IBM Rational Rhapsody and Rational Rose
백서
2009년 6월

Rational. software

IBM Rhapsody 소프트웨어로 전환하십시오.

생산성 및 품질 향상을 위한 지침

Scott Niemann, IBM

목차

2 전체 개요
2 MDD와 CASE
3 IBM Ration Rhapsody 와 IBM Rational Rose
7 영역별 모델링
14 결론

IBM Rational Rose는 UML을 시스템 설계 환경에 적용하는 데 필요한 단계를 설정합니다. IBM Rational Rhapsody는 모델 기반 개발(MDD) 솔루션과 새롭게 진화하는 기술 및 개발 표준을 처리하는 추가 기능들을 함께 제공합니다.

전체 개요

지난 10년 간 UML(Unified Modeling Language)을 사용한 모델링은 시스템 아키텍처와 설계 의도를 설명하는 강력한 수단으로 진화했습니다. IBM® Rational® Rose® 소프트웨어를 통해 IBM은 UML을 시스템 시장에 도입하는 데 핵심 주자 역할을 수행해 왔습니다. Rational Rose 소프트웨어는 UML을 통해 시스템의 설계를 지원할 뿐만 아니라 소프트웨어에 포함된 아키텍처 프레임워크를 다른 라이프사이클 개발 도구와 통합하여 개념 구축에서 대상 시스템에 이르는 완전한 솔루션을 구현하게 해줍니다. 더욱이 기술 및 개발 표준에 따라 업데이트되어, IBM Rational Rhapsody® 소프트웨어와 같은 최신 솔루션이 새로운 기능을 제공할 수 있도록 유용성이 한층 강화되었습니다.

Rational Rose는 시스템과 소프트웨어 아키텍처 설계 시 수반되는 복잡성을 줄이고, 엔지니어가 쉽게 이해하고 도입할 수 있는 UML을 공용 언어로 활용하여 협업 기능을 강화할 목적으로 개발된 애플리케이션입니다. 개발에 따른 효과는 매우 성공적이었으나 기술의 발전과 시스템 복잡성의 증가를 극복하려면 CASE(Computer Aided System Engineering) 기술에 존재하는 간극을 메울 수 있는 솔루션이 필요합니다. 모델 기반 개발(MDD)을 사용하면 테스트와 구현이 가능하며 보유 중인 기존 지적 재산을 Rational Rose 모델 또는 소스 코드의 형태로 쉽게 활용할 수 있습니다.

MDD와 CASE

모델 기반 개발은 UML 2 모델을 사용하여 설계 뿐만 아니라 요구사항과 구현까지도 명시합니다. 이렇게 다양한 부분의 설계 프로세스 사양이 실행 가능하며 시스템 개발 중에 이를 검증할 수 있습니다. Rational Rose 솔루션 구현의 기반이 되는 CASE 기술은 기본적으로 시스템 아키텍처 문서화와 구현 진행 방법 결정을 위해 사용됩니다. 그러나, 문서는 정적이기 때문에 CASE 기술은 사양이 올바른지 여부 또는 시스템이 작동하는 방식을 검증하는 최적의 방법을 제공할 수 없습니다. 그러므로 설계의 초기 단계에서 시스템에 오류가 발생하더라도 구현이 완료될 때까지 발견되지 않을 수 있습니다.

주요 특징

그림 1은 라이프사이클 초기에 오류를 발견하지 못할 경우 부담하게 되는 비용이 얼마나 늘어나는지를 보여줍니다. 고수준(HL) 및 소프트웨어(SW) 설계 초기 단계에 발생하는 대다수의 오류는 주로 요구사항에 대한 오해에서 비롯됩니다. 구현 단계 이전에 오류를 포착하면 비용과 시간을 모두 절약할 수 있습니다. 라이프사이클이 진행되고 아키텍처와 구현 기능들이 추가될수록 설계 오류를 찾아내 수정하는 데 소요되는 비용은 기하급수적으로 늘어날 수 있습니다.

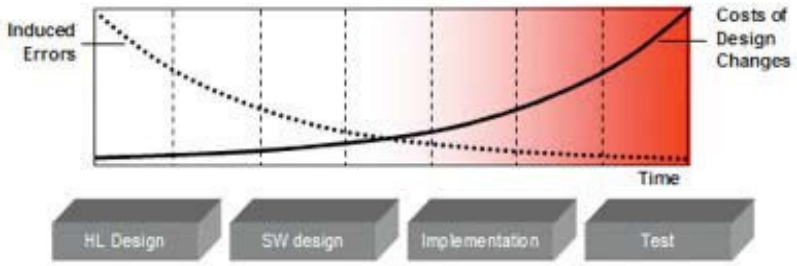

그림 1. 설계 단계의 오류 비용

Rhapsody는 구현 이전에 오류를 찾아 수정함으로써 시간과 비용을 절약할 수 있도록 도와줍니다.

IBM Rational Rhapsody를 통한 MDD 방식의 목표는 HL 설계 초기에 결함을 발견하여 비용 지출을 최소화하는 것입니다. 설계 단계에 오류를 수정하면 비용을 절감할 수 있을 뿐만 아니라 개발에 소요되는 시간도 단축됩니다.

IBM Ration Rhapsody 및 IBM Rational Rose

넓게 보면, IBM Rational Rhapsody 소프트웨어가 최신 UML 표준뿐만 아니라 SysML(Systems Modeling Language)과 같은 영역별 표준 프로파일도 지원하기 때문에 Rational Rose는 Rhapsody의 부분 집합입니다. 이러한 유연성 덕택에 개발자는 문제가 되는 영역을 기술하는 데 가장 적절한 설계 언어를 사용할 수 있습니다. MDD 플랫폼인 IBM Rational Rhapsody는 기본적으로 단순한 모델링을 뛰어넘는 환경의 여러 가지 측면에 사용됩니다. 해당되는 측면에 대해서는 이 백서의 후반부에서 다시 살펴보겠습니다. 백서에서 설명하는 마이그레이션 프로세스에서는 Rational Rose 소프트웨어 사용자가 데이터를 마이그레이션한 후에 IBM Rhapsody의 고급 기능을 활용할 수 있습니다.

IBM Rational Rose 사용자는 데이터를 쉽게 Rhapsody 소프트웨어로 마이그레이션하여 고급 MDD 기능의 장점을 활용할 수 있습니다.

주요 특징

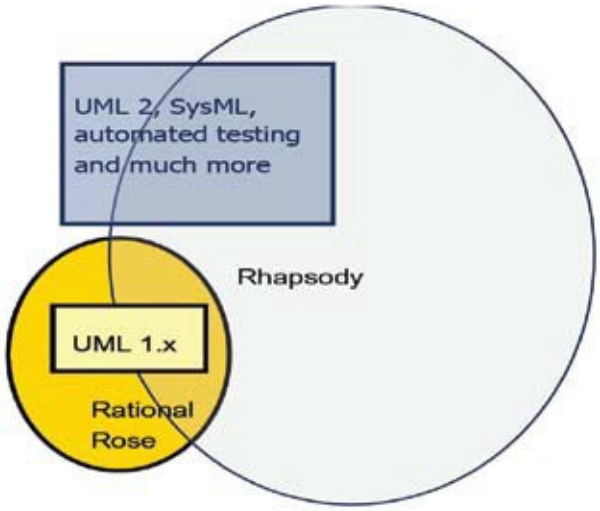

그림 2. IBM Rational Rose 및 Rational Rhapsody

IBM Rational Rhapsody의 강력한 모델 기반 테스트 및 디버그 기능은 이 솔루션을 업계의 경쟁 제품과 차별화합니다.

모델 기반 테스트

Rational Rhapsody를 차별화하는 주요 요소는 테스트 기능, 특히 모델 기반 테스트입니다. 이 기능을 통해 엔지니어는 아키텍처 구축 중에 유효성을 검증하여 개발 프로세스 초기 코드 작성 이전에 결함을 제거할 수 있습니다. 이 기술은 설계 단계 후반에 오류를 수정할 때도 적은 시간이 소모되기 때문에 낮은 비용으로 개발을 가속화하는 데 도움이 됩니다. 또한, 회귀 테스트가 매일 밤 실행되도록 설계 시 테스트 실행을 자동화할 수 있습니다.

모델 기반 테스트의 핵심이 되는 두 가지 측면은 모델 수준 디버깅과 자동화된 요구사항 기반 테스트입니다.

모델 수준 디버깅

모델 수준 디버깅은 시스템 구축 시 시스템을 테스트하고 시스템에 구현 시 결함을 줄여 줍니다. 디버깅의 정밀도를 여러 레벨로 조정할 수 있으며 시스템 레벨과 소스 레벨 구현이 가능합니다.

주요 특징

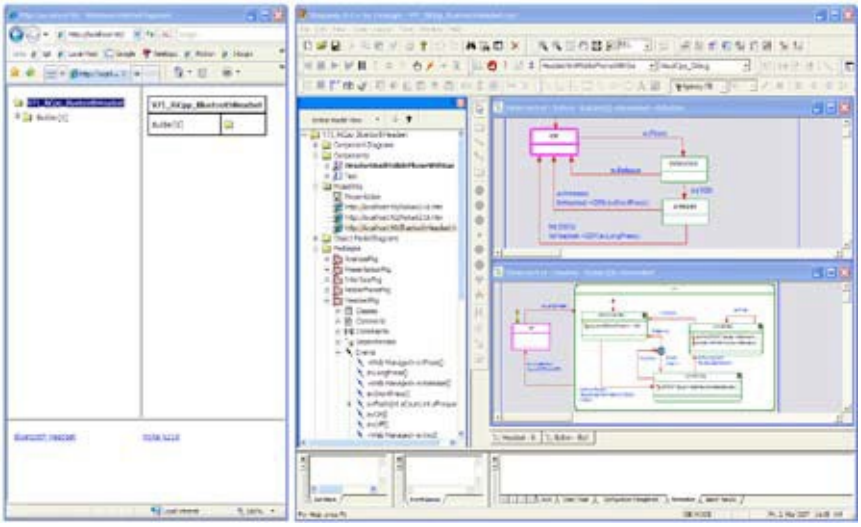

그림 3. 모델 수준 디버깅

시각적인 모델 레벨 디버깅으로 구현 시 구성요소가 구성되기도 전에 통합 시스템이 작동하는 방법을 확인할 수 있습니다.

그림 3에서, 시스템 실행 시 오른쪽의 설계가 강조표시됩니다. 따라서 특정 시스템 구성요소에서의 현재 위치가 서브시스템인지 또는 실제 클래스인지 정확히 파악할 수 있습니다. 왼쪽의 HTML 패널은 설계에 자극을 주입하는 데 사용됩니다. 이 패널은 Rational Rhapsody 애플리케이션에서 자동으로 생성되므로 패널에 표시되는 이벤트를 클릭하여 시스템을 간편하게 테스트할 수 있습니다. 휴대전화와 같이 개발 중인 디바이스 시스템인 경우 패널을 확장하여 시스템을 더욱 실제와 유사하게 표현할 수 있습니다. 이러한 방식을 통해 고객, 판매자, 엔지니어 모두에게 설계 의도를 효과적으로 알릴 수 있습니다.

개발자들은 흔히 하드웨어를 사용할 수 없는 채로 시스템의 기능 테스트를 수행해야 합니다. 이 때 모델 레벨 디버깅을 사용하면 수명 주기 중 현재 단계에서 각종 결함이 제거되기 때문에 통합과 시스템 테스트에 소요되는 시간을 줄일 수 있습니다. 하드웨어를 사용할 수 있게 되면 애플리케이션을 대상 플랫폼에 구축할 수 있으며, 대상에서 호스트 플랫폼으로의 링크가 연결되어 있을 경우 모델 레벨 디버깅을 실제 대상에서 수행할 수 있습니다.

주요 특징

테스트 시나리오를 요구사항 데이터베이스에 연결하면 각각의 테스트가 고객의 요구사항을 검증하는 방법 및 컴플라이언스를 보장하도록 각 요구사항을 테스트하는 방법을 추적 및 확인할 수 있습니다.

요구사항 기반 테스트

요구사항 기반 테스트를 활용하면 시스템이 IBM Rational DOORS®, IBM Rational RequisitePro®와 같은 요구사항 데이터베이스에서 또는 Microsoft® Word/Excel 문서로 유지관리되는 고객의 요구사항을 충족하는지 검증할 수 있습니다. 이러한 요구사항을 MDD 환경에서 모델 요소로 표현하고 시스템이 고객의 요구사항을 만족시키는 방법을 설명하는 시나리오와 연결할 수 있기 때문에 이러한 시나리오를 회귀 테스트를 위해 자동으로 구축 및 실행이 가능한 테스트 벡터로 활용할 수 있습니다. 시나리오는 단순히 외부 자극에 대한 시스템의 예상 작동을 설명합니다. 시나리오와 고객 요구사항을 연결하면 요구사항 데이터베이스와 동기화 및 추적 정보를 제공할 수 있습니다. 그림 4에서 이 프로세스를 보여주며, 요구사항 데이터베이스를 먼저 Rhapsody 소프트웨어로 가져온 다음 설계 수준 테스트 벡터로 연결합니다.

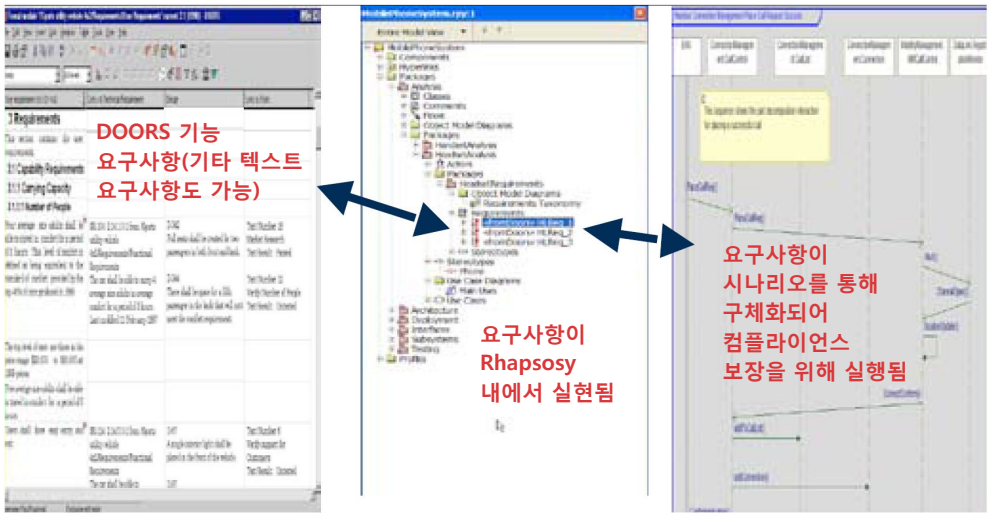

그림 4. 설계 요구사항 링크 테스트

주요 특징

MDD 환경의 실행 결과에 따라 모델링된 통신 시스템에서 요구사항이 충족되는지 여부를 판별할 수 있습니다. 이는 그림 5에서 오른쪽의 테스트 벡터를 결과의 성공/실패를 가늠하는 벤치마크로 사용할 수 있다는 의미입니다. 실행 모델은 시스템이 실제로 수행 중인 작업을 표시하고 이를 벤치마크와 자동으로 비교할 수 있습니다. 추적성은 결과 실행 시스템을 통해 충족된 고객 요구사항과 여전히 컴플라이언스가 필요한 영역을 손쉽게 파악할 수 있도록 도와 줍니다. 이 방법으로 고객의 기대에 부응하는 시스템을 보다 편리하고 효율적으로 구현할 수 있습니다.

UML의 강력한 기능을 영역별 프로파일과 결합하면 업종 고유의 설계 의도를 명확히 반영한 모델을 작성할 수 있습니다.

영역별 모델링

UML은 모델링을 위한 강력한 언어이지만 시스템 정보에 대한 일반적인 템플릿만을 제공합니다. 모호함을 줄이고 동료와 고객에데 설계 의도를 명확히 전달하려면 해당 영역에 보다 가까운 모델링이 필수적입니다. IBM Rational Rhapsody에서 지원하는 최신 버전의 SysML은 시스템 엔지니어가 고수준 시스템 구성요소를 구현하고 부속 서브시스템을 모델링하는 데 필요한 기능 분석 접근법을 사용할 수 있도록 해줍니다.

Rhapsody 소프트웨어는 소프트웨어 및 시스템 엔지니어링과 같은 기존 분야 외에도 자동차, 우주 항공, 방위 및 통신 산업 분야에 대한 영역별 프로파일을 제공합니다. 또한 조직에서 진정한 설계 의도를 효율적으로 명시할 수 있도록 프로파일을 작성할 수 있습니다. 그림 5는 우주 항공 및 방위 시장에서 사용되는 프레임워크인 DoDAF(Department of Defense Architecture Framework)를 Rhapsody 솔루션을 통해 표현한 예입니다.

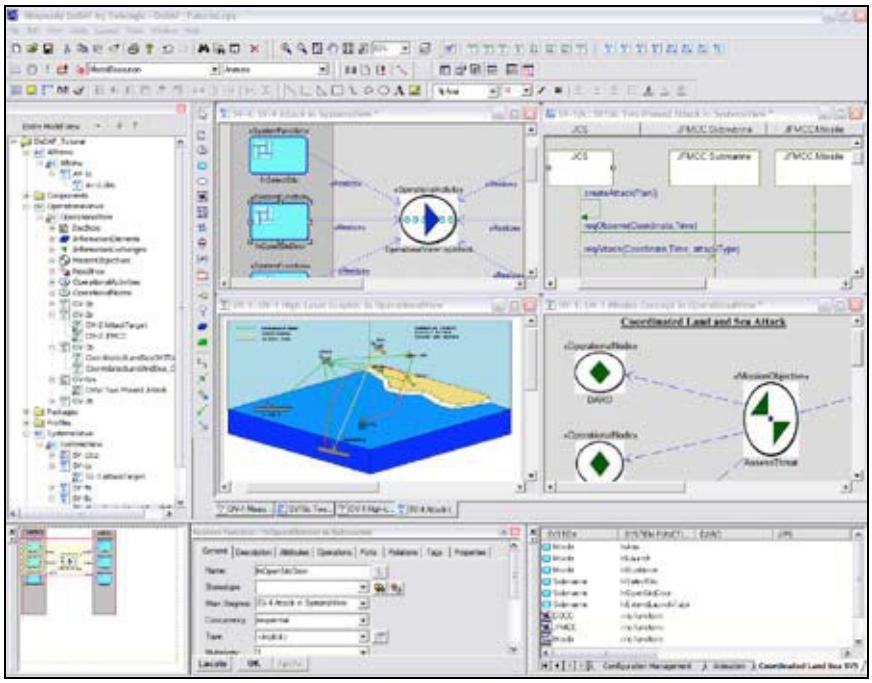

그림 5. 업종별 프로파일의 예 - DoDAF

조직별 목표에 따라 프로파일을 생성할 수 있습니다. 프로파일은 UML 2의 확장이므로 모델 교환에 사용되는 XMI(XML Metadata Interchange)와 호환이 가능합니다. 시스템 설계자는 Rhapsody 소프트웨어에서 모델링 템플릿으로 사용할 프로파일을 모델링하여 조직의 시스템 개발자에게 해당 템플릿을 배포할 수 있습니다. 이러한 방법을 사용하면 우수 사례와 지침을 준수하는 데 매우 효과적입니다.

주요 특징

Rhapsody 소프트웨어는 다양한 구성 관리 도구화 통합이 가능하고 협업 개발 환경을 지원하는 유연한 코딩 환경을 제공합니다.

협업

Rational Rhapsody의 다양한 장점은 강력한 소프트웨어 구현을 위해 모든 노력을 기울이는 개발 팀에서도 그 진가를 발휘합니다. Rational Rhapsody 소프트웨어는 모델과 코드가 통합된 환경을 제공합니다. Rational Rhapsody에서는 작업을 작성하고자 할 때 C 언어로 직접 코드를 쓰는 방법으로도 작성이 가능합니다. 정보는 모델에 저장됩니다. 모델과 코드를 별도로 관리하지 않고 구성 관리(CM) 리포지토리에서 체크인 또는 체크아웃하는 하나의 모델 요소를 통해 병행 개발을 대폭 줄이고 단순화할 수 있습니다.

시각적 차이 비교 및 병합

Rhapsody 소프트웨어의 시각적 차이 비교 및 병합 기능은 모델과 코드의 통합을 가능하게 해줍니다. 그림 7에서와 같이 Rhapsody 소프트웨어는 자동 병합 기능을 기본 제공할 뿐만 아니라 IBM Rational ClearCase® 소프트웨어와 같은 CM 도구와 통합할 수도 있습니다. 예를 들어, Rational ClearCase 소프트웨어의 자동 병합 기능을 실행하면 Rhapsody 솔루션 차이 비교 및 병합 도구에서 자동으로 설계(모델 및 코드)를 통합하도록 설계되었습니다. 충돌이 발생할 경우, 시스템 개발자는 IBM Rational Rhapsody에서 차이점을 그래픽으로 확인하고 문제를 해결할 수 있습니다. 모델과 코드 병합을 통해 통합을 간소화하고 통합 시간을 대폭 줄일 수 있습니다.

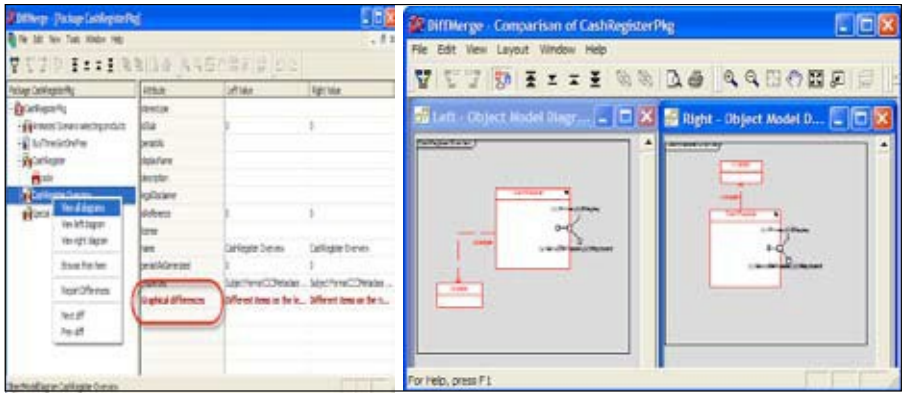

그림 6. Rhapsody 소프트웨어 차이 비교 및 병합

주요 특징

그림 6에서 왼쪽 창은 모델 데이터를 보여주며, 병합이 필요한 차이점이 있는지 여부를 표시합니다. 모델 데이터 중 일부가 코드일 경우 개발자는 공통 텍스트 차이 비교 도구가 포함된 Rational Rhapsody 솔루션의 통합 기능을 통해 Rational ClearCase 소프트웨어 또는 다른 CM 도구와 같은 원하는 차이 비교 도구를 선택하여 사용할 수 있습니다. 단, 이러한 도구는 변경사항을 자동으로 병합할 수 없을 때만 필요합니다.

그래픽의 경우, 그림 6에서 오른쪽의 다이어그램에는 해결할 수 없는 변경사항을 표시하는 방법이 나와 있습니다. 사용자는 해당 변경사항을 단계별로 처리하여 병합 후보를 작성할 수 있습니다.

이 기능은 Rational ClearCase 애플리케이션의 유형 관리자 기능에 통합되어 자동 병합 및 Rational Rhapsody 소프트웨어 구별/병합 기능은 실제로 Rational ClearCase 소프트웨어 명령행 또는 버전 트리를 통해 수행됩니다.

Rational Rhapsody의 동기화 기능을 사용하면 코드 변경을 모델에 반영하고 모델 변경을 코드에 반영할 수 있습니다.

코드 중심 워크플로우

대부분의 개발자에게는 효율적이고 효과적인 코딩 기능이 무엇보다 중요합니다. Rational Rhapsody 솔루션은 쉽게 코드를 작성하고, 손으로 직접 작성해야 하는 코드가 자동으로 출력으로 제공되는 모델을 설계할 수 있게 해주는 강력한 도구를 다양하게 제공합니다. 어떠한 방법을 선택하더라도 Rational Rhapsody 소프트웨어를 사용하면 코드와 모델이 항상 동기화되고 설계 문서가 구현을 정확히 표현할 수 있습니다. 코드 중심의 워크플로우는 시각화를 통한 구성요소 재사용과 동기화로 세분화됩니다.

주요 특징

IBM Rhapsody를 사용하면 Rational Rose 사용자가 Rhapsody의 고급 모델링 및 테스팅 기능을 활용하여 지적 재산을 재사용할 수 있습니다.

코드 동기화

Rational Rhapsody의 코드 동기화 기능은 사용자가 코드를 가장 익숙하고 효율적인 방법으로 작성하면서 설계와 모델의 동기화를 유지할 수 있게 해줍니다. 편집 환경에 코드를 입력하면 시스템에서 자동으로 모델을 업데이트합니다. 이러한 "코드 중심" 기능을 사용하면 소스 파일에서 정보의 배치가 모델링된 환경에 반영되도록 보장됩니다. 즉, 데이터 및 작업을 파일의 특정 위치에 추가하면 해당 정보가 향후 엔지니어링 작업 시 그대로 유지됩니다.

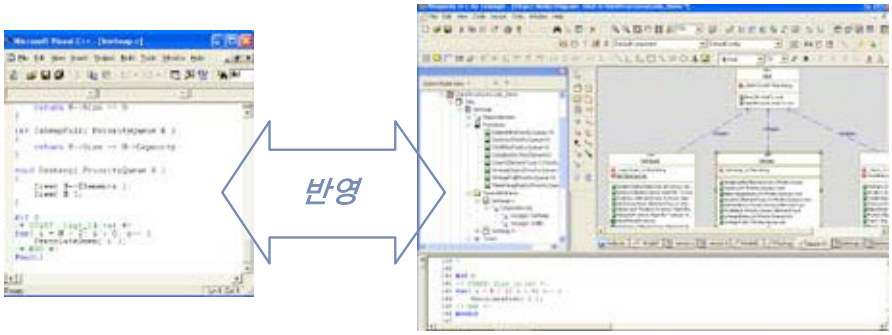

그림 7. 코드 동기화 코드 또는 모델을 업데이트하면 모두 동기화되어 코드 배치가 Rhapsody 애플리케이션 환경에 반영됩니다.

개발자는 모델링을 통해 시스템 아키텍처에 대한 전반적인 이해를 높일 수 있도록 코드를 시각화할 수 있습니다.

시각화

대부분의 경우 소프트웨어는 Rational Rose 애플리케이션에서 작성된 설계 사양을 준수하도록 작성되기 때문에 Rational Rose 사용자에게 소프트웨어 재사용은 특히 중요합니다. 때때로 새로운 시스템 구성요소를 완전히 새로 작성해야 할 경우도 있지만, 대부분의 경우 Rhapsody 소프트웨어는 검증된 지적 재산을 재사용함으로써 경쟁에서 앞설 수 있는 동력을 제공합니다.

주요 특징

종종 이러한 지적 재산은 곧 코드를 의미합니다. 임베디드 시스템에서 코드는 주로 C++, C 또는 Java™ 언어로 작성됩니다. 모델을 사용하면 코드를 시각화할 수 있으므로 아키텍처를 쉽게 이해하는 데 도움이 됩니다. 필요 시 지적 재산 캡처를 위해 코드를 모델로 분해하여 이를 진행 중인 설계의 일부로 활용할 수 있습니다. 모델은 또한 설계와 구현에 집중하고 문서화를 위한 시간을 절약할 수 있도록 설계 문서를 자동으로 작성하는 기능도 제공합니다.

Rational Rhapsody 소프트웨어의 구성요소 기반 설계 기능은 재사용 가능한 구성요소를 쉽게 작성할 수 있도록 하여 설계를 가속화하고 원활한 협업 환경을 지원합니다.

구성요소 기반 설계

구성요소 기반 설계는 모델링 방법을 사용하여 소프트웨어를 재사용 가능한 구성요소로 작성하는 플러그 앤 플레이 개발 환경을 작성합니다. Rational Rhapsody는 이러한 개발 방법을 활성화하기 위해 다음을 제공합니다.

- *UML 2 호환성*
- *클래스, 서브시스템 또는 기타 소프트웨어 단위와 같은 개별 구성요소 저장 기능*
- *다른 프로젝트로 구성요소 내보내기 또는 참조 기능*

이러한 기술을 결합하면 협업 환경이 개선되고 재사용이 활성화되어 설계 단계를 단축할 수 있습니다.

구성요소 기반 설계의 또다른 장점은 거의 모든 실시간 대상에 대해 코드 골격이 아닌 완전한 애플리케이션을 구현하는 기능입니다. 타사 또는 다른 그룹의 구성요소를 통합하는 빌드 프로세스에서 Rhapsody 소프트웨어는 빌드 스크립트로 쉽게 통합할 수 있는 풍부한 명령행 인터페이스를 제공합니다. 손쉽게 만들 수 없는 Make 파일을 자동으로 생성할 수 있으며 빌드 파일 생성 방법에 관한 정확한 사양을 지정할 수도 있습니다.

그림 8. Rhapsody 애플리케이션 아키텍처

그림 8은 Rhapsody 애플리케이션을 간략히 묘사한 것입니다. Rhapsody 애플리케이션은 UML 2 또는 시스템 모델링에 사용되는 다른 영역별 프로파일을 표현합니다. Rhapsody 소프트웨어를 사용하면 상태 및 활동 다이어그램 동작을 포함하여 코드 동작을 완벽하게 모델링할 수 있으며 그래픽 구성요소 다이어그램과 같은 시각화 도구를 사용하는 레거시 코드를 쉽게 통합할 수도 있습니다. 그런 다음 호스트 환경에서 Linux®, Wind River VxWorks, 16비트 운영 체제(OS)와 같은 실제 대상 시스템, 또는 OS가 아닌 대상 시스템으로도 신속하게 대상을 재지정할 수 있습니다.

주요 특징

Rational Rhapsody 애플리케이션은 기반이 되는 대상 운영 체제에 독립적으로 설계되었습니다. Rhapsody 애플리케이션은 기본 운영 체제의 서비스를 항상 사용하지만, Rhapsody 애플리케이션이 추가 서비스를 필요로 하지 않습니다. 이는 애플리케이션이 일반적으로 하나의 특정 OS에 대해 작성될 수 있도록 해주는 씬 추상화 계층(thin abstraction layer)을 통해 구현됩니다. Rational Rhapsody 환경의 메뉴를 사용하면 구체적인 운영 체제를 편리하게 선택할 수 있습니다. 이 메커니즘은 거의 모든 OS에 확대 적용이 가능합니다.

Rational Rhapsody는 MDD 방식을 활용하여 기업에서 높은 품질의 시스템을 가장 신속하고 효율적으로 개발할 수 있도록 지원합니다.

결론

Rational Rhapsody로 전환하여 MDD의 최신 기능을 현재 개발 환경에 도입하면 생산성과 시스템 성능을 개선하고 시장 출시 시간을 단축하는 동시에 비용을 절감할 수 있습니다. MDD 방식이 제공하는 기본적인 장점은 다음과 같습니다.

- *모델 기반 테스팅으로 개발 라이프사이클 초기의 오류 발생이 줄어듭니다.*
- *UML 2 기반의 영역별 모델링을 통해 조직 내 커뮤니케이션이 개선됩니다.*
- *문서화를 자동화 및 간소화할 수 있습니다.*
- *복잡한 시스템 통합 및 팀 협업 환경이 개선됩니다.*
- *코드 중심 워크플로우를 사용하는 현재 소프트웨어 개발 프로세스에서 기존 IP를 재사용합니다.*
- *거의 모든 플랫폼 및 기존 도구 인프라에서 플러그 앤 플레이 방식을 사용하여 애플리케이션 개발 시간이 단축됩니다.*

자세한 정보

IBM Rational Rhapsody에 대한 자세한 내용은 IBM 담당자 또는 IBM 비즈니스 파트너에게 문의하거나 다음 웹 사이트를 참조하십시오.

ibm.com/software/rational

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
June 2009

RAW14047-USEN-01