기술 개요

Agilent OpenLab

애질런트 기기 제어 프레임워크 (Agilent ICF)

타사 크로마토그래피 데이터 시스템을 사용하는 애질런트 기기를 쉽게 제어하기 위한 개방형 시스템 접근법

서론

본 기술 개요에서는 타사 크로마토그래피 데이터 시스템(CDS) 또는 기타 소프트웨어 솔루션을 사용하는 애질런트 액체 크로마토그래피(LC)와 가스 크로마토그래피(GC) 기기를 제어해주는 애질런트 기기 제어 프레임워크(ICF) 소프트웨어에 대해 소개합니다. 이 개요는 ICF와 RapidControl.NET(RC.NET) 기기 드라이버의 기술적 개념 및 소프트웨어 구성 요소가 어떻게 개방형 시스템 환경과 상호작용하는지에 대해 다룹니다.

오늘날의 경쟁적인 환경에서, 많은 실험실들은 비용을 절약하고, 교육 시간을 제한하며, 규제 준수 절차를 간소화해야 합니다. 단일 공급업체 CDS의 적용은 이 모든 목표를 달성하는 방법 중 하나입니다. 실험실이 타사 CDS로 표준화되어 있더라도, 계속해서 세계적 수준의 애질런트 기기를 사용할 수 있습니다. 이것을 실현하기 위해, CDS 공급업체는 ICF의 이점을 활용할 수 있습니다. ICF는 타사 시스템을 사용하는 애질런트 LC 및 GC 기기를 제어하기 위한 개발에 드는 노력을 훨씬 간소화해 드립니다.

Agilent

Trusted Answers

ICF의 이점

ICF는 사용자에게 다음과 같은 이점을 제공합니다.

- 여러 CDS에 적용되는 일관적인 외관 및 분위기
- 150개 이상의 다양한 옵션 및 기능을 포함한 애질런트 LC, CE, GC, Headspace 모듈 지원
- 정기적으로 새로운 기능을 통한 유지보수 업데이트 및 추가 기기에 대한 지원 제공
- 모든 기기 기능을 완벽하게 통합하는 포괄적인 사용자 인터페이스
- 모듈별 뷰 대신, 전체 상태를 볼 수 있는 통합 시스템 뷰 기능

ICF는 개방형 시스템에 대한 애질런트의 약속을 반영하고 있습니다. ICF 출시 이전, 소프트웨어 개발자들은 낮은 레벨의 기기 제어 코드를 사용하여 애질런트 기기를 위한 native-mode의 드라이버를 만들었습니다. 애질런트 ICF는 드라이버 개발에 드는 노력을 최소화합니다. ICF에는 필요한 기기 드라이버와 사용자 인터페이스(그림 1)를 포함하고 있으며, 소프트웨어 연결을 위해 간단한 프로그래밍 인터페이스를 제공합니다. 현재 및 미래의 애질런트 기기를 완전히 제어하기 위해, 소프트웨어 공급업체는 더이상 기기 제어 코드를 작성할 필요 없이 자사의 소프트웨어가 ICF에 수용할 수 있는 어댑터만 개발하면 됩니다.

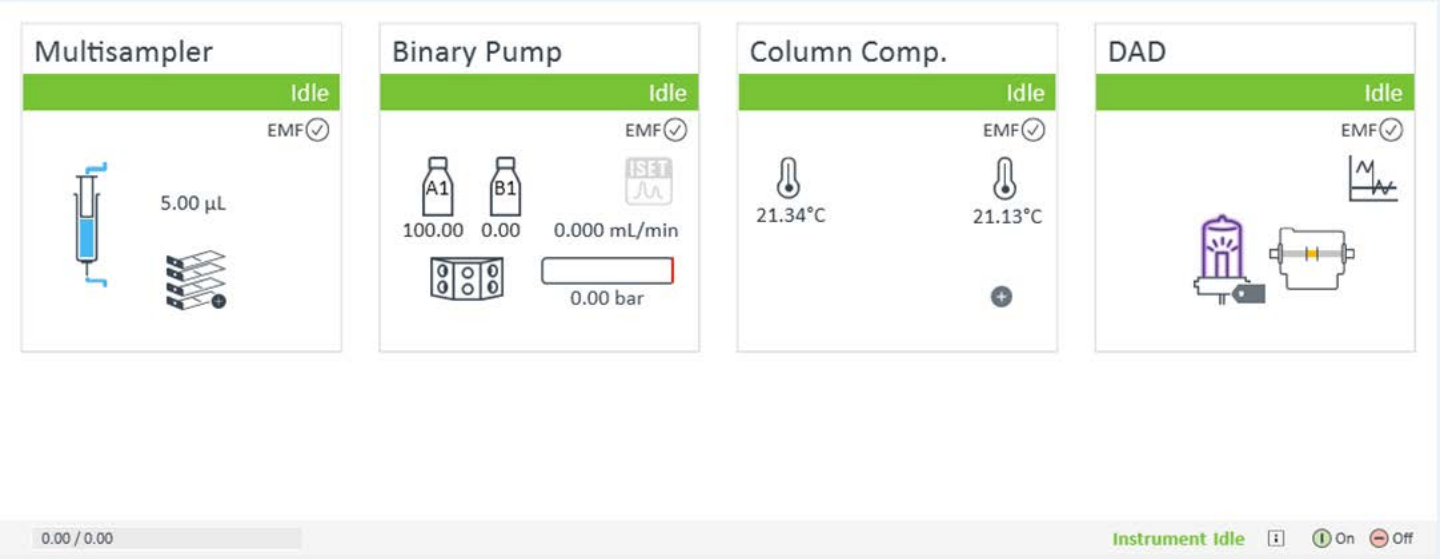

그림 1. Agilent 1290 Infinity LC 시스템을 위한 기기 제어 사용자 인터페이스.

기기 제어

개요

일반적인 분석 기기 제어 프로그램은 다음과 같이 특성 규명됩니다.

- 전체 아키텍처
- 기능 블록
- 배치 전략

분석 응용의 아키텍처는 모듈식 또는 모놀리식(monolithic) 기기 제어를 실현하기 위해 사용되는 추상 층에 의해 정의됩니다 (그림 2). 표 1은 그림 2에 사용된 전문 용어의 정의를 제공합니다.

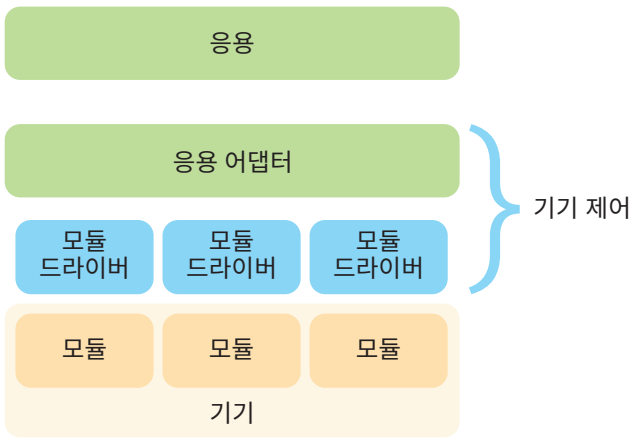

그림 2. 기기 제어 개요

표 1. 기기 제어의 전문 용어 정의

응용	핵심 분석 응용. 일반적으로 시퀀스 실행과 같은 제어 자동화 포함
응용 어댑터	기기 드라이버를 응용 인터페이스로 전환하는 소프트웨어 어댑터
모듈 드라이버	소프트웨어 층, 통신 프로토콜 처리와 같은 모듈 제어별 기능 포함
기기	분석 하드웨어. 하드웨어 옵션을 갖춘 모놀리식(monolithic) 기기 또는 다중 하드웨어 모듈 조합으로 구축된 모듈식 기기일 수 있습니다.

기능 블록

기능적 관점에서, 기기 제어는 보통 표 2에 기재된 기능 블록으로 구성됩니다. ICF 기능 블록에 대한 보다 자세한 정보는 부록 A를 참고하십시오.

표 2. 기능 블록

기기 구성	옵션 및 설정과 관련된 물리적 기기에 대한 설명
기기 분석법 처리	분석 실행과 관련된 파라미터 세트 처리 정의
기기 상태 표시	기기 상태를 반영합니다.
기기 실행 제어	기기로부터 측정된 데이터 수집을 포함한 물리적 통신과 분석 실행 제어를 모두 처리합니다.

배치 모델

분석 응용의 배치 전략은 단일 노드(워크스테이션) 또는 분산 (클라이언트/서버) 배치에 따라 달라질 수 있습니다(그림 3a 및 b). ICF 배치 모델에 대한 보다 자세한 정보는 부록 B를 참고하십시오.

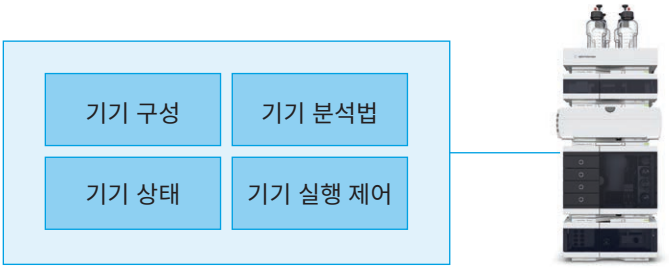

그림 3a. 일반적인 워크스테이션 배치의 사례. 기기 실행 제어를 포함한 모든 기능 블록은 단일 기계 내에 있습니다.

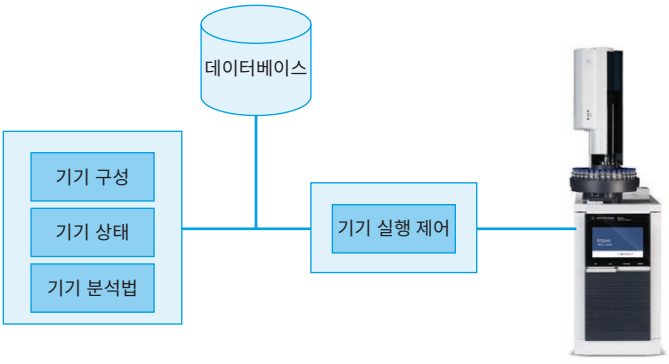

그림 3b: 클라이언트/서버 배치의 사례. 모든 기능 블록(기기 실행 제어 제외)은 클라이언트 기계에 분포됩니다. 이 사례에서, 기기 실행 제어는 보통 수집 컨트롤러로 작동하는 기기와 가까운 독립 기계 내에 위치합니다. 기기 구성 또는 수집 분석법과 같은 기기 관련 데이터는 원격으로 저장될 수 있습니다.

ICF 및 RC.NET 기기 제어

애질런트는 완벽하게 통합된 기기 제어 요건을 충족하기 위해 2개의 관련 표준인 RC.NET와 ICF를 개발하였습니다. RC.NET는 모듈 레벨의 하드웨어 드라이버 적용 표준을 정의하며, ICF는 기기 레벨에서 기기 제어를 응용에 통합하는 인터페이스를 정의합니다. 다시 말하면, ICF는 여러 RC.NET 드라이버를 통합하여 결합된 기기 뷰를 생성합니다.

전체 아키텍처

RC.NET 및 ICF 표준이 포함되어 있을 때의 일반적인 추상층을 그림 4에 나타냈니다. 응용 및 기기 층을 동일하게 유지할 수 있지만, 기기 드라이버와 응용 어댑터는 필요한 경우에 보다 정교한 입도로 모델링될 수 있습니다(그림 4). 호스트 CDS 또는 워크스테이션 아키텍처와 독립적으로 존재하는 기기 모듈을 제어하려면, 기기 드라이버 층을 RC.NET 드라이버로 대체해야 합니다. ICF 추상 층에 대한 보다 자세한 정보는 부록 C를 참고하십시오.

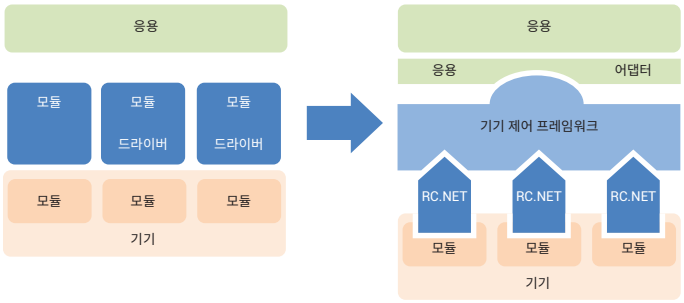

그림 4. ICF와 RC.NET 사용 시의 기기 제어 추상 층

애질런트는 기기 드라이버를 제공하며, 개별 모듈에 대해 특정된 드라이버가 있습니다. ICF 드라이버 패키지에 대한 보다 자세한 정보는 부록 D를 참고하십시오. 응용 어댑터는 ICF의 기능을 호스트 CDS로 투명하게 통합해야 합니다. 어댑터는 ICF만 통합하므로 복잡하지 않으며, 전체 기기 제어에 필요한 통합된 기능을 제공합니다. **주의:** ICF는 모듈 레벨에서 기능을 확장 또는 변경하기 위해 사용하는 도구가 아닙니다. ICF와 비 RC.NET 모듈과의 협업에 관한 보다 자세한 정보는 부록 E를 참고하십시오.

이점

- RC.NET는 응용과 독립되는 하드웨어 모듈용 드라이버 실현을 촉진합니다.
- ICF는 RC.NET 드라이버에 층을 구축하며, 이는 여러 RC.NET 드라이버를 모아 통합된 기기 뷰 기능을 제공합니다.
- 여러 드라이버가 모아져 있어, 일부 기능 및 동기화 작업은 ICF 층에 포함되며, 응용 어댑터 층은 가벼워집니다.

ICF 및 RC.NET 아키텍처

RC.NET와 ICF는 응용 프로그램 인프라 및 배치와 무관하게, 폭넓은 응용에서 모두 작동할 수 있도록 설계되었습니다. RC.NET와 ICF는 호스트 응용 내에 개별 구축 블록으로 사용할 수 있을 뿐만 아니라 다음과 같은 특성 및 고려사항을 공유합니다.

- 서로 다른 구성 요소 간 통신은 기초 응용 인프라의 구성 요소 사이의 XML 교환에 기반하고 있습니다.
- 사용자 인터페이스 구성 요소는 기능적 구성 요소와 분리되어 있고, 이는 호스팅 응용 사용 시나리오의 유연성을 극대화합니다.
- 데이터 컨테이너(구성, 분석법, 사전 처리, 상태)는 블랙 박스로 압축 및 처리됩니다.
- 데이터 컨테이너는 XML 조각으로 모델링되어 두 표준의 통신 패러다임을 지원합니다. **주의: 이들 XML 조각의 구조와 내용은 호스트 응용에 사용할 수 없습니다.**
- 구성 요소가 저장 유형 및 구조와 무관하도록, 데이터 컨테이너의 구조는 호스팅 응용에 따릅니다.
- 호스트 응용은 드라이버별 설정과 분리됩니다. 이는 드라이버 버전 의존성을 포함한 프로그램 레벨에서 드라이버별 명령을 파악해야 할 필요성을 제거합니다.
- 데이터 마이그레이션 인터페이스는 데이터 컨테이너의 내용을 구 버전에서 새로운 버전으로, 그리고 서로 다른 구성 간에 이전할 수 있도록 합니다.
- 데이터 액세스 인터페이스는 필요한 경우에 반사와 유사한 메커니즘을 사용해, 데이터 컨테이너에 대한 일반적 데이터 액세스를 제공합니다. 결과적으로 데이터 구조는 예를 들어 요구한대로 쿼리 및 수정이 가능하도록 동적 액세스 가능합니다.
- 실행 제어는 시퀀스 실행 및 데이터 분석 기능과는 분리되어 있습니다.
- 실행 제어의 범위는 단일 시료 분석이지만, 중복 주입, 이중 동시 주입 또는 헤드스페이스 작동에 대해서도 제공됩니다. 시퀀스 제어나 실행과 같은 진일보된 자동화는 응용으로 처리해야 하며, 그 결과, 서로 다른 자동화 시나리오에서 서로 다른 구성 요소가 사용될 수 있습니다.
- 시료 수집으로부터 얻은 데이터는 미처리 포맷의 상태로 호스트 응용에 투명하게 되돌아가며, 기기로 전달됩니다. 진일보된 데이터 조작 및 표시 단계는 인터페이스와 완전히 분리되어 있으며, 응용 프로그램 층의 범위에 속합니다.

또한 ICF는 기기 중심 설계를 제공합니다. 처리된 구성 요소와 데이터는 언제나 단일 모듈 레벨이 아니라 기기 레벨로 표시됩니다. 기기 레벨이라는 것은 RC.NET 드라이버로 제어되는 모든 모듈은 ICF의 일부임을 의미하며, 이는 RC.NET 드라이버용 플러그-인 아키텍처를 제공합니다. 결과적으로 새로운 모듈이 많은 실현 노력 없이 쉽게 배치될 수 있습니다. 일단 설치되면, 새로운 모듈은 자동으로 응용에 사용될 수 있습니다. 고객은 단지 응용에 새로운 모듈에 사용할 새 RC.NET 드라이버를 설치하기만 하면 됩니다.

결론

ICF와 RC.NET는 보완적인 기기 제어 구성 요소입니다.

- ICF는 정의된 하드웨어 세트를 지원하기 위해 RC.NET 드라이버 필요
- ICF는 추상 층을 도입해 서로 연결되어 완전한 기기를 표시하는 모든 RC.NET 드라이버의 통합된 뷰 제공
- RC.NET는 단일 모듈 제어를 위해 설계됨. RC.NET는 응용 프로그램에서 독립적인 모듈 지원을 위해 드라이버를 개발할 수 있도록 지원.

RC.NET는 응용 프로그램에서 독립적인 기기 하드웨어 지원을 실현하기 위한 방법을 제공합니다. ICF는 특정 사용자 응용 범위 내의 일반적인 분석 기기 지원을 제공하는 기술을 제공합니다.

ICF는 타사 CDS를 연결할 수 있도록 함으로써, 애질런트 기기 드라이버의 투명한 연결을 가능하게 합니다. 기기 제어 방면에, ICF는 애질런트 CDS와 동일한 기능을 제공합니다.

이들 구성 요소는 함께 CDS가 분석 기기와 상호작용하는 방법을 표준화함으로써, 타사 CDS를 사용하는 여러 제조업체의 기기 제어를 가능하도록 합니다.

부록

부록 A.

ICF 기능 블록

ICF는 일반적 데이터 시스템 기능을 포함하는 구성 요소들을 제공(그림 5).

분석법 구성 요소: 분석법 처리 기능(두 분석법을 위한 감사 추적 생성, 분석법 보고서 작성)

분석법 UI 구성 요소: 도움말과 도구 팁을 포함한 분석법 파라미터 편집용 UI(분석법 파라미터, 시간표)

제어 구성 요소: 하드웨어와 실행 제어 기능의 상호작용(분석법 다운로드, 실행 시작/중지, 크로마토그래피 데이터 수집)

구성 요소: 구성 처리 기능(두 구성을 위한 감사 추적 생성, 구성 보고서 작성)

구성 UI 구성 요소: 도움말과 도구 팁을 포함한 구성 편집용 UI

상태 UI 구성 요소(옵션): 전용 기능을 위한 직접 상호작용(UV 램프 켜기/끄기)을 포함한 기기 하드웨어의 전체 상태를 보여주는 UI

사전 처리 구성 요소(옵션): 사전 처리 파라미터, 하드웨어가 지원되는 경우만. 사전 처리는 샘플러의 맞춤화된 시료 처리 단계를 정의합니다. 일부 응용에서, 이 기능은 주입기 프로그램으로도 알려져 있습니다.

사전 처리 UI 구성 요소(옵션): 사전 처리 파라미터 편집용 UI

위의 구성 요소에 이외, ICF는 조작 또는 well plate 편집과 같은 추가사용 사례도 지원합니다.

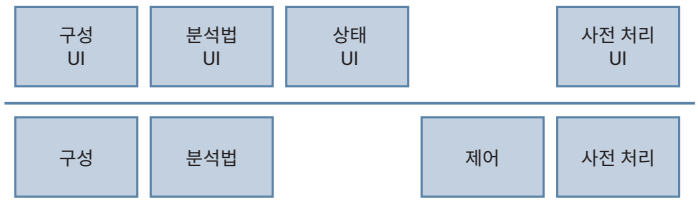

그림 5. ICF 기능 블록

부록 B.

ICF 배치 모델

ICF는 구성 요소들을 응용 독립적인 인프라와 배치 전략을 갖춘 응용 프로그램에 의해 함께 사용되고 연결될 수 있는 독립적인 구축 블록으로 정의합니다(예를 들어, 워크스테이션 또는 클라이언트/서버 환경). 서로 다른 구성 요소 간에 XML 조각 이동 시 통신을 발생합니다. DCOM 또는 RPC와 같은 구성 요소 사이에는 직접적인 기타 연결이 없습니다. 따라서 구성 요소들은 서로 다른 배치 전략과 호환됩니다. 그림 6에서 볼 수 있듯이, 통신은 개별 수집 컨트롤러를 사용합니다.

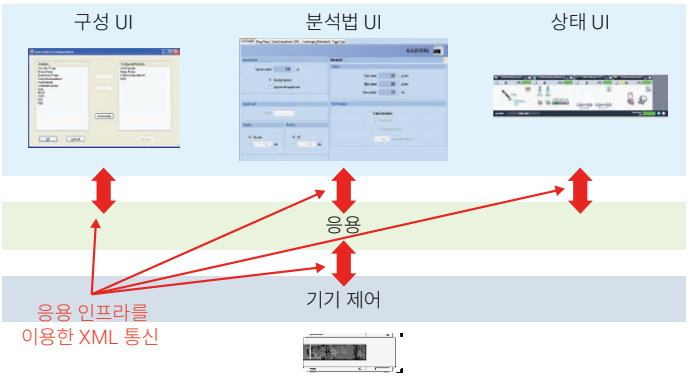

그림 6. 개별 수집 컨트롤러를 사용한 통신

부록 C.

ICF 추상 층

층 간의 통신 프로토콜의 서로 다른 레벨은 모듈식 기기의 프로토콜이 한층 더 간소화되었음을 보여줍니다(그림 7).

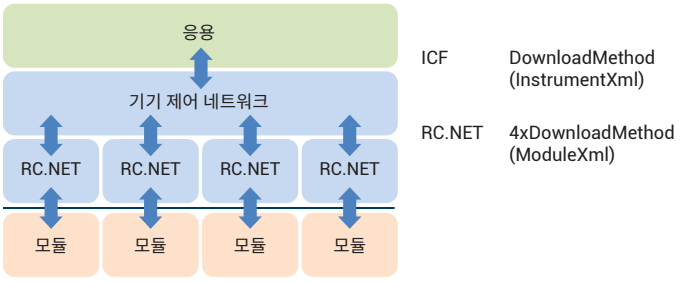

그림 7. ICF 추상 층

부록 D.

ICF 드라이버 패키지

ICF용 배치 전략은 프레임워크 설치와 드라이버 패키지 설치 두 가지 부분에 기반합니다.

프레임워크 설치는 호스트 응용의 콘텍스트 내에서 ICF 구성 요소의 버전을 설치합니다. 프레임워크의 설치는 응용 프로그램이 ICF 드라이버 패키지를 사용하도록 합니다. 프레임워크는 사용자 상호작용 없이, 호스트 응용의 일부로서 설치될 수 있습니다.

드라이버 패키지 설치는 드라이버 패키지를 설치합니다. 드라이버 패키지는 기기 구성 과정 중에 해당 패키지가 선택된 경우 사용 가능한 모듈을 정의합니다.

프레임워크 시작 과정 중, 프레임워크는 ICF 내에서 사용할 수 있는 설치된 패키지를 위해 시스템을 스캔합니다.

부록 E.

ICF와 비 RC.NET 모듈의 협업

ICF가 RC.NET 이용 가능 모듈을 위한 시스템 뷰를 제공하지만, RC.NET 드라이버가 없는 모듈을 포함한 "혼합" 기기를 실행하는 것은 가능합니다. 이러한 경우, 호스트 응용은 ICF 기기를 개입된 비-RC.NET 모듈과 동기화해야 합니다. 이 시나리오에서 ICF 층은 "혼합" 모듈 드라이버로 작동됩니다(그림 8).

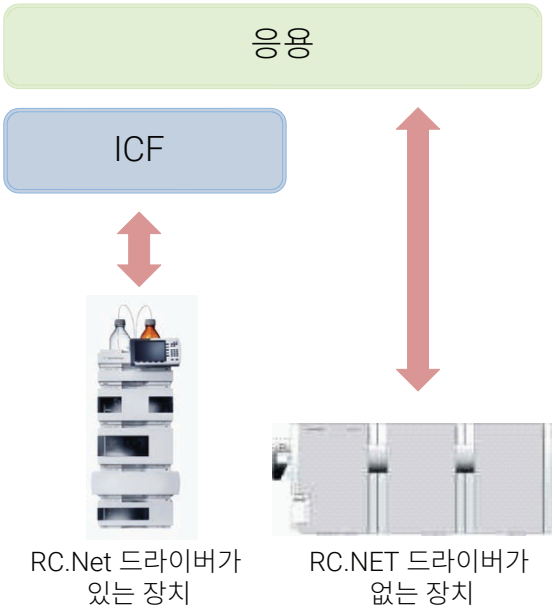

그림 8. RC.NET 기능 및 비-RC.NET 드라이버를 갖춘 혼합 기기 환경

애질런트 기기 제어 프레임워크에 대해 더 자세히 알아보시려면 다음 사이트를 방문하십시오.
https://www.agilent.com/en/products/software-informatics/partner-program-for-openlab-software/instrument-control-program

www.agilent.com

2018년 7월 9일, 한국에서 인쇄
5991-0049KO

서울시 용산구 한남대로 98, 일신빌딩 4층 우)04418
한국애질런트테크놀로지스(주) 생명과학/화학분석 사업부
고객지원센터 080-004-5090 www.agilent.co.kr

Agilent
Trusted Answers