイスやネットワークの情報を取得し、スケジュール機能部で使用するComputeResourceを作成
    - インフラ情報収集管理部が作成した各種インフラ情報をもとに、配備先の「領域」や各領域に配備済みの回路やPodを識別し、スペックや使用状況を管理する
  - DF配備時に、WBFunctionコントローラからの情報をもとに、ComputeRsourceを更新



基本配備機能：
WBFunctionコントローラ、各種Functionコントローラ

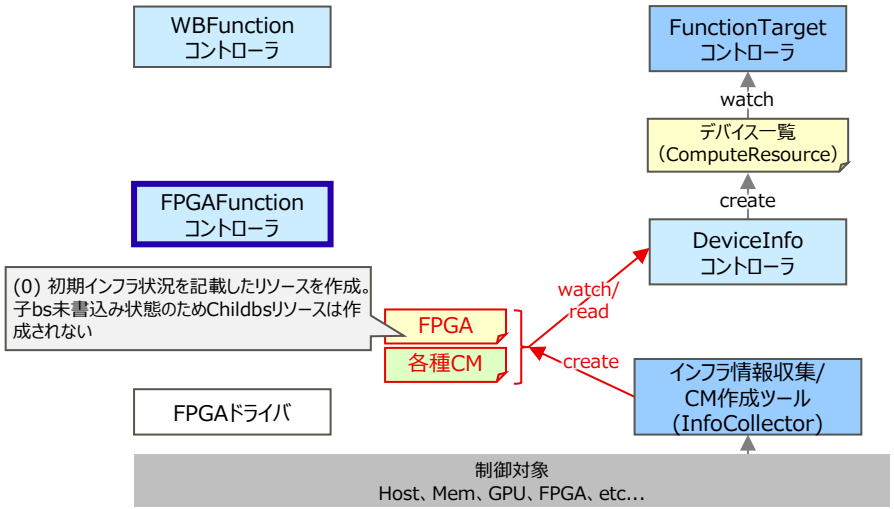
- WBFunctionの配備に必要な情報(コンフィグ情報、デバイス情報)を取得し、配備先のデバイスに対応した各Function系(CPUFunction/GPUFunction/FPGAFunction)のCRを作成する
- 各種FunctionのCRを受けて、対応するFunctionコントローラが動作する。各種処理モジュールを配備する
  - CPU/GPUの場合はPodを作成
  - FPGAの場合はFPGAドライバを介して子bs書込みや回路設定を実施（詳細は次ページ）



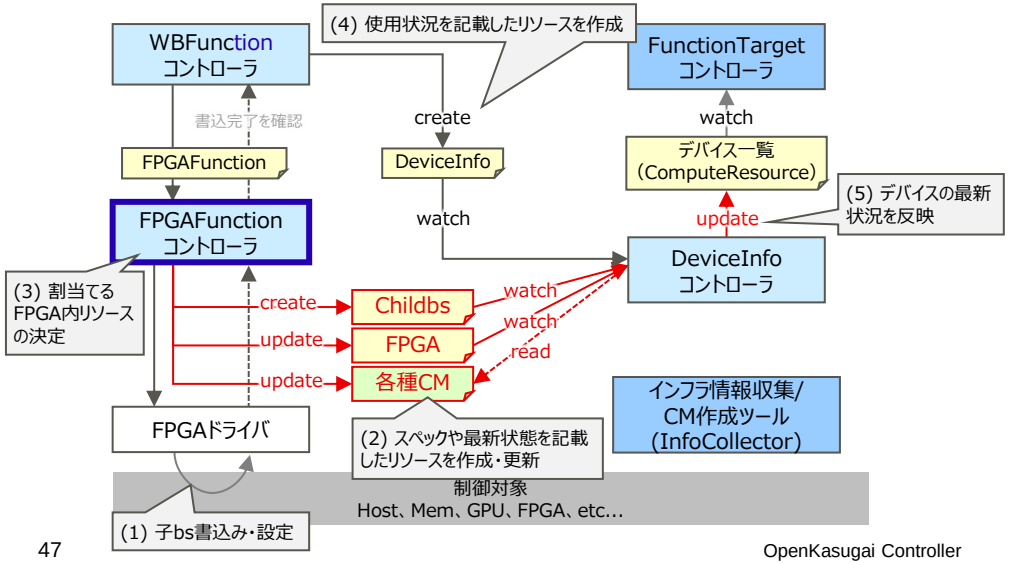
基本配備機能：
FPGAFunctionコントローラ①

- FPGAFunctionのCRを受けて、必要に応じて子bs書込みを行った上で、各種処理モジュールを配備する
  - 配備先が書込み済領域の場合、払い出されたデバイス情報に従って、FPGAドライバを介して各種設定を実施
  - 配備先が未書込み領域の場合、払い出されたデバイス情報に従って、FPGAドライバを介して子bsを書込み、その後、各種設定を実施
- 配備に伴うFPGA(FPGAリソース)や子bs(Childbs)のスペックや現在の状態(書込み中等)を管理・作成・更新する

参考：システム構築時 ※詳細は後述

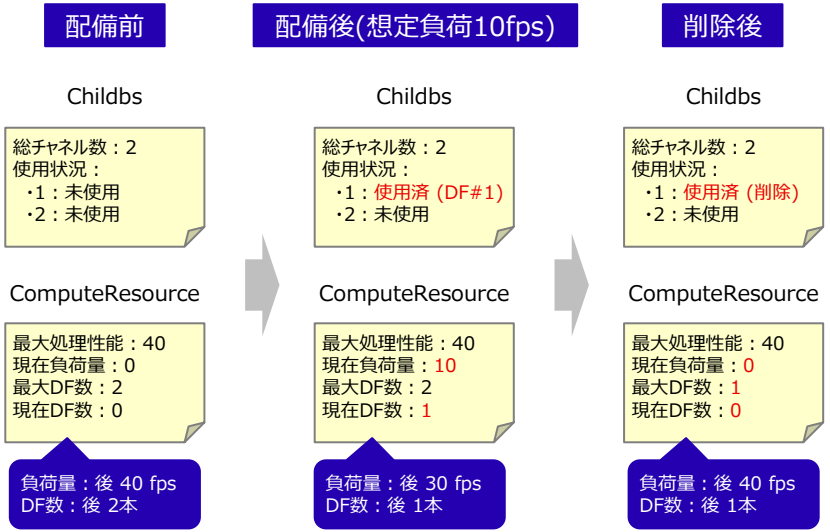


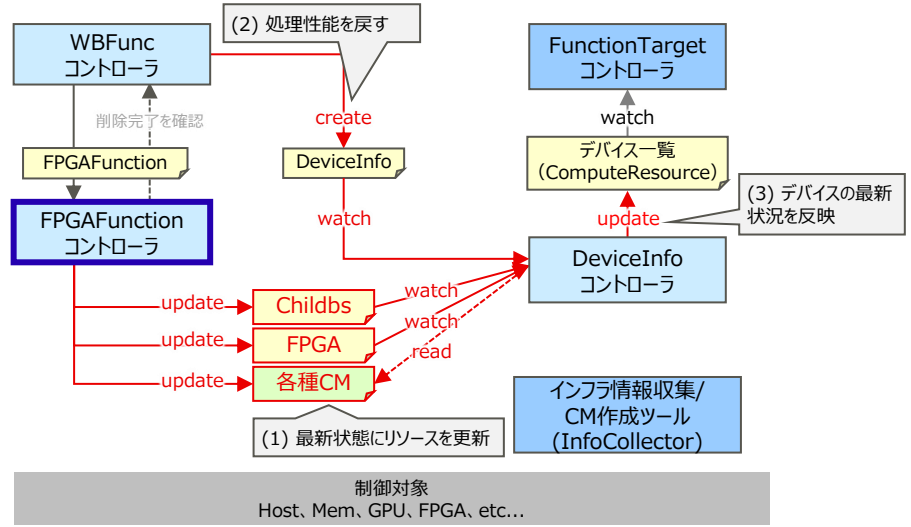
子bs書込みを伴うFPGAFunction配備時



基本配備機能：
FPGAFunctionコントローラ②

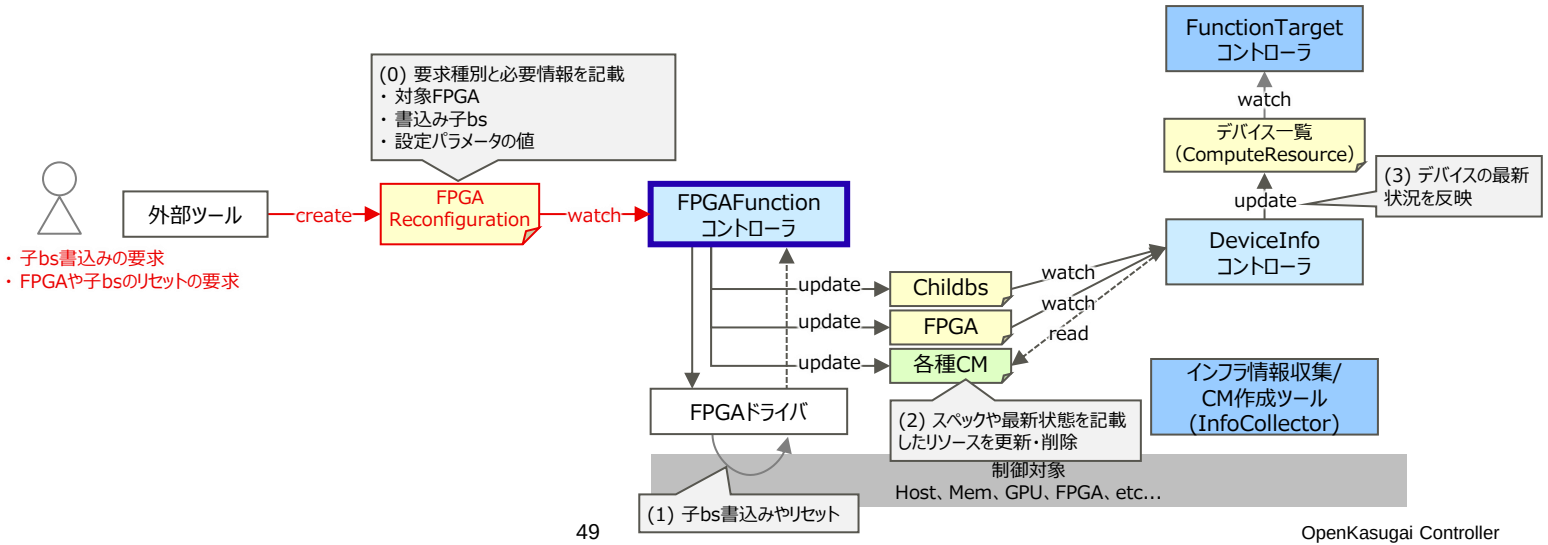
- DF削除に伴い回復するリソースと回復しないリソースを意識して、各種リソースの情報を更新する
  - 処理可能な負荷量：DF削除により処理可能な負荷量は回復するため、現在負荷量を配備前に戻す（想定負荷量分だけ現在負荷量を減らす）
  - 配備可能なDF数：使用済チャネルは再使用できないため、DF削除により配備可能なDF数は回復しない。そのため、現在DF数を配備前に戻す際、併せて最大DF数を減らすことで、削除前後で配備可能なDF数が変わらないようにしている
- FPGAの停止・設定の削除自体はPCIeConnectionコントローラの接続処理の停止の一環として実施する





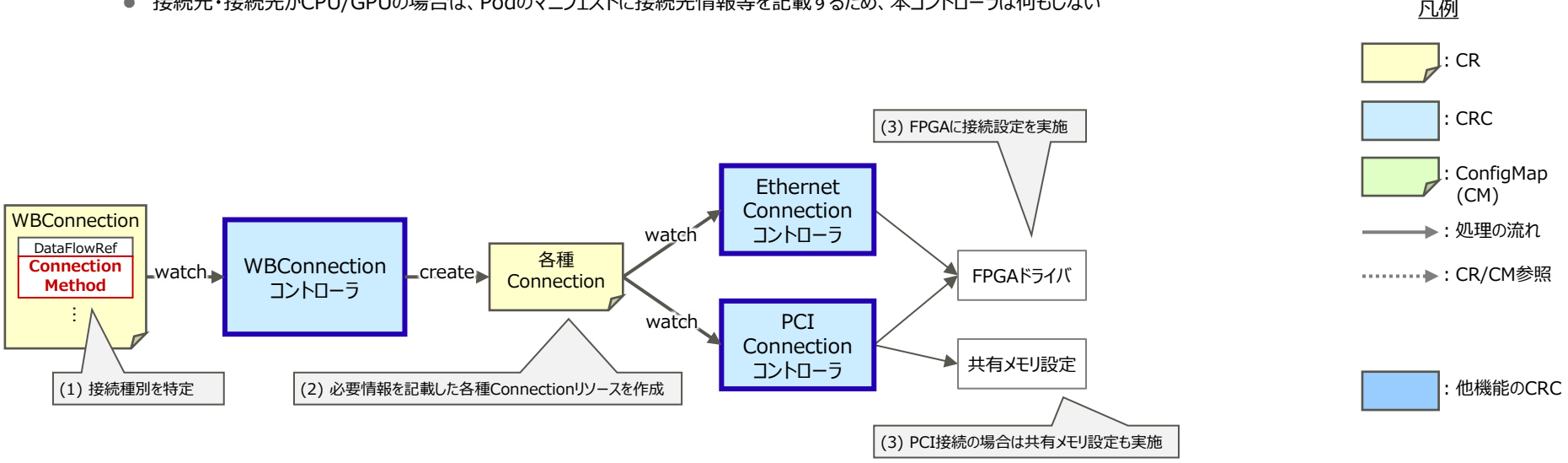
基本配備機能：
FPGAFunctionコントローラ②

- 専用のCR(FPGAReconfiguration)を介してFPGAコントローラに子bsの手動書込みや各種リセットを直接要求できる
  - 子bsの手動書込みの場合、設定するパラメータを細かく指定して回路設定できる
  - 前述した通り、各種リセットを要求された際、本コントローラ実装では子bsの上書き処理を行っている
- 手動書込みやリセットを実施した場合、FPGAリソースやChildbsのスペックや現在の状態を更新する
  - FPGAのリセットにより子bs未書込状態に戻った場合、元々書込まれていた子bsに対応する既存リソース(Childbs)は削除する



基本配備機能：WBConnectionコントローラ、EthernetConnection/PCIConnectionコントローラ

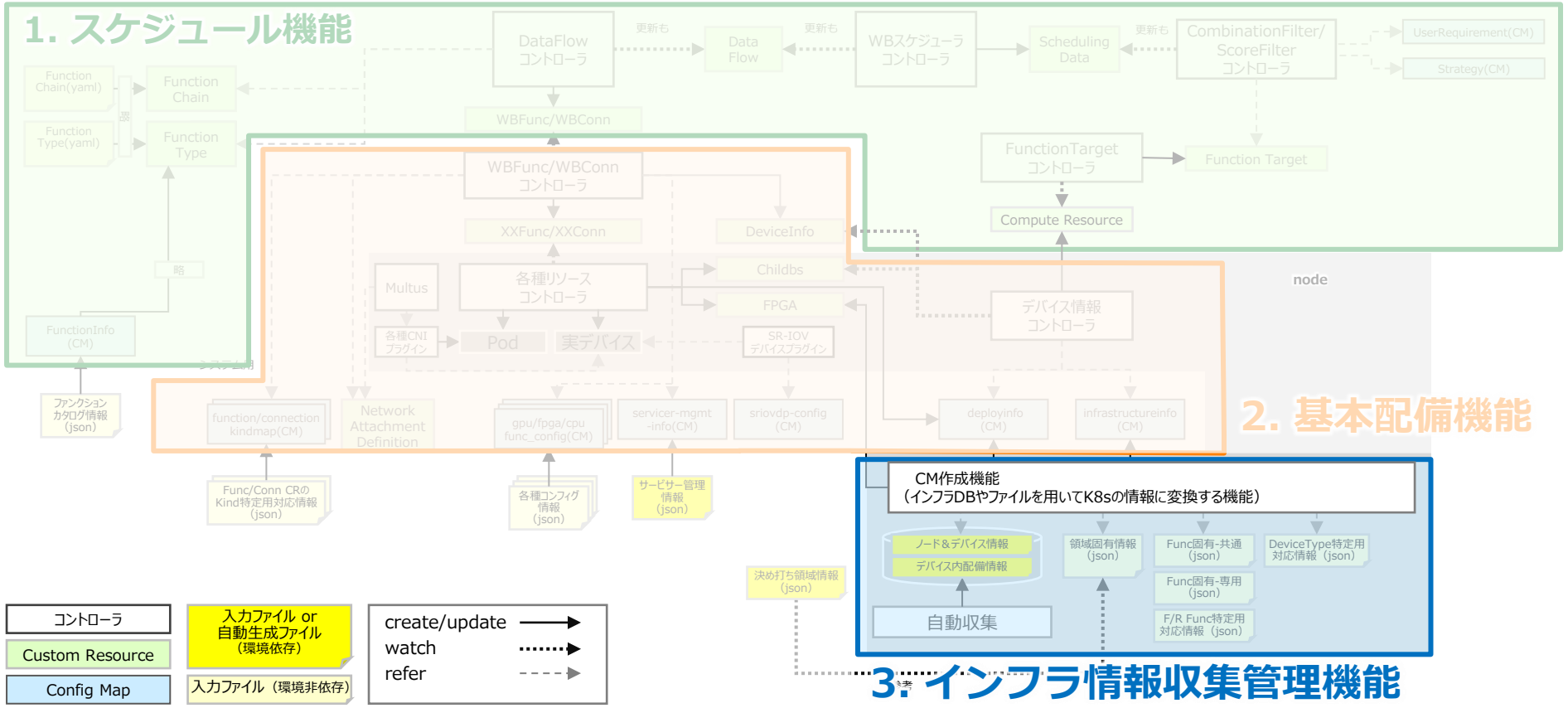
- WBConnectionの配備に必要な情報(コンフィグ情報、デバイス情報)を取得し、接続種別に対応した各Connection系(EthernetConnection/PCIConnection)のCRを作成する
- 各種ConnectionのCRを受けて、対応するConnectionコントローラが動作する
  - Ethernetの場合は、接続元・接続先がFPGAならばFPGAドライバを介して接続設定を実施
  - PCIeの場合は、DMA用の共有メモリの設定を実施し、接続元・接続先がFPGAならば同様に接続設定を実施
    - 接続元・接続先がCPU/GPUの場合は、Podのマニフェストに接続先情報等を記載するため、本コントローラは何もしない



インフラ情報収集管理機能

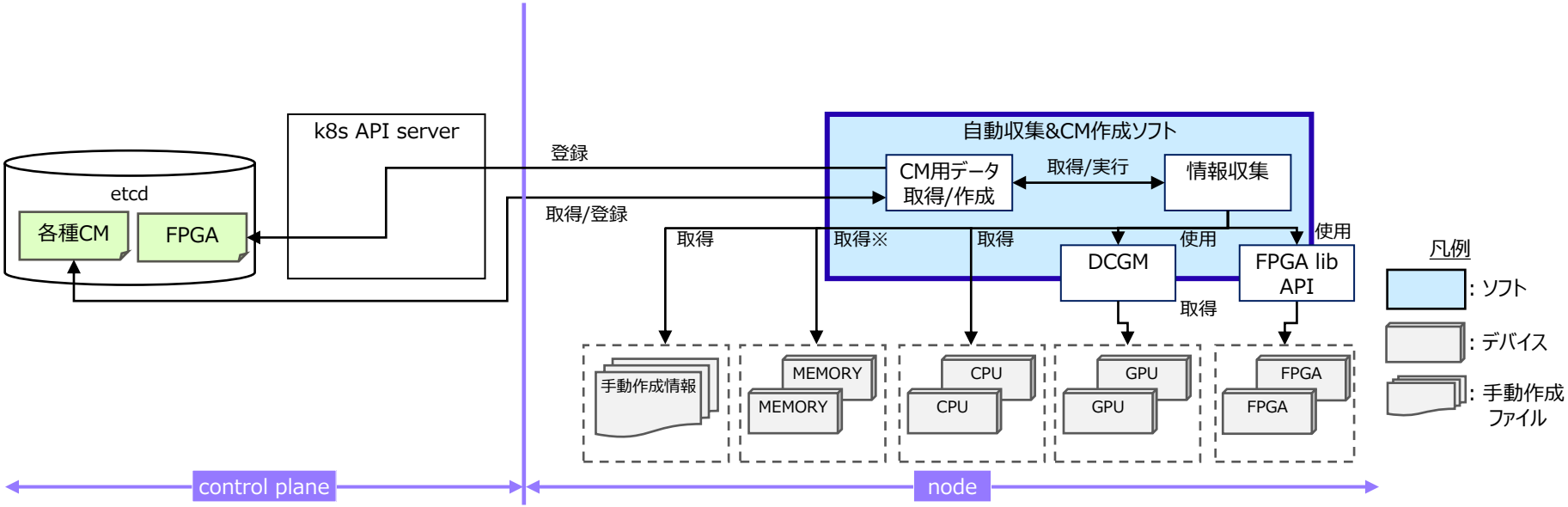
- インフラ情報を自動収集してK8sリソースを自動作成する機能を提供する
  - インフラの構成情報や書込まれている領域情報など、環境依存情報の収集とk8s上での活用を自動化
    - 環境構築の作業量を削減できる
    - 別環境への横展開を容易化する
  - DFに依存する情報(DF毎に異なる情報)を自動補完
    - ユーザ(DF配備要求者)のDF配備に向けた準備の作業量を削減できる

インフラ情報収集管理機能:
機能構成



インフラ情報収集管理機能:
自動収集&CM作成ソフト

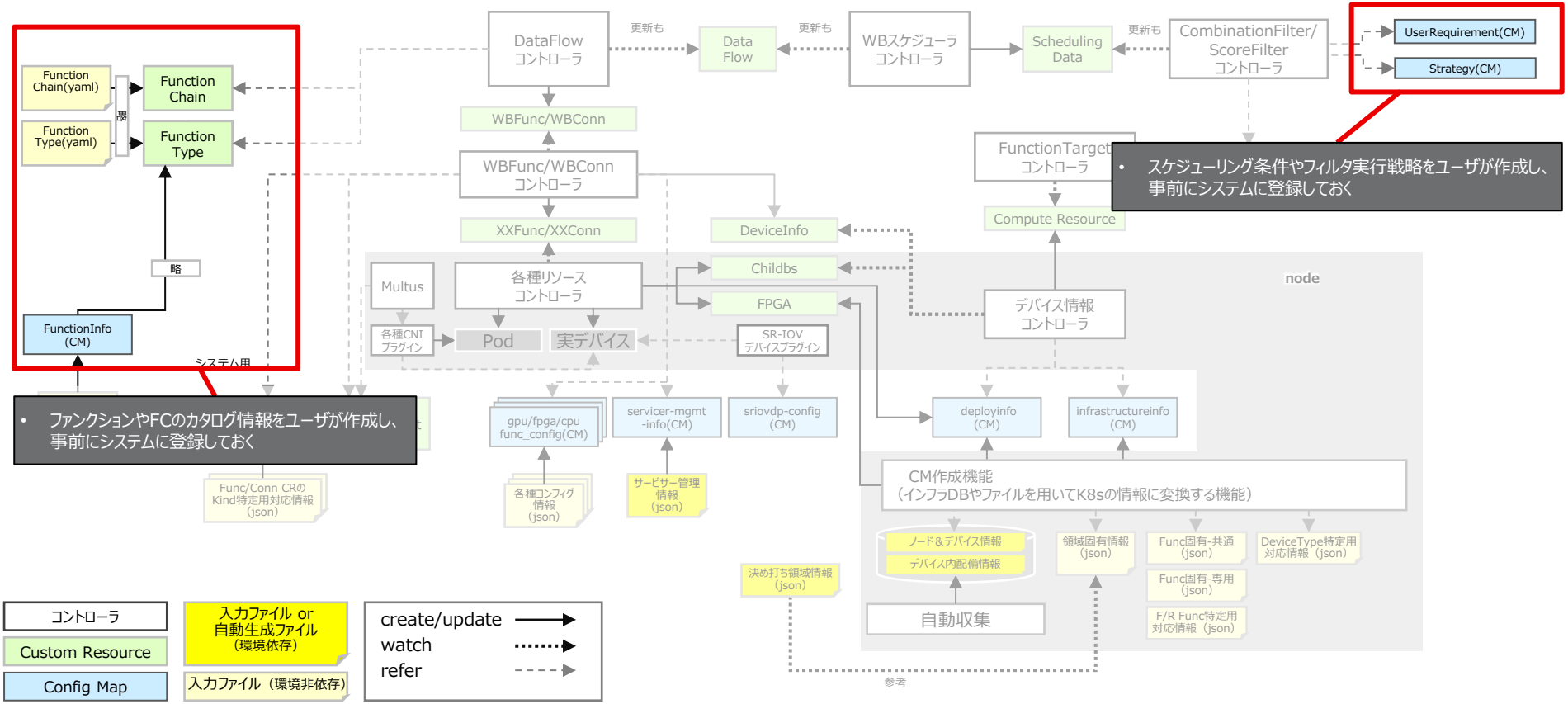
- FPGA/GPU/CPU/メモリのデバイス情報を収集する
- 収集したデバイス情報を元にConfigMap/FPGA CRの作成・登録を行う



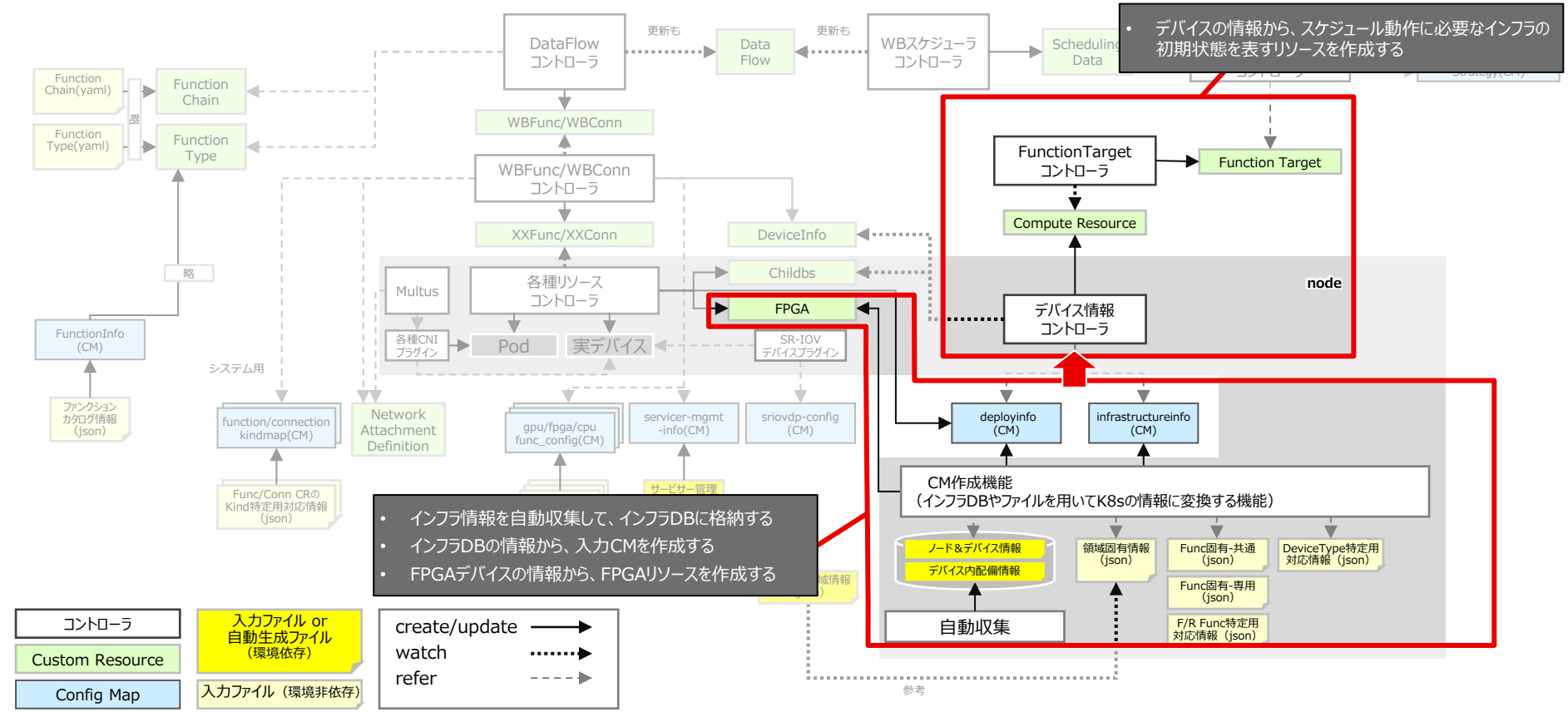
5. 全体での動作の流れ

5.1 DF配備（FPGA子bs書込み有の場合）

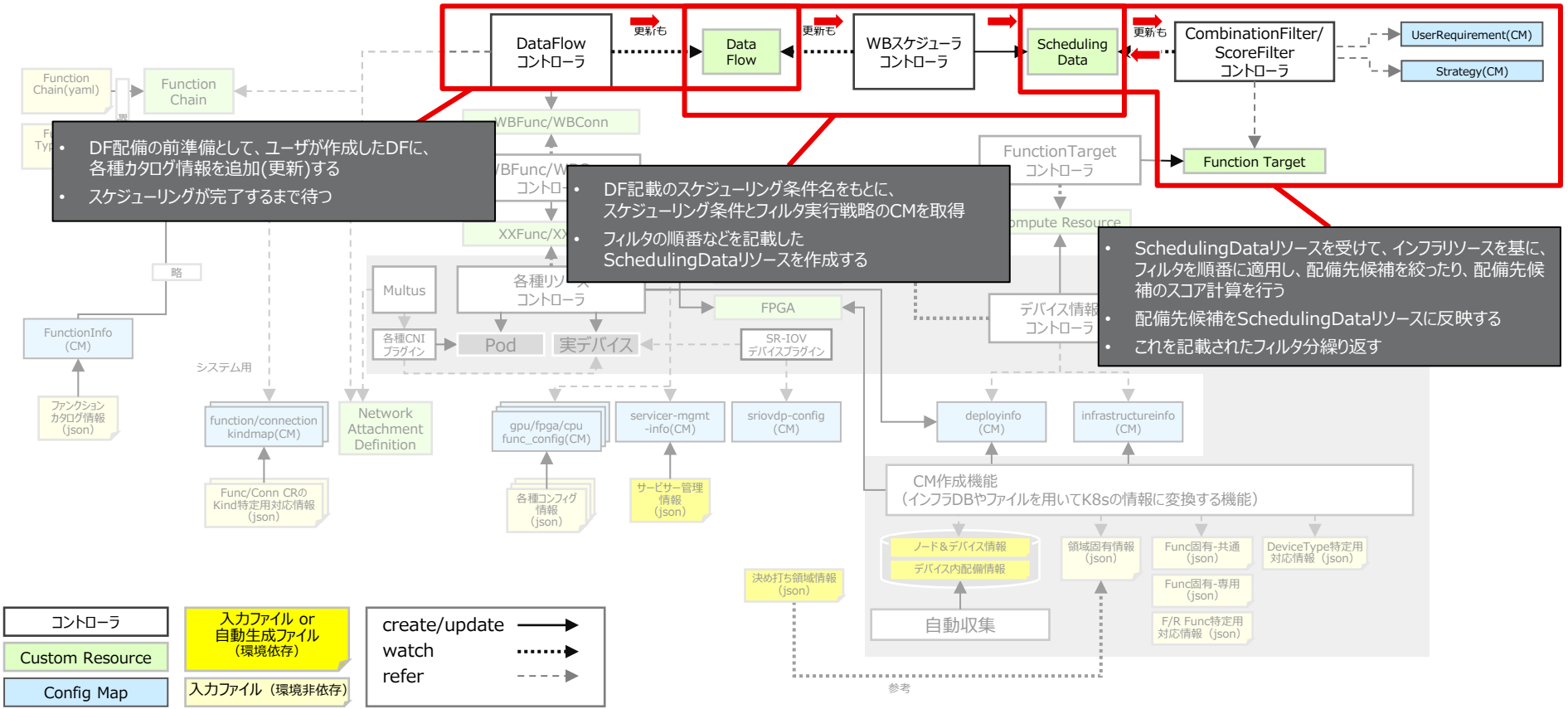
全体での動作の流れ：
① カタログ情報やスケジューリング条件の登録



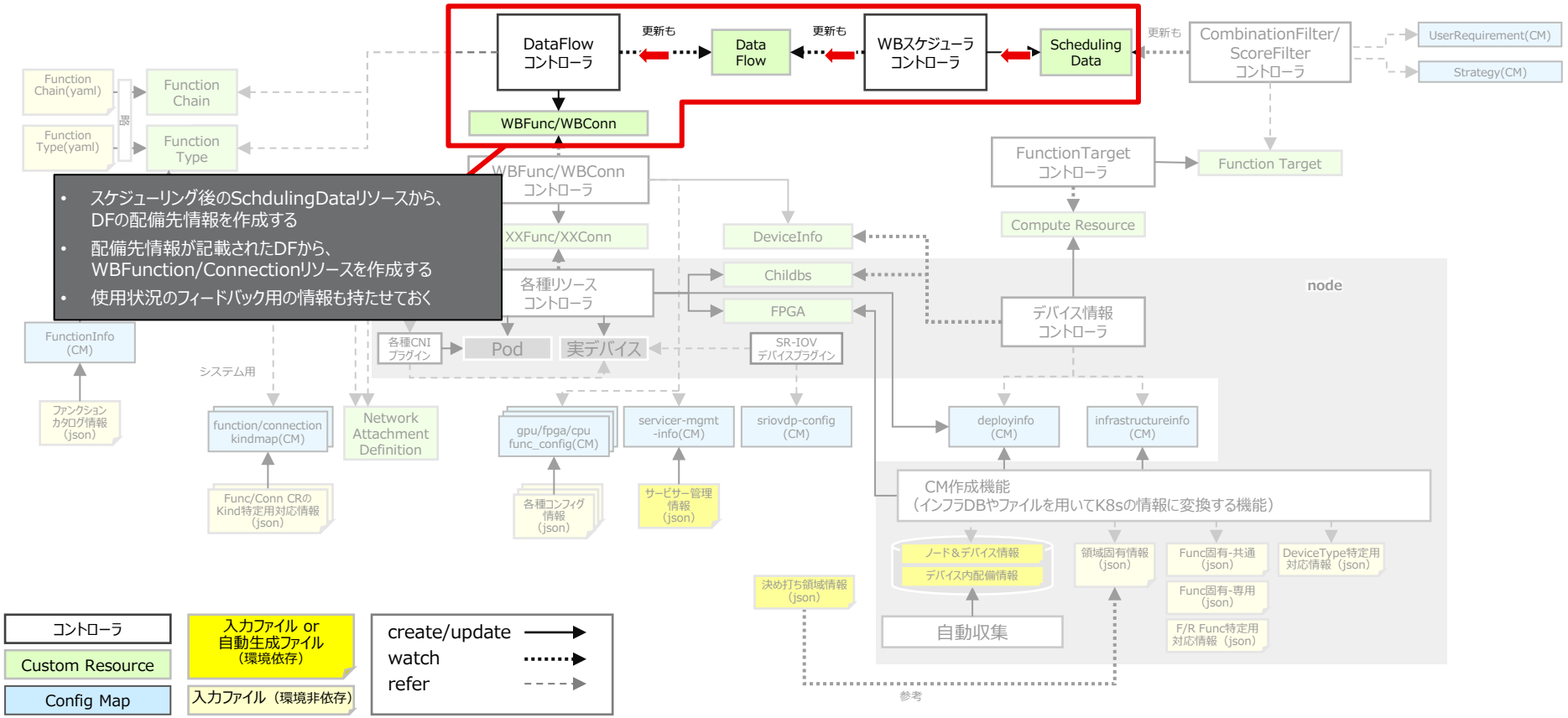
全体での動作の流れ：
② インフラ情報の自動収集、③ 初期状態のインフラリソースの作成



全体での動作の流れ：
④ DF配備の前準備、⑤ スケジューリング準備、⑥ スケジューリング

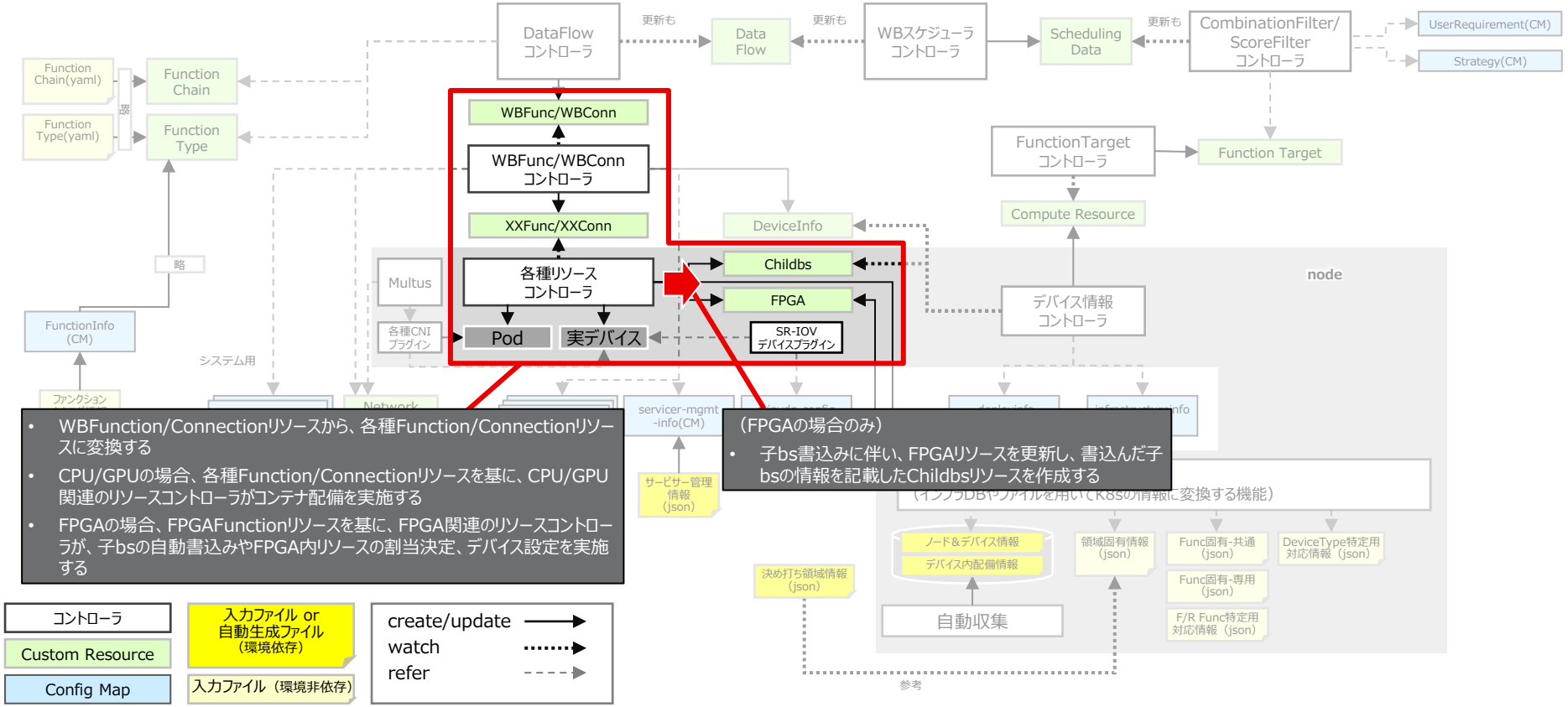


全体での動作の流れ：
⑦ WBFunction/Connectionリソースの作成

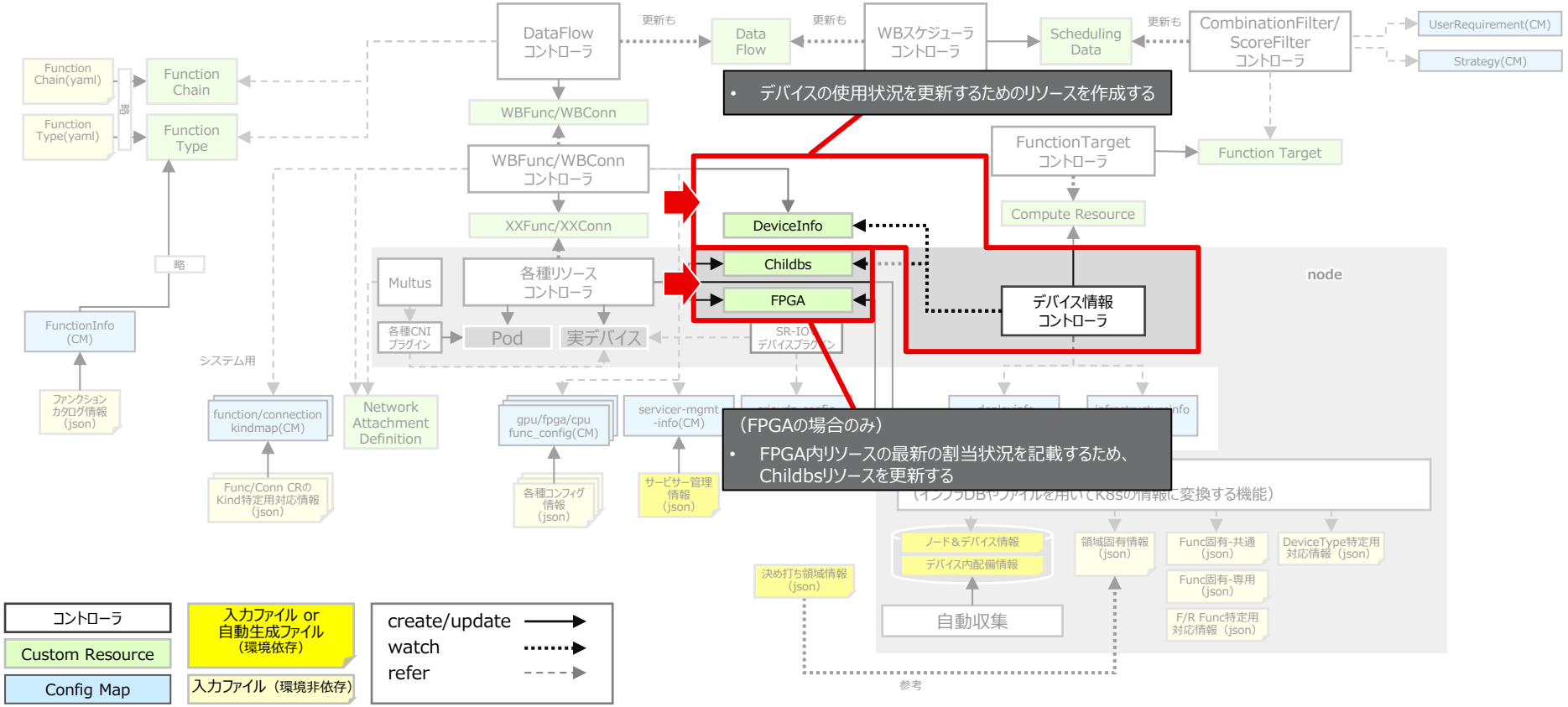


- スケジューリング後のSchdulingDataリソースから、DFの配備先情報を作成する
- 配備先情報が記載されたDFから、WBFunction/Connectionリソースを作成する
- 使用状況のフィードバック用の情報も持たせておく

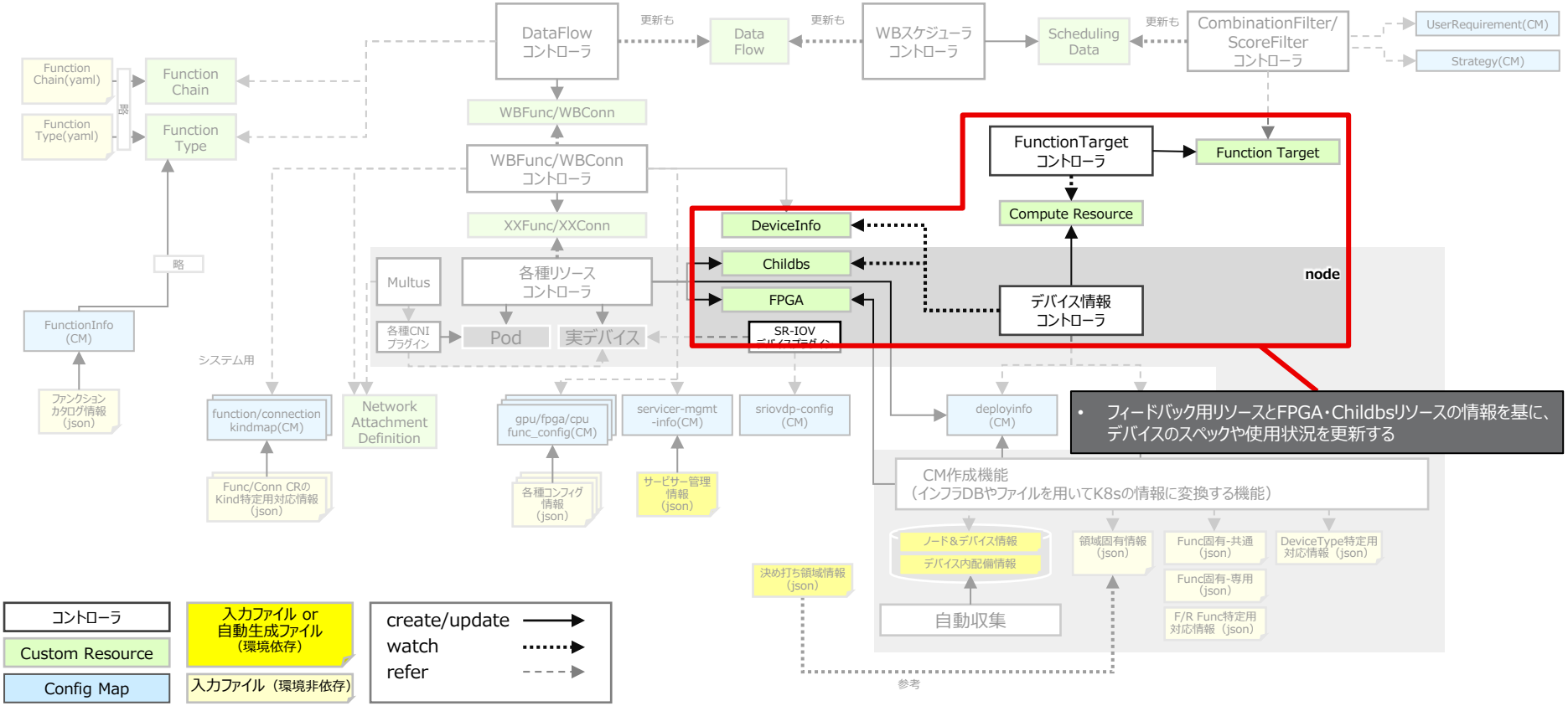
全体での動作の流れ：
⑧ 各種ファンクションの配備・設定



全体での動作の流れ：
⑨ 使用状況のフィードバック用のリソース作成・更新

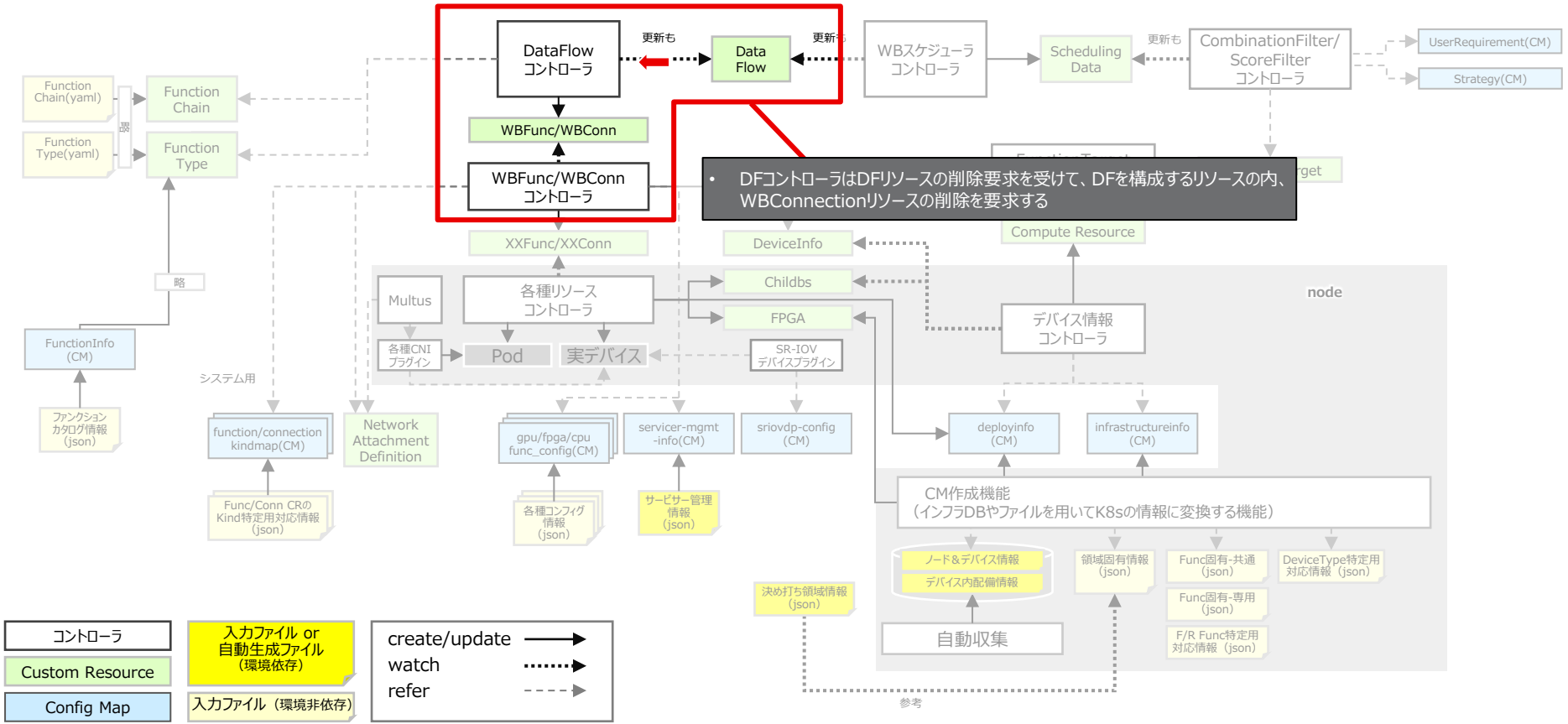


全体での動作の流れ：
⑩ インフラリソースへの使用状況のフィードバック

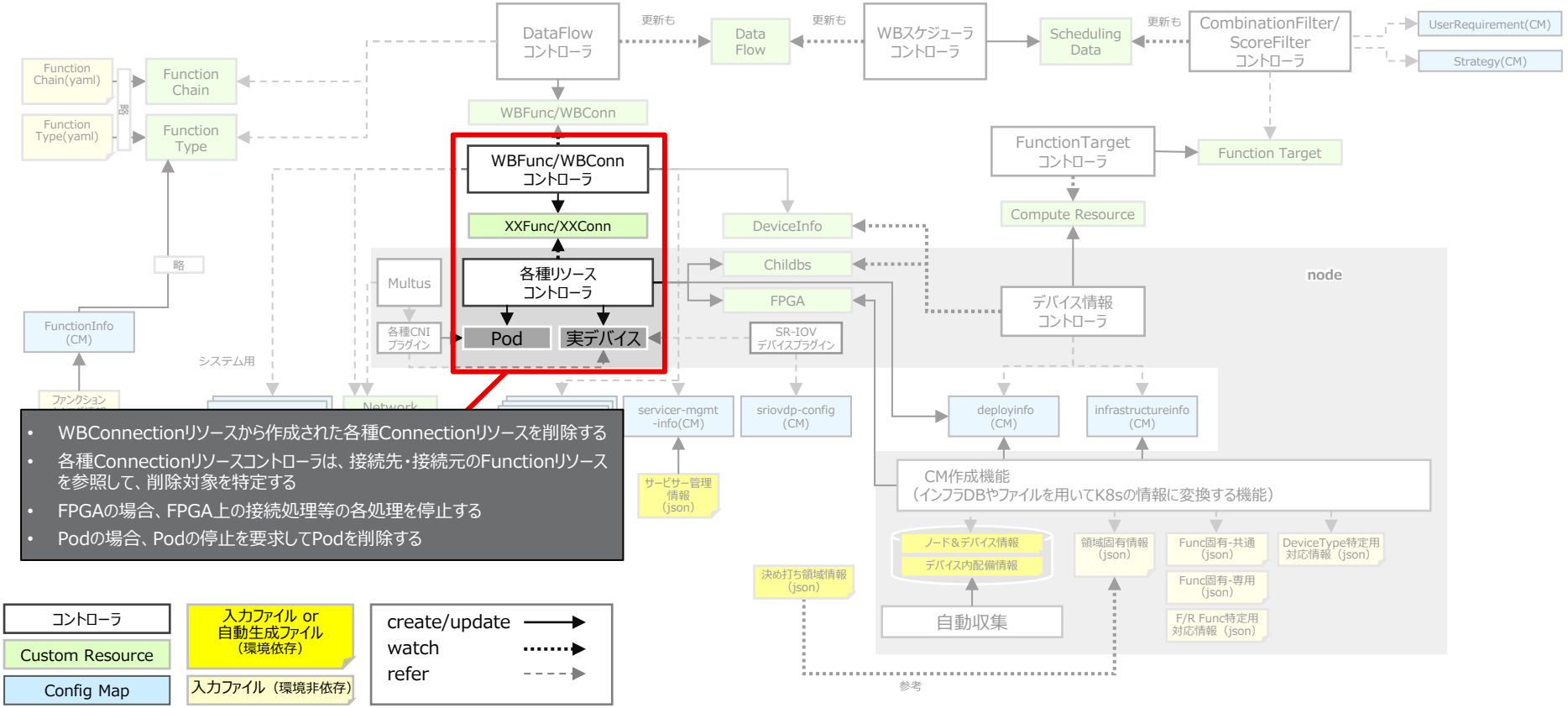


5.2 DF削除

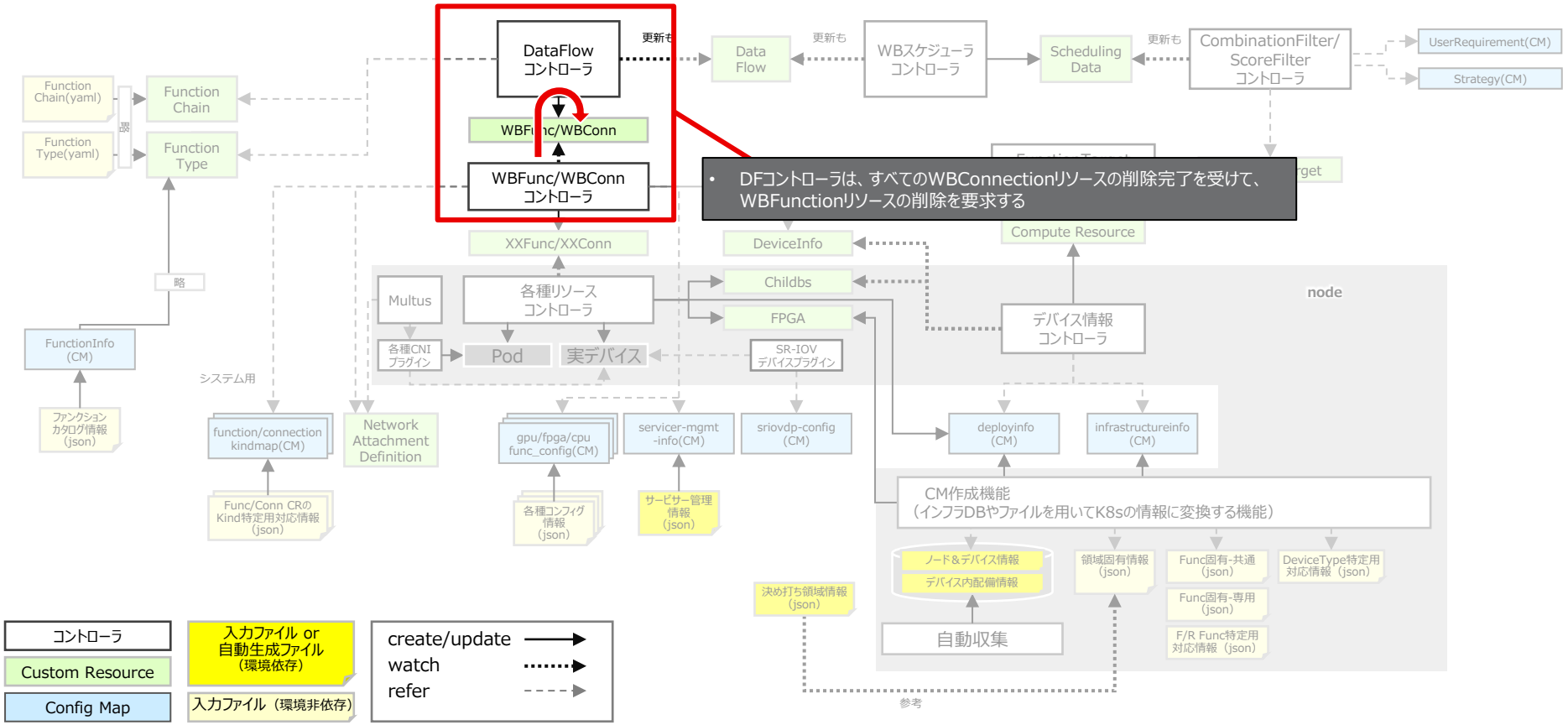
全体での動作の流れ：
① WBConnectionリソースの削除要求



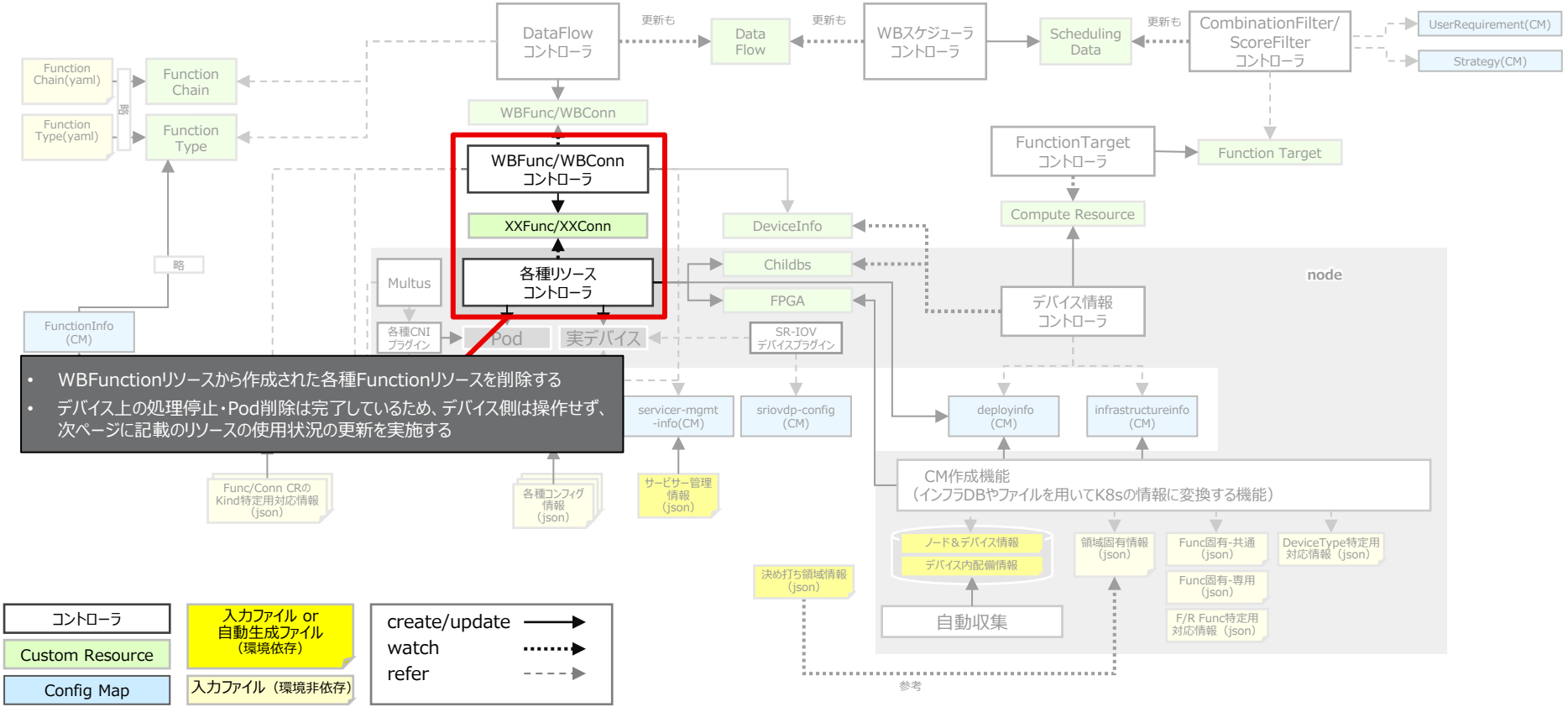
全体での動作の流れ：
② 各種コネクションリソースの削除 ＆ FPGAの処理停止/Pod削除



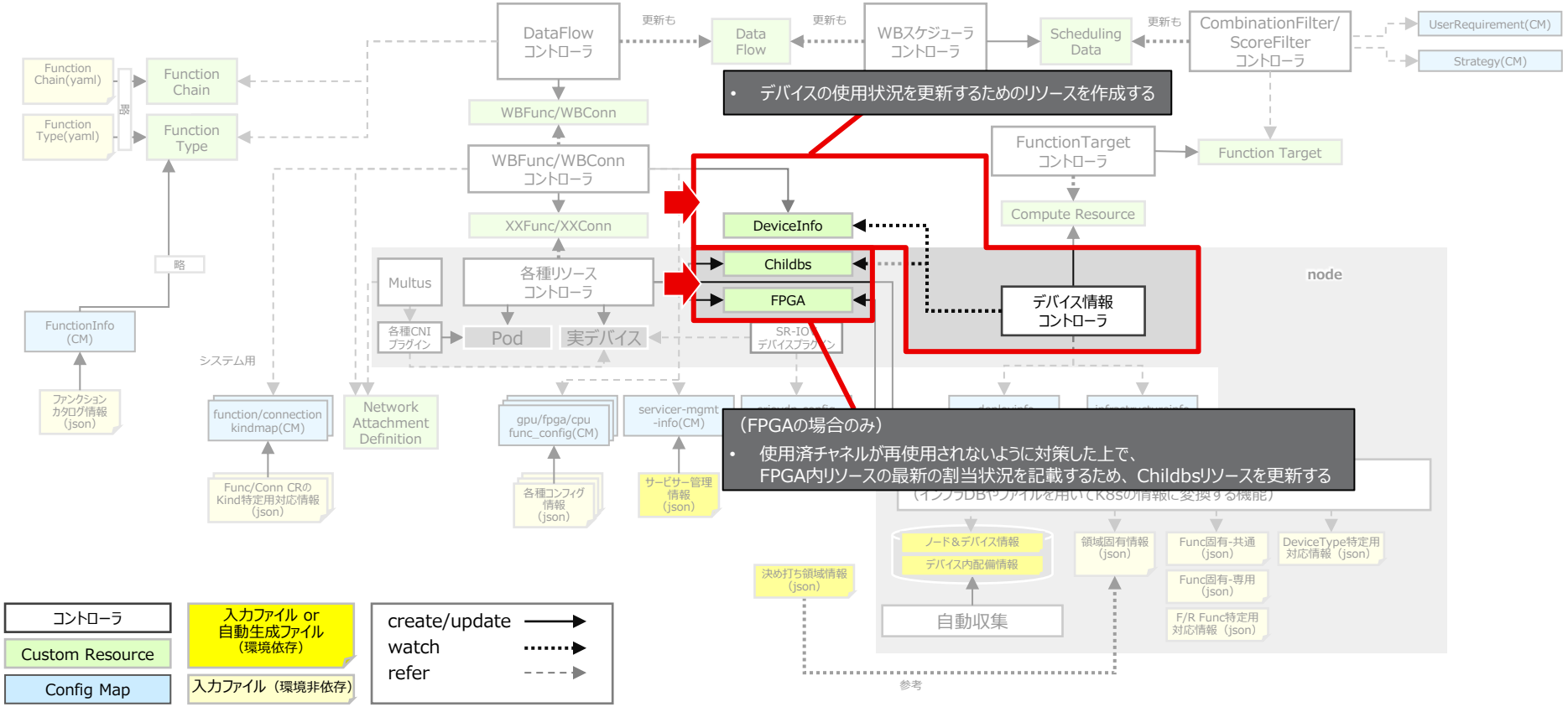
全体での動作の流れ：
③ WBFunctionリソースの削除要求



全体での動作の流れ：
④ 各種ファンクションリソースの削除



全体での動作の流れ：
⑤ 使用状況のフィードバック用のリソース作成・更新



全体での動作の流れ：
⑥ インフラリソースへの使用状況のフィードバック