技術概要

Agilent OpenLab

Agilent 機器コントロールフレームワーク（ICF）

他社製 CDS によるアジレント製機器のコントロールを簡易化するソフトウェア

はじめに

この技術概要では、アジレント製の液体クロマトグラフィー（LC）機器やガスクロマトグラフィー（GC）機器に対する他社製クロマトグラフィーデータシステム（CDS）やその他のソフトウェアソリューションからのコントールを容易に実現する Agilent 機器コントロールフレームワーク（ICF：Instrument Control Framework）ソフトウェアについて解説します。また、ICF と RapidControl.NET（RC.NET）機器ドライバの基盤をなす技術の概念や、オープンシステム環境におけるソフトウェアコンポーネントの連携についても説明します。

競争の激しい現在の環境において、多くのラボがコストの削減、トレーニング時間の短縮、コンプライアンス対応の簡素化を迫られています。こうした目標を達成する手段の 1 つが、シングルベンダー CDS の導入です。他社製の CDS でラボを標準化する場合でも、世界トップクラスの性能を誇るアジレント製機器が使用できるようになります。ICF を活用することで、CDS ベンダーは他社製ソフトウェアシステムによるアジレント製 LC および GC 機器のコントロールに必要な開発労力の削減が可能となります。

Agilent
Trusted Answers

ICF の利点

ICF には次のような利点があります。

- さまざまな CDS で共通した操作性
- 150 種類以上のアジレント製 LC、CE、GC、ヘッドスペースモジュールと、そのあらゆるオプションや機能に対応
- 新機能や機器の追加を定期的にアップデート
- 機器のあらゆる機能へのシームレスなアクセスに対応するユーザーインタフェース
- モジュールごとのビューではなく、統合システムビューで複数システムの状態を一括確認

アジレントはオープンシステム構想を ICF の設計にとりいれました。ICF の導入前は、ソフトウェア開発者が機器コントロールコードを用いて各アジレント製機器のネイティブ機器コントロールドライバを作成していました。Agilent ICF により、ドライバ開発の労力を大幅に削減できます。ICF は必要な機器ドライバとユーザーインタフェースを備えているほか（図 1）、ソフトウェア接続用のシンプルなプログラミングインタフェースを搭載しています。ソフトウェアベンダーは、自社のソフトウェアと ICF をつなぐアダプタを開発するだけで、現在はもちろん将来的にもアジレント機器のフルコントロールが可能になります。追加の機器コントロールコードの作成は必要ありません。

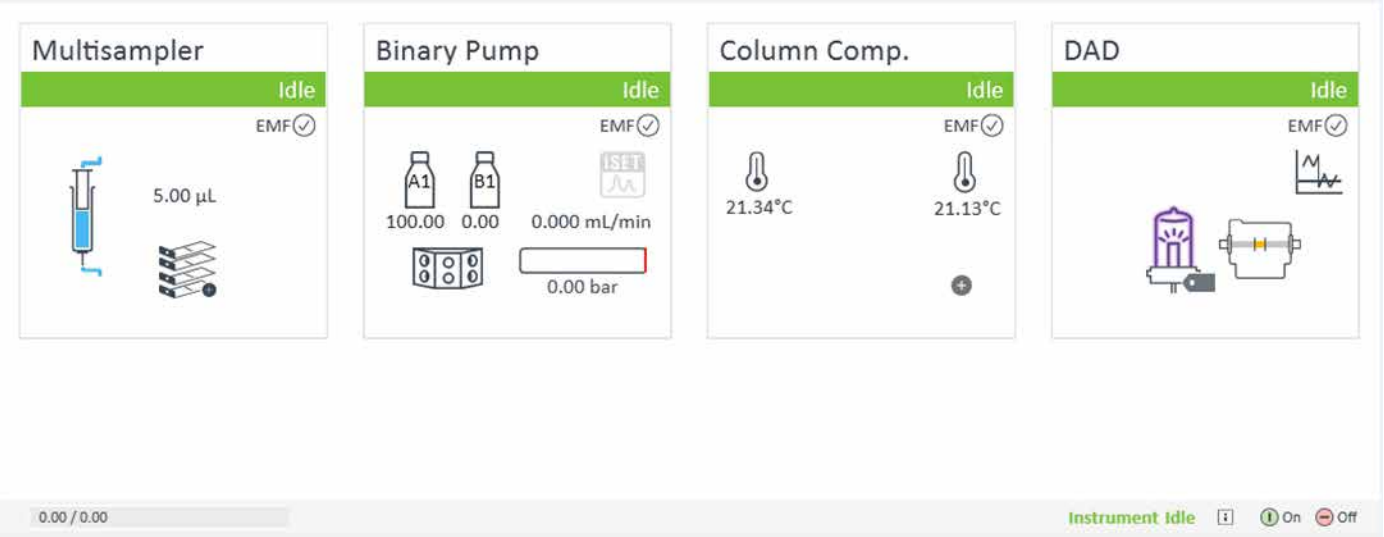

図 1. Agilent 1290 Infinity LC システムの機器コントロールユーザーインタフェース

機器コントロール

概要

一般的な分析機器コントロールプログラムには、次の項目で構成されています。

- アーキテクチャ概要
- 機能ブロック
- 構成モデル

分析アプリケーションのアーキテクチャは、モジュール型の機器または一体型の機器のコントロールに対する、図で示した階層構造によって定義されます（図 2）。表 1 に、図 2 で用いた用語の定義を示します。

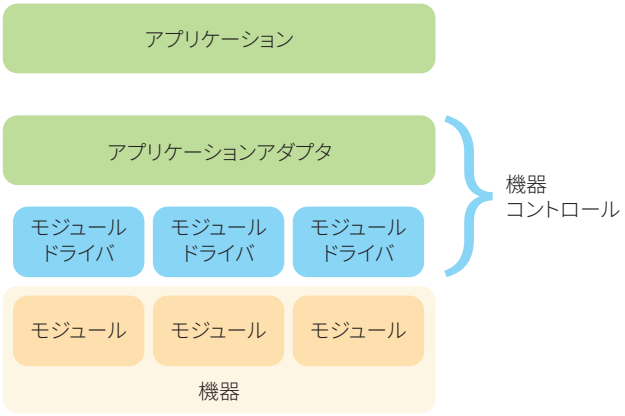

図 2. 機器コントロールの概要

表 1. 機器コントロールの用語の定義

アプリケーション	コアのソフトウェアとなる分析アプリケーション。シーケンス実行などの自動化のコントロール等を含む。
アプリケーションアダプタ	機器のドライバをアプリケーションインタフェースに変換するソフトウェアアダプタ。
モジュールドライバ	通信プロトコルのハンドリングなど、機器モジュールごとのコントロール機能をカプセル化するソフトウェア層。
機器	分析機器ハードウェア。ハードウェアオプションを備えた単一構成の機器である場合と、複数のハードウェアモジュールを組み合わせて構築したモジュール形式の機器である場合があります。

機能ブロック

機能で分類した場合、機器コントロールは一般的に、図 2 に示す機能ブロックで構成されます。ICF の機能ブロックの詳細については、付録 A を参照してください。

表 2. 機能ブロック

機器設定	物理機器のオプションや設定に関する記述
機器メソッド処理	分析に関連する一連のパラメータの処理を定義
機器ステータス表示	機器の状態を反映
機器のランコントロール	機器からの測定データの取り込みなど、分析の物理的な通信とコントロールの両方の処理

構成モデル

分析アプリケーションの構成モデルは、シングルノード（ワークステーション）から分散型（クライアント/サーバー）の構成まで、多岐にわたります（図 3a および 3b）。ICF の構成モデルの詳細については、付録 B を参照してください。

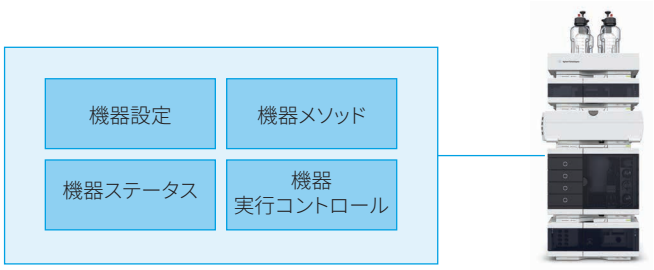

図 3a.一般的なワークステーション構成の例。機器のランコントロールを含め、すべての機能ブロックが 1 台のマシン内に存在。

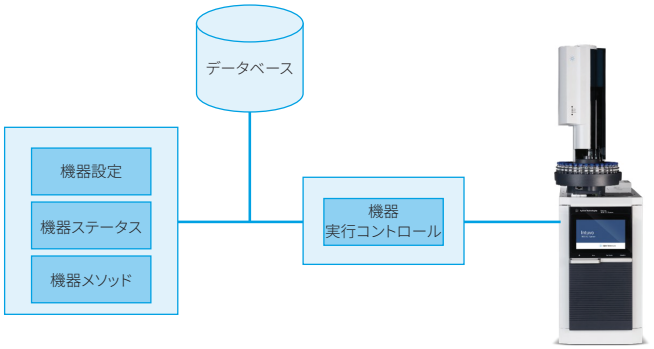

図 3b：クライアント/サーバー構成例。機器のランコントロールを除くすべての機能ブロックは、クライアントマシンごとに配置します。この例では機器のランコントロールは、異なるマシンに配置されています。このマシンは通常、装置の近辺に配置してデータ取り込みをコントロールします。機器コンフィグレーションや取り込みメソッドといった機器関連のデータは離れた場所に保管されます。

ICF と RC.NET による機器コントロール

アジレントは、統合された機器コントロールに対する要件に適応するため、RC.NET と ICF の関連した 2 つの標準化技術を開発しました。RC.NET は、ハードウェアドライバをモジュールレベルで実装するための技術を定義します。一方 ICF は、機器コントロールを機器レベルでアプリケーションに統合するためのインタフェースを定義します。つまり、ICF は複数の RC.NET ドライバを集約して、統合された機器ビューを作成します。

アーキテクチャ概要

RC.NET と ICF スタンダードを用いた階層の全体図を図 4 に示します。アプリケーションと機器の層は変わらないものの、機器ドライバとアプリケーションアダプタを必要に応じて細部までモデル化できます（図 4）。ホスト CDS やワークステーションのアーキテクチャの影響を受けずに機器モジュールをコントロールするには、機器ドライバ層を RC.NET ドライバに置き換えます。ICF の階層図の詳細は、付録 C を参照してください。

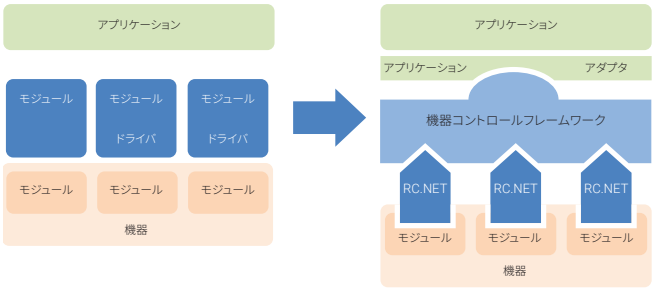

図 4. ICF と RC.NET を使用する際の機器コントロール階層図

アジレントはモジュールごとに機器ドライバを提供しています。ICF のドライバパッケージの詳細については、付録 D を参照してください。アプリケーションアダプタは、ICF の機能をホストの CDS ソフトウェアに統合して一体感を持って使用できるようにするために必要です。このアダプタはシンプルに ICF とインテグレーションするためのものです。機器のフルコントロールに必要な機能をインテグレーションします。**注：**ICF は、モジュールレベルでの機能の変更または拡張のためのツールではありません。ICF と RC.NET 以外のモジュールの連携の詳細については、付録 E を参照してください。

特長

- RC.NET により、アプリケーションに関係なく、ハードウェアモジュールのドライバを容易に実装できます。
- ICF は 、RC.NET ドライバの上に構成した階層を利用した複数 RC.NET ドライバの統合により、一括表示ビューで複数の機器状態を参照できます。
- 複数のドライバを集約し、機能と同期タスクの一部が ICF 層に組み込まれているため、アプリケーションアダプタ層は軽量となっています。

ICF と RC.NET のアーキテクチャ

RC.NET と ICF は、システム構成やアプリケーションインフラストラクチャとは無関係に、幅広いアプリケーションで動作するように設計されています。RC.NET と ICF の共通点としては、ホストアプリケーションソフト内の独立した構成要素であることに加えて、次のような性質と考慮すべき事項が挙げられます。

- コンポーネント同士の通信は、基盤であるアプリケーションインフラストラクチャ経由で、コンポーネント間の XML のデータ交換で実現しています。
- ユーザーインタフェースコンポーネントは機能コンポーネントから分離されているため、ホストアプリケーションの柔軟性が大幅に向上します。
- データコンテナ（設定、メソッド、前処理、ステータス）はカプセル化され、内部のデータはブラックボックスとして処理されます。
- RC.NET と ICF 間の通信手法をサポートするために、データコンテナは XML フラグメントとしてモデル化されます。**注：これらの XML フラグメントの構造と内容は、ホストアプリケーションでは使用されません。**
- データコンテナの構造は、ストレージの種類や構造を問わずコンポーネントが機能するように、ホストアプリケーションによって決定されます。
- ホストアプリケーションは、ドライバ固有の設定から切り離されています。そのため、ドライバのバージョンによる違いなども含め、ドライバ固有のコマンドをプログラミングレベルで把握して使用する必要はありません。
- データマイグレーションインタフェースにより、データコンテナの内容を異なるコンフィグレーション間で旧バージョンから新バージョンに転送することができます。
- データアクセスインタフェースは、必要に応じてデータを複写するメカニズムを使用して、データコンテナへの一般的なデータアクセスを可能にします。これにより、データ構造に対して、必要なデータ検索や修正などダイナミックなアクセスが可能になりました。
- ランコントロールは、シーケンス実行およびデータ解析機能から分離されています。
- ランコントロールは、シングルサンプル分析でオーバーラップ注入、同時デュアルインジェクション、ヘッドスペースオペレーションなどを対象としています。シーケンスコントロールと実行など、さらなる自動化についてはアプリケーションで処理する必要があり、それによって多様な自動化でコンポーネントを使用できます。
- サンプルアクイジションデータは、未処理の状態で透過的に機器からホストアプリケーションへと返されます。以降のデータの処理と画面表示のステップは、インタフェースから完全に切り離され、アプリケーション層内で実行されます。

さらに、ICF は対象の機器を中心として設計しています。コンポーネントとデータの処理や取り扱いは常に、単一モジュールレベルではなく、機器レベルで提供されます。機器レベルとは、RC.NET ドライバによってコントールされるすべてのモジュールが ICF の一部であることを意味しています。ICF が RC.NET ドライバ向けのプラグインアーキテクチャを提供します。そのため、新しいモジュールを簡単に構成できて、実装に手間がかかりません。インストールされた新しいモジュールは、アプリケーションで自動的に利用可能になります。ユーザーは新しい RC.NET ドライバをインストールするだけで、アプリケーション内で新規モジュールを使用できるようになります。

結論

ICF と RC.NET は相互に補完的な機器コントロールコンポーネントです。

- ICF は、目的の機器ハードウェアセットをサポートするために、RC.NET ドライバを必要とします。
- ICFは、RC.NET ドライバを組み合わせた階層を図で示したように、ドライバをつなぎ合わせることで機器全体の状態を把握できます。
- RC.NET はシングルモジュールのコントロールために設計されています。RC.NET により、モジュールをサポートするドライバをアプリケーションから独立して開発できます。

RC.NET により、アプリケーションに依存せずに機器ハードウェアのサポートが可能となります。ICF は、特定のユーザーアプリケーション内の分析機器の一般的なサポートを提供する技術です。

ICF によって、アジレント製機器ドライバと他社製 CDS との透過的な接続が可能になります。機器コントロールの観点から、ICF によりアジレント製 CDS の機能と同じ機能の提供が可能です。

これらのコンポーネントを組み合わせることで、CDS と分析機器のやりとりを標準化でき、他社製 CDS を用いたマルチベンダー機器コントロールが可能になります。

付録

付録 A.

ICF 機能ブロック

ICF は、データシステムの機能として要求される次のようなコンポーネントを提供します（図 5）。

メソッドコンポーネント：メソッドハンドリング機能（メソッドの監査証跡の作成、メソッドのレポート出力）

メソッド UI コンポーネント：メソッドパラメータ（メソッドパラメータ、タイムテーブル）を編集するためのユーザーインタフェース(User Interface : UI)。ヘルプやツールチップの機能も含みます。

コントロールコンポーネント：ハードウェアとの相互通信等のコントロール機能。(メソッドのダウンロード、分析の開始/停止、クロマトグラフィデータの収集)

コンフィグレーションコンポーネント：コンフィグレーションの処理機能（コンフィグレーション操作の監査証跡の作成、コンフィグレーションのレポート出力）

コンフィグレーション UI コンポーネント：コンフィグレーションを設定するための UI。ヘルプやツールチップなどを含みます。

ステータス UI コンポーネント（オプション）：機器ハードウェアの全体的なステータスを表示する UI。専用機能（UV ランプのオン/オフ）の直接的なコントロールを含みます。

前処理コンポーネント（オプション）：前処理用パラメータの処理機能。(ハードウェアでサポートされている場合)。前処理コンポーネントは、サンプラのカスタマイズされたサンプル処理手順を設定します。一部のアプリケーションでは、この機能はインジェクタプログラムとも呼ばれます。

前処理 UI コンポーネント（オプション）：前処理パラメータを編集するための UI

ICF は、上記のコンポーネント以外の動作やウェルプレートの編集など、広範な用途をサポートします。

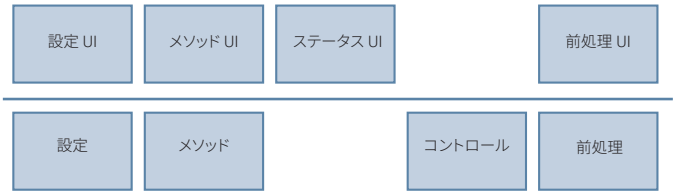

図 5. ICF 機能ブロック

付録 B.

ICF の機能モデル

ICF はコンポーネントを独立したブロックの組み合わせとして定義します。これらのコンポーネントは、アプリケーションのインフラストラクチャや構成モデル（例、ワークステーションまたはクライアント/サーバー）を基盤としてアプリケーションと連結が可能です。通信は、異なるコンポーネント間で XML フラグメントを渡すことで実現します。コンポーネント間で DCOM や RPC などの他のダイレクト接続を利用する必要はありません。これにより、コンポーネントは多様な構成モデルと互換性がとれます。図 6 に示すように、通信は取り込みコントローラーを別途使用します。

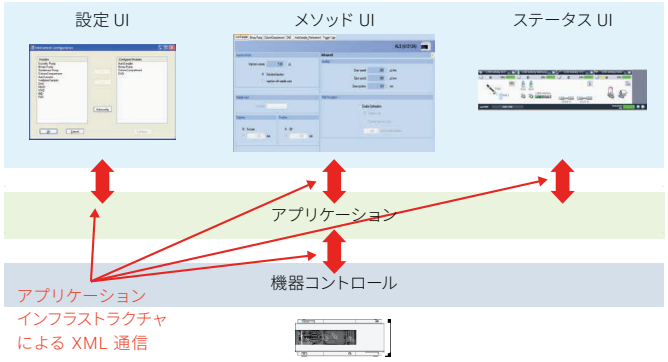

図 6. 取り込みコントローラーを用いた通信

付録 C.

ICF の階層モデル

階層間の通信プロトコルを階層化します。この構成による機器モジュールの通信プロトコルの大幅な簡素化を示しています。(図 7)。

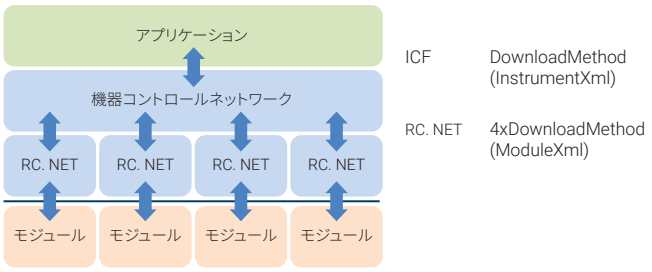

図 7. ICF の階層モデル図

付録 D.

ICF のドライバパッケージ

ICF の構成モデルは、フレームワークのインストールとドライバパッケージのインストールという 2 つの部分から成り立っています。

フレームワークのインストールによって、ホストアプリケーションのソフトウェアと共に指定の ICF コンポーネントがインストールされます。フレームワークのインストールにより、アプリケーションが ICF ドライバパッケージを使用できるようになります。フレームワークは、ユーザーが操作せずにホストアプリケーションの一部としてインストールできます。

ドライバパッケージのインストールでは、ドライバパッケージがインストールされます。ドライバパッケージは、機器のコンフィグレーション実施時にその選択したパッケージが使用可能なモジュールを定義します。

フレームワークの起動時に、フレームワークがシステムをスキャンし、ICF 内で利用可能なインストール済みパッケージを検索します。

付録 E.

ICF と RC.NET 以外のモジュールの連携

ICF は RC.NET 対応モジュールのシステムビュー提供だけでなく、RC.NET ドライバ以外のモジュールを含む「混成」機器を実行することも可能です。この場合、ホストアプリケーションは、ICF 機器と連携する RC.NET 以外のモジュールの同期をとる必要があります。この場合、ICF 層が「複合」モジュールドライバとして機能します（図 8）。

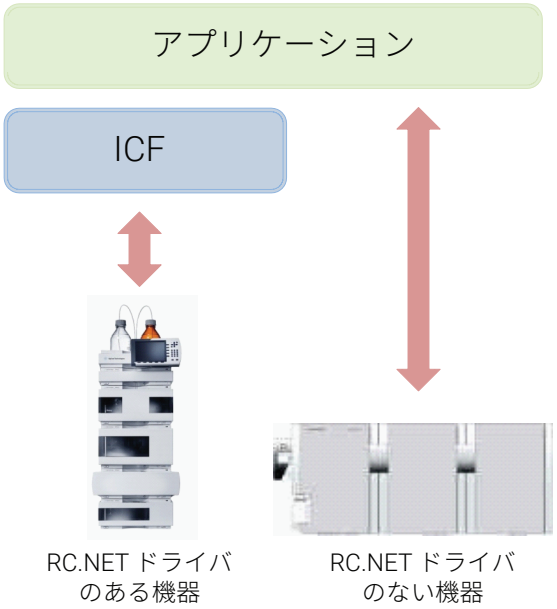

図 8. RC.NET 対応ドライバと RC.NET 以外の ドライバの混成機器環境

Agilent 機器コントロールフレームワークの詳細については、下記 URL を参照ください。

https://www.agilent.com/en/products/software-informatics/partner-program-for-openlab-software/instrument-control-program

ホームページ

www.agilent.com/chem/jp

カストマコンタクトセンタ

0120-477-111

email_japan@agilent.com

本製品は一般的な実験用途での使用を想定しており、
医薬品医療機器等法に基づく登録を行っておりません。
本文書に記載の情報、説明、製品仕様等は予告なしに
変更されることがあります。

アジレント・テクノロジー株式会社
© Agilent Technologies, Inc. 2018
Printed in Japan, July 19, 2018
5994-0049JAJP

Agilent
Trusted Answers