SIEMENS
Ingenuity for life

Siemens Digital Industries Software

차량용 임베디드 애플리케이션 개발 조정

애플리케이션 개발 및 품질 보증

개요

오늘날 가장 흥미롭고 혁신적인 자동차 기능은 매우 복잡하면서 정교한 임베디드 소프트웨어 애플리케이션을 통해 구현됩니다. 최신 차량에는 수 십 개 시스템을 제어하는 억 단위 코드 라인이 들어 있습니다. 자동차 소프트웨어 엔지니어는 최종 소비자에게 간단하고 직관적인 사용자 경험을 제공하기 위해 이러한 엄청난 복잡성을 관리해야 합니다. 엔지니어는 애플리케이션 코드를 개발하고 품질 보증 테스트를 실시하는 과정에서 설계 요구사항, 주요 시스템 속성 및 제약 조건, 나아가 개별 코드 라인에 대한 명확한 가시성을 제공하는 통합 플랫폼이 필요로 합니다.

Piyush Karkare
Siemens Digital Industries Software

소개

자동차 산업의 대대적인 트렌드에 의해 임베디드 소프트웨어 개발이 점차 차량 개발의 근간이 되어가고 있습니다. 차량 자동화, 연결성, 전기화 및 공유 모빌리티(ACES) 트렌드는 첨단 운전자 지원 시스템(ADAS), 배터리 관리, V2X (Vehicle-to-Everything) 통신 등과 같은 기능을 구현하기 위한 매우 정교한 소프트웨어의 필요성을 부각시킵니다 (그림 1).

자동차 제조사와 경영진은 이러한 트렌드에 발맞추기 위해 다른 주요 기술 중에서도 소프트웨어 개발을 자체화하겠다는 목표 아래 버티컬 통합을 추진하고 있습니다. 공급업체 또한 상당한 수준의 소프트웨어 개발이 수반되는 획기적인 차량 기능을 제공해야 하는 치열한 경쟁에 직면해 있습니다. OEM과 공급업체는 소프트웨어를 핵심 역량화하려고 시도하는 가운데 상당한 난관을 마주합니다. 최신 차량용으로 개발하는 소프트웨어는 특히 안전 및 운전자 지원 기능 자동화 관련 첨단 기술을 탑재해야 합니다. 또한 소프트웨어는 신뢰성과 기능 면에서 틀림이 없어야 합니다.

새로운 자동차 기능을 구현하려면 매우 복잡한 소프트웨어가 필요하며, 이는 전반적인 차량 내 소프트웨어 복잡성이 크게 증가시킵니다. 오늘날 차량 소프트웨어 콘텐츠는 보통 1억 5천만 개 이상의 코드 라인으로 이뤄져 있습니다. 소프트웨어 애플리케이션이 가진 각각의 복잡성은 이러한 소프트웨어 애플리케이션의 통신과 상호 작동이 필요한 차량 시스템 간의 상호작용으로 인해 더욱 복잡해집니다. 자동차 소프트웨어 엔지니어는 최종 소비자에게 간단하고 직관적인 사용자 경험을 제공하기 위해 이러한 엄청난 복잡성을 관리해야 합니다.

그림 1: 자율성, 연결성, 전기화 및 공유 모빌리티 트렌드는 차량용 소프트웨어에 대한 수요를 촉진합니다.

점점 더 까다로워지는 임베디드 애플리케이션 개발

소프트웨어 엔지니어는 소프트웨어 아키텍처로 변경 및 시스템 업데이트를 평가하고, 시스템 수준 정의를 통해 필요한 기능이 정상 작동하는지 확인하기 위해 소프트웨어 모델을 구성하면서 이미 개발 프로세스에 깊이 관여하고 있습니다. 많은 OEM 및 공급업체가 애플리케이션 개발을 위해 모델 기반 소프트웨어 개발 방식을 채택했습니다. 아키텍처 구축 및 모델링 프로세스는 코드를 작성하거나 수정하기 전에 소프트웨어의 기능을 확인하고 검증하기 위한 것입니다. 핵심 과제는 데이터 일관성을 유지하기 위해 수작업에 의존하지 않고 데이터 흐름 전반에 걸쳐 모든 엔지니어의 작업을 조화시키는 것입니다.

또한, 차량용 임베디드 소프트웨어 테스트 엔지니어는 완성된 소프트웨어 빌드 및 개발 프로세스에 모두에 영향을 미치는 엄격한 안전 규정을 준수해야 합니다. 관건은 엔지니어가 각 설계 의사 결정, 테스트 계획, 테스트 결과 및 개발 과정에서 생성된 관련 데이터를 검수할 수 있어야 한다는 점입니다. 이 디지털 스레드는 소프트웨어가 필수 규제를 준수한다는 증거로 필요합니다. 소프트웨어 개발의 복잡성과 속도를 감안할 때, 일정을 맞추거나 문제를 해결하려는 과정에서 이 정보를 놓치기 십상입니다.

차량 기능이 복잡한 소프트웨어 애플리케이션 사용이 늘어나면서 차량용 임베디드 소프트웨어 개발은 점점 더 어려워지고 있습니다. 임베디드 소프트웨어 팀은 계획부터 개발, 납품에 이르는 소프트웨어 개발 라이프사이클을 관리할 수단이 필요하며, 이는 제품 라이프사이클 관리 (PLM)와 그 외 필요한 엔지니어링 솔루션과 상호작용하며 통합 개발 플랫폼을 제공할 수 있어야 합니다 (그림 2). 애플리케이션 개발 및 품질 보증(QA) 과정에서 팀은 설계 요구사항과 주요 시스템 속성, 제약 조건, 나아가 개별 코드 라인에 대한 명확한 가시성을 확보해야 합니다. 이러한 플랫폼은 규제 준수를 보장하고, 관련 엔지니어링 도메인 내 기능 재사용 및 협업을 촉진할 수 있도록 완전한 추적성을 제공합니다.

주요 과제

애플리케이션 개발 과정에서 최대 과제는 소프트웨어 개발 프로세스와 제품 개발 프로세스 간 일관성 부재입니다. OEM은 보통 차량 마일스톤을 사용하여 시스템 수준 차량 기능과 변경, 업데이트를 추적합니다. 반면, 소프트웨어 개발은 빠르게 진행되는 AGILE 및 하이브리드 AGILE 흐름을 통해 이뤄집니다. 이렇게 개발 방식이 달리 진행을 방해하는 체크포인트 문제가 발생할 수 있습니다. 예를 들어, 소프트웨어 팀은 다음 마일스톤까지는 완료되지 않을 수 있는 시스템 수준 업데이트를 마냥 기다려야 할 수 있습니다. 마찬가지로 소프트웨어 팀은 다음 마일스톤에 맞춰 빌드를 제작해야 하는 압박감에 기한을 맞추느라 품질을 보장하지 못할 수 있습니다.

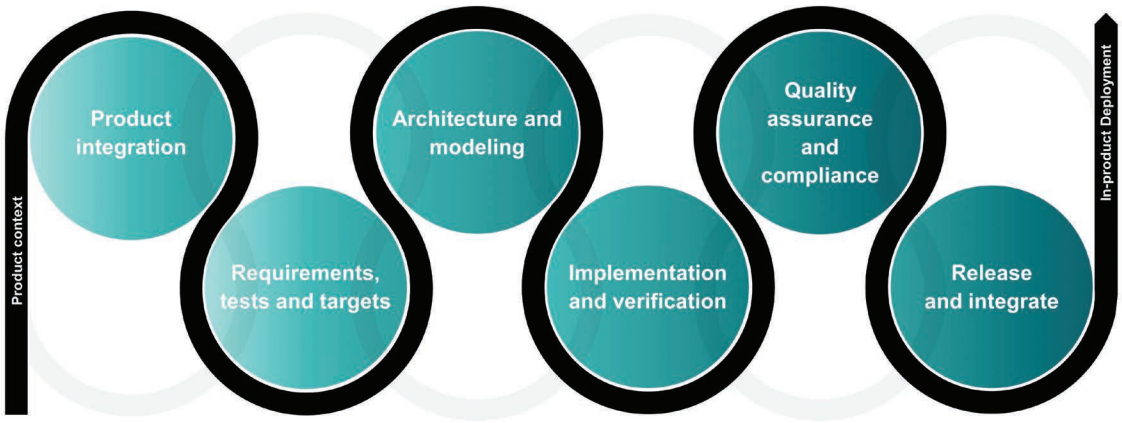

그림 2: 차량용 임베디드 소프트웨어 엔지니어는 애플리케이션 개발을 추적 및 관리하기 위한 통합 플랫폼을 필요로 합니다.

또한, 이렇게 상충된 개발 방식은 팀 간 정보 추적과 가시성을 어렵게 만듭니다. 테스트, 특히 HIL(Hardware-in-the-Loop) 또는 DIL(Driver-in-the-Loop) 테스트 중에는 소프트웨어 팀이 현재 어떤 테스트를 진행하는 지, 나아가 그 테스트를 진행하는 이유까지 파악하는 것이 중요합니다. 여기에는 어떤 데이터 아티팩트가 어떻게 변경됐는지, 어떤 소프트웨어 빌드를 사용할 준비가 됐는지, 소프트웨어 빌드를 확인하고 검증하는 데 필요한 하드웨어 추상화 수준은 어느 정도인지를 파악하는 것이 포함됩니다. 한편 구현 팀은 테스트에서 발견된 문제를 적시에 효율적으로 업데이트하고 해결할 수 있도록 테스트 결과를 확인할 수 있어야 합니다.

자동화된 코드 생성은 소프트웨어 모델링과 코딩을 통합해 팀의 민첩성을 높이는 데 도움이 될 수 있지만, 체계성/유기성이 부족한 데이터 관리 솔루션은 통합 및 조정 중에 위험을 증가시켜 시간 절감 효과를 상쇄합니다. 엔지니어는 최신 하드웨어 및 시스템 수준 제약 조건과의 일관성을 유지하기 위해 코드 구현이 필요합니다. 이를 위해 엔지니어는 테스트 결과를 반영해 애플리케이션 빌드를 실시간으로 업데이트할 수 있도록 해주는 감사 가능한 변경 관리 프로세스를 필요로 합니다.

제품 및 소프트웨어 엔지니어링을 아우르는 감사 가능한 변경 관리 프로세스는 컴플라이언스를 보장하는 데 매우 중요합니다. 첫째, 소프트웨어 팀이 소프트웨어 기능에 영향을 미치는 새롭거나 모호한 규정에 신속하게 대응할 수 있도록 지원해야 합니다. 둘째, 변경 관리는 OEM이 규제 준수를 입증하는 데 필요한 증거를 보유할 수 있어야 합니다.

유연한 구현 및 검증 프로세스를 구축하는 것은 코드를 작성하는 엔지니어만의 책임이 아닙니다. 소프트웨어 테스트와 테스트 범위는 요구사항에 따라 정해져야 하며, 개발 주기 초기에 시작돼야 합니다. 결함은 발견이 늦을수록 비용 부담이 커지기 때문에, 구현 과정에서 결함을 조기에 감지하는 것이 상당히 유리합니다. 설계 요구사항, 아키텍처 제약 조건 및 시스템 작동에 대한 컨셉트를 최적화하고 검증할 수 있으려면 애플리케이션 설계를 초기에 모델링하고 시뮬레이션 및 실행해야 합니다. 소프트웨어 엔지니어는 시스템 맥락에서 초기 하드웨어-소프트웨어 연동 해석을 실시해 개발 프로세스 초기에 문제와 결함을 발견할 수 있습니다.

테스트를 완료한 후 권장 변경 사항을 구현하려면 시간이 오래 걸릴 수 있습니다. 구현 팀이 복잡한 테스트 결과를 분류해가면서 수정할 사항을 결정해야 하는 경우 특히 그렇습니다. 요구사항 및 시스템 제약 조건에 대한 정확한 문서와 추적성을 관련 팀과 관계자에게 제공하면 빠르고 정확한 솔루션이 가능해집니다.

마지막으로, 시스템 수준 변경에서 개별 코드 라인에까지 이르는 포괄적인 추적성은 종종 지속적인 변경 및 업데이트로 인해 약화됩니다. 개발 프로세스 전반에 걸쳐 명확한 가시성을 유지하는 것은 소프트웨어 콘텐츠 및 여러 기능 간의 종속성이 증가함에 따라 점점 더 복잡해집니다. 이 문제는 엔지니어가 추적성을 유지하기 위해 통합되지 않은 여러 툴을 탐색해야 하는 경우 더욱 증폭될 뿐입니다. 스프레드시트, 데이터베이스 및 자체 개발 연결에 의존하는 솔루션은 복잡하며, 조직 성장에 맞춰 확장되지도 않습니다.

통합 플랫폼을 이용한 임베디드 애플리케이션 개발 및 품질 보증

애플리케이션 개발 및 품질 보증 프로세스에 수반되는 복잡한 동시 작업을 추적하고 관리할 수 있는 솔루션이 현재 출시돼 있습니다. 이러한 솔루션들은 임베디드 애플리케이션 개발에 필요한 포괄적인 추적성과 가시성을 제공합니다. 개발 조정 솔루션이 있는 임베디드 애플리케이션 개발 과정을 살펴보겠습니다 (그림 3).

기능적 설계 및 모델링 – 코딩을 위해 모델 및 테스트 완료 소프트웨어 구성요소 활용

소프트웨어 팀은 강력한 소프트웨어 구성요소 아키텍처, 필요한 작동의 진정한 기능 트윈을 표현하는 소프트웨어 모델, 그리고 코딩이 완료되기 전에 소프트웨어 애플리케이션 요소를 정의하고 검증하는 데 필요한 통신의 완전성을 보장하는 모델 상호작용을 생성합니다. 소프트웨어 아키텍트와 엔지니어는 일반적으로 MATLAB 또는 SIMULINK와 같은 전용 툴을 사용해 이 작업을 완료합니다. 최신 소프트웨어 개발 조정 솔루션은 이러한 툴을 확장해 아키텍처, 모델, 요구사항, 테스트 방법, 손실 분석, 안전 규정 준수 등을 직접 연결합니다. 결과물이 어느 정도 완성도를 갖추면 소프트웨어 개발 솔루션은 애플리케이션 요구사항 및 제약 조건과 관련해된 모델과 아키텍처, 구성요소 상호 작용의 진행 상황과 일관성을 추적합니다. 이는 구성요소 간의 모든 상호 작용을 정확하게 나타내는 강력한 아키텍처와 관련된 모델 표현을 활용해 모든 요구사항과 사양, 작동도 포괄합니다.

소프트웨어 엔지니어는 애플리케이션 개발용 통합 플랫폼을 활용해 기능 아키텍처 간 일관성을 보장할 수 있으며, 여러 팀이 정확하고 일관된 정보에 기반해 아키텍처 관련 절충안을 도출할 수 있도록 지원합니다. 이러한 솔루션은 모델의 애플리케이션 요구사항 충족 여부를 자동으로 추적하고, 모델과 관련 요구사항 간 추적성도 제공합니다. 결과적으로 엔지니어는 모드 기반 소프트웨어 개발을 보다 효과적으로 활용해 문제를 조기에 파악하고 다운스트림 프로세스 중에 일관성을 유지할 수 있습니다 (그림 4).

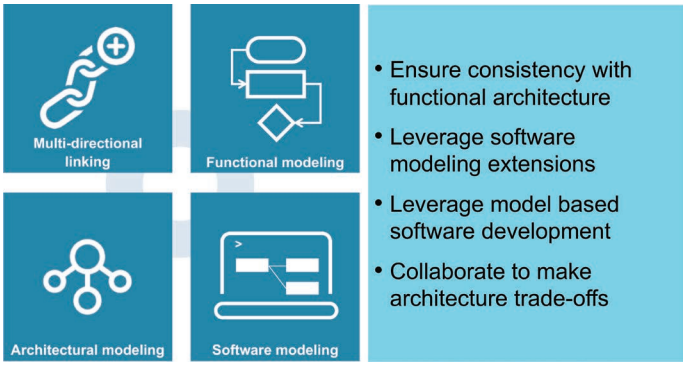

그림 4: 통합 애플리케이션 엔지니어링 플랫폼은 요구사항에서 소프트웨어 아키텍처, 모델 등에 이르는 추적성을 보장해 아키텍처 및 모델링 프로세스를 지원합니다.

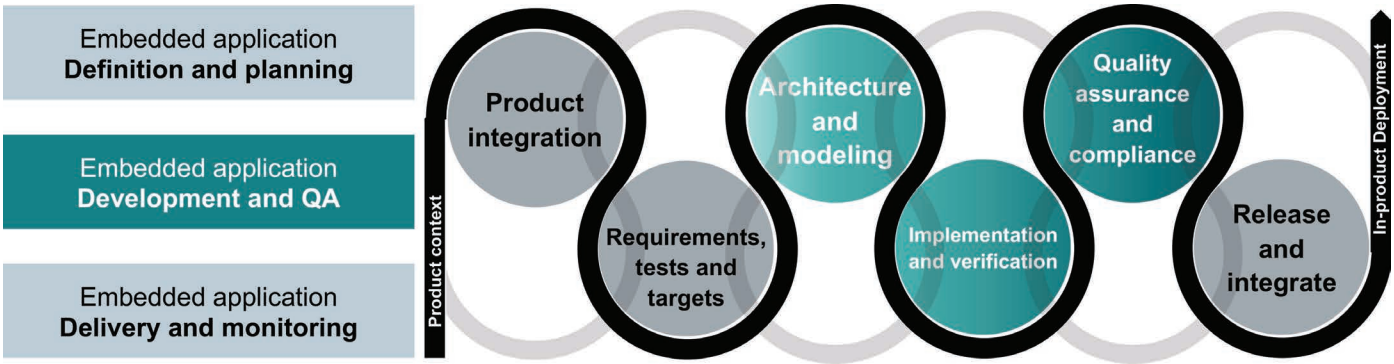

그림 3: 전체 애플리케이션 개발 흐름에서의 애플리케이션 개발 및 품질 보증.

소프트웨어 모델링은 엔지니어가 실제 코딩에 상당한 시간을 투입하기 전에 시스템 요구사항 충족 여부를 확인합니다. 애플리케이션 개발 플랫폼은 프로젝트별로 결정할 수 있는 맞춤형 조건에 기반해 팀원에게 코딩 작업을 자동으로 할당하도록 구성할 수 있습니다. 엔지니어가 이 작업을 완료하면 애플리케이션 개발 플랫폼은 선택한 소프트웨어 코드 관리 (SCM) 솔루션 및 통합 개발 환경 (IDE)에 대한 확장으로 진행 상황을 모니터링 할 수 있습니다. 또한, 애플리케이션 개발 플랫폼은 다양한 코드 성능(정적 및 동적) 및 코드 커버리지 툴과의 통합을 통해 비기능적 요구사항에 대한 컴플라이언스도 추적할 수도 있습니다.

긴밀하게 조정된, 일관된 소프트웨어 모델링 방식은 프로세스 진행을 단축시킬 뿐만 아니라 SOTIF (Safety Of The Intended Functionality) 등의 기술을 탑재해 소프트웨어가 의도한 대로 작동하고 설계 단계에서 미리 위험을 방지할 수 있습니다. SOTIF 방식을 통합하면 오류가 발생할 것으로 가정하는 안전 목표를 적용해 위험을 완화하는 보편적인 기능적 안전 방식을 보완합니다. 이 조합으로 매우 강력한 차량용 임베디드 소프트웨어 애플리케이션이 구현됩니다.

코드 구현 및 통합

엔지니어는 애플리케이션 엔지니어링 플랫폼을 활용해 모델링 및 코드 구현 작업을 통합할 수 있으며, 이를 통해 모델 생성과 코딩 간 중복 작업을 제거할 수 있습니다. 요구사항에서 모델, 코드, 테스트까지 바로 연결해 구현 과정에서 검증이 바로 이뤄질 수 있습니다. 또한 이러한 연결로 시스템 수준 정의의 제약 조건 및 변경 사항, 하드웨어 사양 등을 항상 최신 상태로 구현에 적용할 수 있습니다. 애플리케이션 개발 플랫폼은 감사 가능한 변경 제어 프로세스를 통해 모델과 소프트웨어, 하드웨어, 차량 내부 테스트 결과를 반영해 코드를 유연하게 업데이트할 수 있도록 지원합니다 (그림 5).

반복적 계획 툴은 임베디드 애플리케이션 개발 팀이 시간을 효율적으로 계획하고 작업 완료 시기를 추적해 개발이 일정대로 진행되도록 지원합니다. 특정 조치나 프로세스 단계를 수행하는 데 전자 서명을 요구하도록 플랫폼을 구성하면 주요 관계자가 결정을 검토하고 승인할 수 있습니다. 또한, 임베디드 애플리케이션 개발 플랫폼은 변경 분석 및 구현에 대한 책임을 추적할 수 있어 팀은 수정된 모든 데이터 아티팩트와 각 변경의 근거를 완벽하게 추적해 계획된 업데이트에 대해 소프트웨어 릴리스나 빌드가 원활히 지원되는지 확인할 수 있습니다.

이 계획 솔루션은 경험적 데이터에 기반해 작업 완료에 관한 예측 정보를 제공하는 팀 속도 관리 기능도 제공합니다. 시간, 비용 및 스토리 포인트를 비롯한 다양한 예측 방법이 지원됩니다. 팀 리더나 애플리케이션 소유자는 완료된 워크플로 조건을 정의할 수 있습니다. 예를 들어, 문서가 제출된 경우에만 워크플로가 완료로 표시되게 설정할 수 있습니다. 사용자는 기능 및 품질 일관성을 보장하기 위해 구성요소와 설계 요구사항 사양과 같은 지원 문서를 첨부할 수도 있습니다.

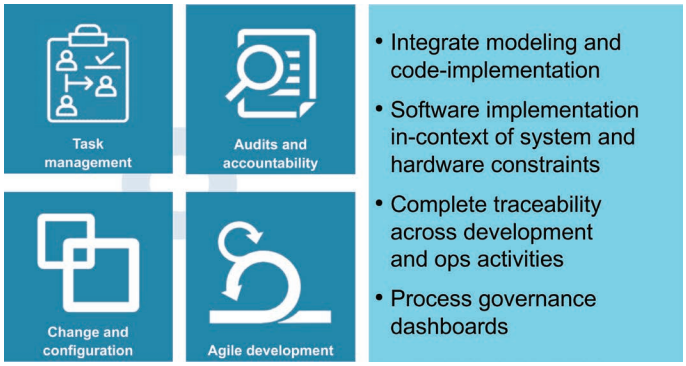

그림 5: 애플리케이션 개발 조정을 위한 통합 플랫폼은 Agile 개발 방법을 지원하는 동시에 작업 관리, 책임성 및 변경 관리 효율을 높여줍니다.

품질 보증

품질 보증은 보통 초기에 시작되어 요구사항, 아키텍처, 모델 및 코드 구현 개발이 진행되고 점차 완성되어가는 과정에서 이뤄지는 연속 프로세스입니다. 품질 보증은 MiL (Model-in-the-Loop), SiL (Software-in-the-Loop), HiL (Hardware-in-the-Loop) 및 ViL (Vehicle-in-the-Loop) 등의 여러 추상화 수준에서 이뤄지는 테스트의 총합입니다. 여기에는 성능 시험장에서 진행하는 가상 HiL 및 물리적 테스트가 모두 포함됩니다. 이 반복적 테스트 흐름은 각 개발 단계에서, 그리고 설계 변경이 구현될 때 소프트웨어 품질을 평가하기 위해 조정될 수 있습니다. 이러한 유형의 테스트 체계는 엔지니어가 개발 프로세스 초기에 필요에 맞게 소프트웨어 모델을 조작하고 검증할 수 있도록 지원해 모델 기반 소프트웨어 설계 흐름을 촉진합니다.

하지만 엔지니어는 소프트웨어 아키텍처, 모델 및 실제 코드를 개발하고 테스트하는 과정에서 추적이 필요하고 해당 소프트웨어 빌드 또는 구성에 연결되어야 하는 새로운 데이터를 계속해서 생성합니다. 각 테스트 라운드마다 소프트웨어 엔지니어는 테스트 케이스, 테스트 실행 전략, 테스트 벡터 및 기타 관련된 데이터

아티팩트를 상세하게 기술하는 테스트 계획을 작성하거나 업데이트해야 합니다. 테스트가 완료되면 테스트 결과를 다시 해당 테스트에 연결하고, 문제를 할당 및 해결해야 합니다. 이러한 데이터 연결은 테스트 커버리지를 보여주며, 요구사항에서 구현에 이르는 완전한 추적성을 지원하는 데 매우 중요합니다.

테스트 계획, 테스트 케이스, 테스트 결과, 관련 데이터 아티팩트, 식별된 문제 및 위험 상태를 망라하는 테스트 가치 사슬을 구성하고 추적하면 전체 소프트웨어 개발을 조율하는 과제와 그 비중이 강화됩니다. 추적하고 관리해야 하는 복잡한 상호 관계의 데이터가 많습니다. 엔지니어가 데이터를 추적하고 관리해야 하는 부담을 느끼면 소프트웨어 엔지니어링 및 개발에 집중할 수 없습니다. 또한, 소프트웨어 팀은 모든 관련 데이터와 함께 최종 빌드에 맞춰 애플리케이션 제공을 조정해야 합니다. 애플리케이션 개발 및 조정 플랫폼의 핵심 기능은 이러한 복잡성을 관리해 제공되는 애플리케이션 바이너리 및 관련 데이터의 품질을 보장하는 데 도움이 될 수 있습니다 (그림 6).

Polarion과 같은 애플리케이션 개발 및 조정 플랫폼은 통합 테스트 관리 툴을 갖추고 있습니다. 이러한 도구를 사용하여 엔지니어는 테스트 케이스를 생성하고 해당 작업 항목에 연결할 수 있습니다. 작업 항목에는 애플리케이션 요구사항, 변경 요청, 다른 테스트 케이스 등이 포함될 수 있습니다. 테스트 관리 툴을 사용하면 각 테스트 케이스 단계를 수정할 수 있으며, 테스트 사양과 테스트 항목 구성을 분리해 단일 절차로 더 큰 테스트 커버리지를 구현할 수 있습니다. 또한, 이러한 솔루션은 외부 테스트 자동화 툴과 통합해 테스트를 자동으로 실행하고 결과를 가져올 수도 있습니다.

이 워크플로는 쉽고 원활한 프로세스 규제 컴플라이언스를 지원합니다. 회사 프로세스 및 우수 사례 정보를 프로젝트 보기에서 바로 사용할 수 있어 프로세스 상세 사항을 다 파악하지 않더라도 모든 팀원이 정해진 프로세스를 준수할 수 있습니다. 또한, Polarion의 워크플로는 요구사항, 작업, 변경 요청, 프로세스 관리, 프로젝트 계획, 시간 관리, 빌드 관리, 소스 코드 감사 및 메트릭, 성숙도 모델 규제 준수 추적을 통합해 고품질의 적시 애플리케이션 개발을 지원합니다.

고급 애플리케이션 개발 및 조정 플랫폼은 결함 및 위험 관리 기능도 갖추고 있습니다. 이러한 기능을 통해 테스트 실패 시 문제 및 위험 보고서를 자동으로 생성하고, 소프트웨어 엔지니어에게 작업을 할당해 솔루션 구현 시간을 단축할 수 있습니다. 임베디드 애플리케이션 개발 플랫폼은 자동으로 문제를 추적하고 구현된 수정 및 추가 테스트를 비롯한 모든 관련 후속 작업을 기록합니다. 추가 테스트 수행 시에는 전체 품질을 개선하고 차량 프로그램에 소프트웨어 도입을 용이하게 하기 위해 하드웨어와 소프트웨어를 함께 시뮬레이션하는 것이 매우 중요합니다. 임베디드 애플리케이션 개발 조정을 위한 통합 플랫폼은 에뮬레이트된 하드웨어와 물리적 하드웨어를 모두 사용해 관련된 모든 하드웨어 구성에서 각 소프트웨어 빌드를 테스트할 수 있도록 지원합니다. 테스트가 수동/자동 방식으로 실행됐는지, 타사 툴에서 수행됐는지, 애플리케이션 개발 플랫폼 환경 내에서 수행됐는 등을 비롯한 모든 테스트 정보가 추적됩니다. 이를 통해 해결 중이거나 해결 완료된 모든 문제에 대한 완벽한 보고서가 개발 과정 전체에 걸쳐 유지 및 관리됩니다.

이러한 기능은 차량용 임베디드 소프트웨어 엔지니어에 몇 가지 주요 이점을 제공합니다. 설계 프로토타입 생성, 시뮬레이션 및 실행을 통해 설계가 요구사항을 충족하는지 확인하고 아키텍처를 최적화하며 제대로 작동하는지 확인 및 검증할 수 있습니다. 또한, 하드웨어-소프트웨어 상호 시뮬레이션을 수행해 시스템 하드웨어 및 주변 장치 컨텍스트에서 설계를 평가할 수 있습니다. 이를 통해 엔지니어는 소프트웨어 또는 하드웨어 매핑에서 시스템 컨텍스트 상 발생하는 문제나 위험을 발견할 수 있습니다. 애플리케이션 개발 플랫폼 솔루션은 저변에서 요구사항, 모델, 아키텍처, 테스트 계획, 테스트 결과 등을 추적해 추적성을 보장하며, 소프트웨어 엔지니어에 적시에 올바른 정보를 제공합니다. 이를 통해 관계자에 정확하고 완전한 문서를 쉽게 제공할 수 있어 문제 해결이 간소화됩니다. 이렇듯 포괄적인 추적성은 중요한 버그가 현장에 도달하는 것을 방지하는 데에도 도움이 됩니다.

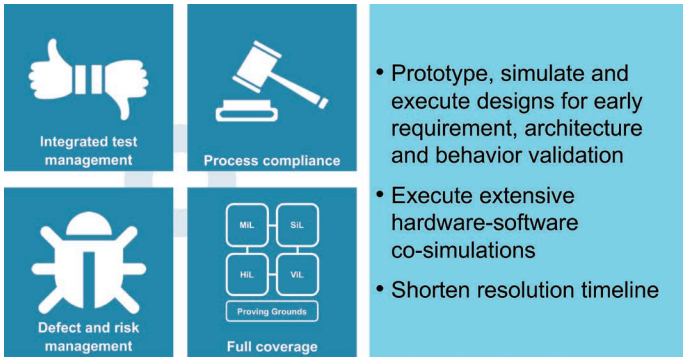

그림 6: 테스트 관리, 프로세스 규제 준수, 위험 관리 툴은 전체 테스트 커버리지를 보장하는 데 도움이 됩니다.

결론

차량용 임베디드 소프트웨어 애플리케이션 개발 및 품질 보증은 구현 하드웨어 (센서 및 액추에이터와 같은 주변 장치 포함)와 시스템 고려사항, 제약 조건 (차량 통합 및 기하학적 종속성 포함)을 지속적으로 고려해야 하는, 복잡하며 얽힌 요소가 많은 프로세스입니다. 통합 소프트웨어 엔지니어링 플랫폼은 계속되는 변경에도 데이터 일관성을 보장하는 엔지니어링 환경을 제공해 개발 프로세스의 모든 관계자가 차량 가변성의 전체 범위와 호환되는 고품질 소프트웨어 애플리케이션 제공에 지속적으로 참여할 수 있도록 지원합니다. 이러한 플랫폼은 애플리케이션 개발에 사용되는 다양한 툴 및 강력한 변경 관리 기능과의 강력한 통합을 통해 이러한 데이터 일관성을 구현합니다.

이러한 기능은 디지털화 되어가는 자동차 시장에서 경쟁하는 데 필요한, 점점 더 복잡해지는 소프트웨어 기능을 제공하는 데 결정적인 역할을 합니다. ADAS, 인포테인먼트 및 광범위한 전기, 전자 및 소프트웨어 기반 기능이 소비자를 대상으로 한 가치를 창출하고 있어 차량의 평균 코드 라인도 증가할 수밖에 없습니다. 현재 고급 차량에는 보통 1억 5천만 개의 코드 라인이 들어갑니다.

미래에는 자율 주행, 차량 연결성, 전기 파워트레인 및 공유 모빌리티 (ACES) 개발이 이러한 복잡성을 한층 높이고 소프트웨어를 자동차 가치 사슬의 최상위에 올려놓을 것입니다. 자율 주행 차량은 작동하는 데 수십억 개의 코드 라인을 소요할 것으로 예상됩니다. 혁신적이고 강력한 차량용 소프트웨어 개발이 자동차 및 모빌리티 기업의 핵심 차별화 요소가 되어가는 가운데, 이러한 복잡성은 임베디드 애플리케이션 개발 조정의 중요성을 한층 증가시킬 것입니다.

Siemens 솔루션이 귀사에 어떤 도움을 줄 수 있는지 자세히 알아보려면 siemens.com/aes에서 블로그, 백서, 팟캐스트, 제품 동영상, 웨비나, 솔루션 기능, 인포그래픽 등을 확인해 보십시오.

Siemens Digital Industries Software

본사
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 972 987 3000

미주 지역
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 314 264 8499

유럽 지역
Stephenson House
Sir William Siemens Square
Frimley, Camberley
Surrey, GU16 8QD
+44 (0) 1276 413200

아태 지역
Unit 901-902, 9/F
Tower B, Manulife Financial Centre
223-231 Wai Yip Street, Kwun Tong
Kowloon, Hong Kong
+852 2230 3333

Siemens Digital Industries Software 소개

Siemens Digital Industries Software는 엔지니어링, 제조 및 전자 설계가 미래와 만나는 디지털 엔터프라이즈를 실현하기 위한 혁신에 박차를 가하고 있습니다. Siemens Digital Industries Software의 솔루션은 규모를 막론한 기업이 조직에 새로운 인사이트와 기회, 혁신을 촉진할 자동화 수준을 제공하는 포괄적 디지털 트윈을 생성할 수 있도록 지원합니다. Siemens Digital Industries Software 제품과 서비스에 대한 자세한 사항은 sw.siemens.com 를 방문하시거나 LinkedIn, Twitter, Facebook 및 Instagram 계정 팔로우를 통해 확인하실 수 있습니다. Siemens Digital Industries Software – Where today meets tomorrow.

siemens.com/software

81256-82844-C1-KO 11/20 LOC