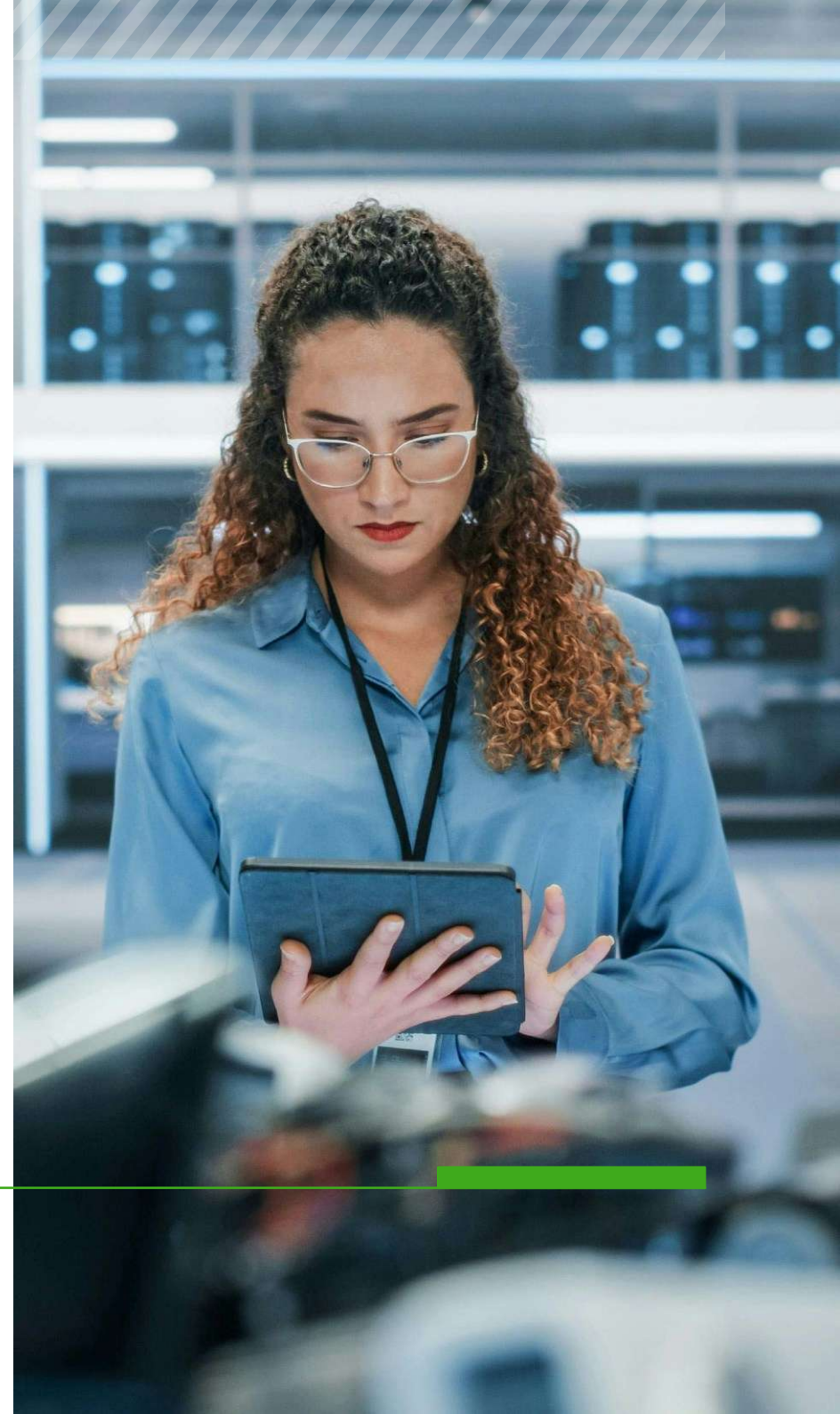




DIGITAL TRANSFORMS PHYSICAL



디지털 제품을 위한 하드웨어 및 소프트웨어 개발 통합



시작하는 말

오늘날의 세계에서 소프트웨어가 갖는 중요성은 점점 더 커지고 있습니다. 그리고 우리가 일상생활과 제품에 소프트웨어를 점점 더 많이 포함시킬수록 소프트웨어 자체도 점점 더 복잡해집니다.

이해하기 쉽게 설명하자면, 1969년 아폴로 11호의 달 착륙에는 약 145,000줄의 코드가 필요했습니다. 하지만 오늘날에는 자동차의 실제 주행까지 최대 1억 줄의 코드가 필요합니다.

제품에 포함된 소프트웨어가 많아질수록 개발이 더 복잡해지고, 오류가 발생할 여지도 많아집니다. 한편, 혁신적이면서 복잡한 고품질 제품을 더 빨리 시장에 출시시켜야 한다는 압박은 그 어느 때보다 큼니다. 여기에 하드웨어 및 소프트웨어의 개발 프로세스와 서비스 혁신 추진 프로세스를 병렬로 효율적으로 관리하여 투명성을 보장하면서도 모든 것을 단일 제품에 통합해야 하는 과제까지 추가됩니다.

많은 제조 기업은 복잡한 제품 개발을 최적화하기 위한 경쟁에서 뒤처지지 않으려고 노력하고 있지만, 프로세스, 시스템 및 팀의 사고방식을 크게 개선하지 못한다면 처음부터 복잡한 제품을 염두에 두고 비즈니스를 설계한 경쟁업체에게 밀려날 수밖에 없습니다.

제조업체의 발전에 핵심이 되는 요소는 다음과 같습니다.

전체 시스템을
체계적인 시각으로
바라보기

시스템 엔지니어링
방법론 활용하기

조직 내 여러 부서 간의
의사소통 증진시키기

올바른 PLM/ALM 도구
사용하기

소프트웨어로 여는 제조업의 새로운 세상

헨리 포드가 이동 조립 라인을 도입하여 자동차 산업을 혁신한 지 100년이 넘는 지금, 제조업은 다시 한번 급격한 변화를 맞이하고 있습니다.

하지만 단일 부분에서의 혁신이었던 예전과 달리, 지금의 변화는 진화하는 비즈니스 모델, 기술 혁신 및 공급망 변화의 조합으로 인해 발생하는 것입니다.

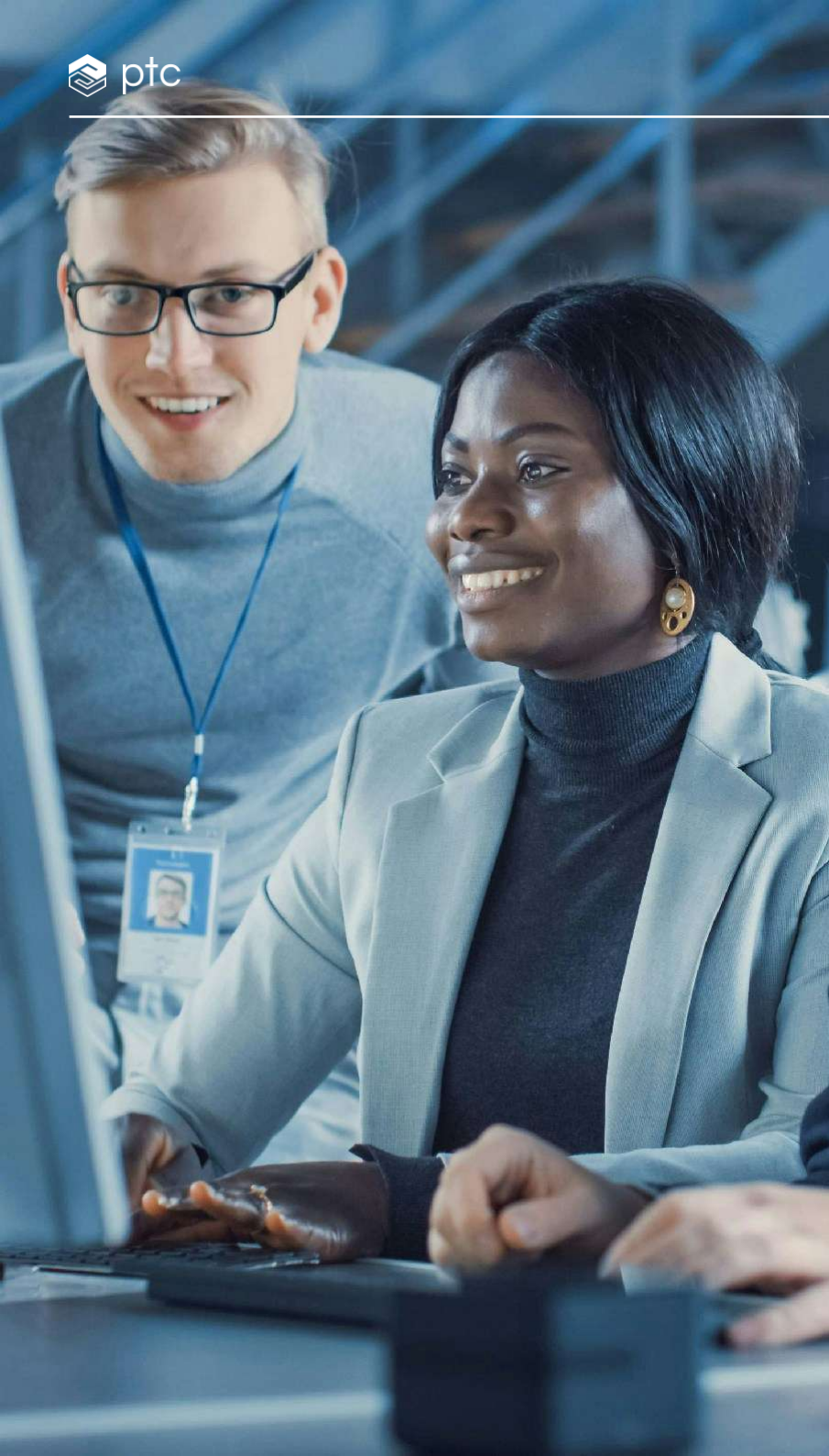
이런 변화의 원동력과 촉진제는 소프트웨어입니다. 제품의 기능은 더 이상 전기 기계적 특성에만 의존하지 않으며, 자동차, 의료 기술, 기계 및 플랜트 엔지니어링, 심지어 자전거에서도 소프트웨어와 하드웨어의 공생이 점점 더 긴밀해지고 있습니다.

비즈니스 관점에서 이러한 발전은 기회이자 도전입니다. 경쟁이 증가하고 혁신에 대한 압박이 커지는 상황에서 소프트웨어 기반 제품을 위해 개발 시간을 단축하고 완전히 새로운 비즈니스 모델을 개척할 수 있게 된다면 새로운 기회를 얻을 수 있기 때문입니다.

전기 자전거를 예로 들어보겠습니다. 이미 센서를 사용해 페달 보조를 제어하고, 디스플레이는 블루투스를 통해 휴대전화와 연결되어 있습니다. 결과적으로, 현재 상태에서 조금만 더 나아가면 클라우드 기반 성능 측정 시스템을 통해 훈련 과정을 추적하고 개인 맞춤형 트레이닝 지침을 제공할 수 있게 됩니다. 또는 언덕이 너무 가파르면 편리하게 추가 파워를 제공하는 부스트(boost) 모드를 휴대전화에서 구매할 수도 있습니다.

그렇다면 이런 미래로 가기 위해 해결해야 하는 과제는 무엇일까요?





학제적 팀(Interdisciplinary teams)

제품 개발자의 관점에서 이러한 변화는 결코 간단하지 않습니다. 기계적 측면에서 변동성이 감소하더라도 소프트웨어와 이들 구성 요소 간의 긴밀한 상호관계로 인해 전체 시스템의 복잡성이 증가합니다. 게다가 하나의 소프트웨어에도 서로 다른 여러 개발 팀(또는 서로 다른 사고방식을 가진 학제적 그룹)이 있으며, 이들은 각자의 방식과 속도로 혁신주기를 관리합니다. 이들의 작업을 조율하고 평행선상의 여러 개발 스트림을 통합하는 것은 큰 도전 과제입니다.

시스템 검증

제조기업은, 특히 안전이 중요한 제품의 경우, 전체 시스템을 어떻게 검증할 수 있을지도 고려해야 합니다. 산업 및 제품에 따라, 표준을 준수하려면 모든 개발 단계와 변경 사항을 요구사항에 대해 추적할 수 있어야 합니다.

따라서 미래에는 전체 제품의 복잡성이 증가하더라도 그 복잡성의 큰 요인이 전기 기계적 측면에서 소프트웨어로 이동할 것입니다. 그렇다고 해서 상황이 더 쉬워지는 것은 아니지만, 물리적 세계가 부과하는 제약을 제거함으로써 복잡성을 제어할 다른 방법을 적용할 수 있기 때문에 비례적으로 더 높은 수준의 효율성과 확장성을 부여할 수 있습니다.

이는 성과 향상의 정도를 훨씬 더 큰 수준으로 높이려면 복잡성 제어에 필요한 노력보다 더 많은 노력을 기울여야 한다는 의미입니다.

물론 말처럼 쉽지는 않습니다. 그래서 지금부터 이러한 변경 관리를 지원하는 도구의 종류를 살펴보겠습니다.

소프트웨어 기반 제품의 라이프사이클 관리를 위한 도구

기계 공학, 전자 공학, 그리고 점점 더 복잡해지는 소프트웨어 간의 기능적 공생은 컴퓨팅 파워의 강력한 성장 덕분에 가능해졌습니다.

그러나 여기에도 단점은 있습니다. 많은 요소가 상호작용할 때마다 통계적으로 오류가 발생할 확률이 증가하는 것입니다.

비교를 위해 예를 들자면, 1994년의 리눅스 커널은 약 20만 줄의 코드가 필요했습니다. 2018년까지 그 수치는 2,500만 줄 이상으로 증가했습니다. 상황을 더 악화시키는 것은 구조가 부족하여 문제 해결의 효율성이 감소한다는 점입니다. 시간이 지나면서 소프트웨어 개발에 사용되는 도구 키트는 오늘날 우리가 알고 있는 애플리케이션 수명 주기 관리(ALM)로 발전했으며, 그 프로세스 모델은 현대의 시스템 엔지니어링에 통합되었습니다.

아직까지도 많은 기업이 여러 개별 컴포넌트를 관리하는 데 초점을 맞추고 있습니다. 하지만 요구 사항 명세서에서 최종 제품에 이르기까지 일관성을 보장하는 것은 기업 전반에 걸쳐 지속적으로 확립해 온 제품 수명 주기 관리(PLM) 개념(방법 및 도구)로는 일반적으로 지원할 수 없는 부분입니다.

전기 기계적 요소가 지배하는 세계에서 주요 과제는 다음과 같습니다.

- 설계 프로세스 지원의 필요성
- 다양한 컴포넌트에 대한 기술적 데이터 관리
- 하나의 팀으로써 효율적 작업을 수행할 수 있는 역량



소프트웨어 기반 제품 개발에 대한 일반적 접근방법

주로 기계 및 전기 기계 제품에 중점을 두고 성장한 기업들의 경우, 통합하는 소프트웨어가 점점 많아질 때 다음과 같은 경향을 보입니다.



소프트웨어를 하드웨어 부속물로 취급

이는 소프트웨어를 하드웨어의 확장 또는 추가 기능으로 간주할 때 발생합니다. ("소프트웨어는 또 다른 부품 번호일 뿐"이라는 맥락에서 생각하는 경우)

소프트웨어 컴포넌트는 전기 기계적 컴포넌트와 동일시되며 부품 번호가 할당됩니다.

제품 구조에서 최소한 ECU를 식별할 수 있을지라도, 평면 BOM에서 모든 상호 종속성을 식별하는 것은 궁극적으로 불가능할 것입니다.



ALM과 PLM의 공존

어떤 경우에는 PLM 세계와 ALM 세계가 평행선상에서 독립적으로 설정되어 있습니다. 이런 상태는 "저쪽에서 무엇을 하고 있는지 모르겠지만, 관심도 없다"라고 설명하는 것이 가장 적절할 것입니다.

소프트웨어 개발자는 전기 기계적 개발의 제약에서 벗어나 이제 ALM 세계에서 동적 역량을 완전히 아우를 수 있습니다. 소프트웨어는 짧은 이터레이션과 매우 애자일한 방식으로 현재 고객 요구 사항을 충족하도록 최적화 됩니다.

그러나 두 세계가 완전히 분리되어 있는 시나리오에서 간과하기 쉬운 것은 PLM과 ALM 세계 간의 상호 동기화입니다. 안타깝게도 이런 동기화의 부재로 인해 발생하는 불일치는 생산에서 완제품까지 영향을 미치게 됩니다.

두 가지 접근방법의 장단점

위의 두 시나리오 모두 이상적이지는 않습니다.

소프트웨어를 하드웨어의 "애드온"으로 취급하는 것은 기능 통합 수준이 특히 높지 않거나 제품 변경 빈도가 높지 않은 경우라면 가능합니다. 즉, 소프트웨어가 명확하게 정의된 특정 국지적 작업을 해결하기 위한 것이라면 전체 시스템에 미치는 영향은 크지 않을 것입니다. 게다가 소프트웨어 개발의 민첩성에 대한 큰 기대는 없어야 합니다.

두 번째 접근 방식은 소프트웨어 관점에서는 가능할 수 있지만, 제품과 그 가치 창출 프로세스에 대한 전체적인 시각을 수립하는 데는 적합하지 않습니다. 이 경우 다양한 도구와 부서 간의 효율적인 협업과 검증을 보장하는 것이 여전히 문제로 남게 됩니다.

제품 변형의 복잡성과 기업의 상대적 규모에 따라 두 가지 전략 모두 잠시 동안은 작동할 수 있습니다. 직원들이 동기를 갖고 방법론적 및 절차적 결함을 보완하는 경우도 자주 있고, 기업이 시스템 문제에 대한 관용의 정도를 놀라운 수준으로 끌어올리기도 합니다.

이로 인해 위험을 감수하려는 의지가 높아져 제대로 검증되지 않은 제품이 시장에 출시되는 결과를 낳을 수 있습니다. 고객도 또한 경쟁사 선택하지 않을 이유를 더 이상 찾을 수 없을 때까지(적어도 잠시 동안은) 일정 수준의 제품 불완전성을 받아들일 것입니다.



PLM 및 ALM 다시 보기: 증가하는 복잡성에 대응하기

소프트웨어 기반 제품의 개발 최적화라는 "예술"은 모든 관련 당사자에게 투명성, 효율적인 협업 허브 및 발전에 필요한 모든 도구를 제공하는 프로세스, 방법 및 도구를 확립하는 데 있습니다.

그렇기는 하지만, 여전히 모든 개별 도메인을 조직화해서 최종 제품이 단일 제품으로서의 모든 요구사항 및 기능을 충족하도록 만들어야 합니다. 그러나 이것은 도구와 방법론의 문제만이 아닙니다. 광범위하고 깊은 조직적 변화가 필요하며, 이를 위한 의지가 중요합니다. 조직 내에서 변화를 감독하고 조직화를 위한 강력한 지침을 제공하는 역할을 맡을 사람을 지정하는 것이 좋습니다.

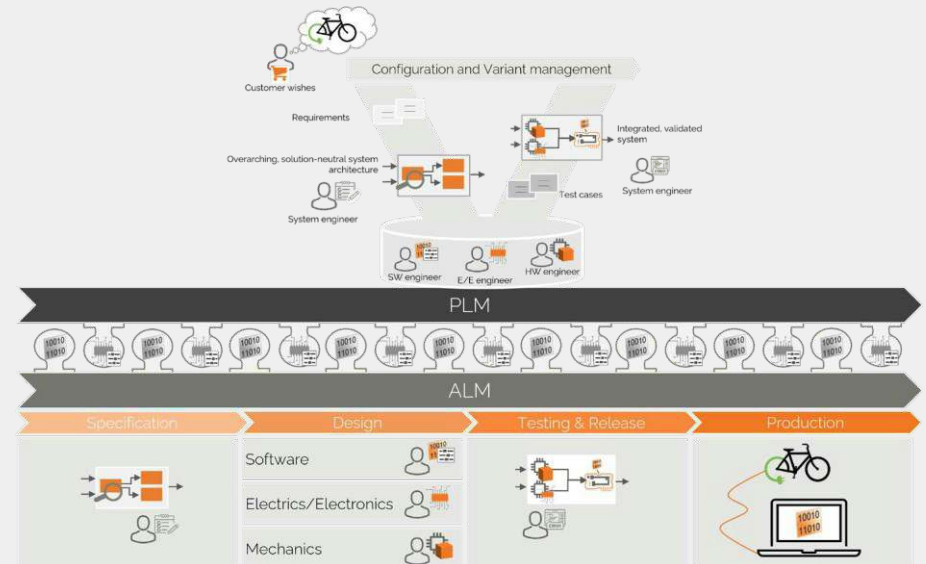
방법론적 기초

조직이 새로운 사고방식과 작업 방식을 수용할 의지가 생기면, 다음 단계는 솔루션 개발을 위한 공동 절차 모델을 확립하는 것입니다.

즉흥적인 계획과 조정(다양한 분야 간의 산발적이고 무질서한 상호 작용)은 효과가 없습니다. 협업적인 개발 프로세스를 기술 개발의 복잡한 환경에서 적극적으로 관리해야 합니다.

여기서 시스템 엔지니어링에서 사용되는 방법이 적절한 기초를 제공할 수 있습니다. 이 방법에는 제품 또는 시스템의 모든 구성 요소를 공유된 요구 사항에 맞게 조정할 수 있는 매우 유용한 도구 키트가 이미 포함되어 있습니다.

시스템 엔지니어링 표준 중 하나를 정확히 구현하는 것이든 단순히 지침으로 사용하는 것이든, 이는 거의 선호도의 문제입니다. 물론 일부 산업에서 요구되는 것처럼 고객이나 기타 이해 관계자에게 특정 표준에 대한 준수 여부를 증명해야 하는 경우에는 다릅니다. 이 경우에는 방법이 매우 중요합니다.



적절한 절차 모델의 사용

기업이 적절한 절차 모델(예: V-모델)을 선택하고 이를 나침반처럼 사용하는 것이 중요합니다.

소프트웨어 기반 제품의 맥락에서 이것이 의미하는 바는 무엇 일까요? 가장 처음에는 제품의 역할이 무엇인지, 그리고 이 제품이 충족해야 하는 기타 요구 사항(예: 표준)은 무엇인지 파악하는 것이 중요합니다. 이는 특정 접근 방식을 염두에 두지 않고 가능한 공정하게 이루어져야 합니다. 다음으로 구체적인 단계를 살펴보겠습니다

제품의 역할 설명

제품과 관련된 요구 사항이 명확해지면 다음 단계는 각 하위 분야(기계, 전자, 소프트웨어)가 기여할 기능을 결정하는 것입니다. 여전히 모든 참여자가 긴밀하게 협업해야 하는 단계에 있습니다. 그러나 모든 것을 마지막 세부 사항까지 지정하려는 유혹에 굴복하지 마십시오. 전자 자전거를 예로 들어보겠습니다. "전자 자전거 잠금장치" 기능에 대한 요구 사항을 충족하려면 이 단계에서는 자전거의 디스플레이를 사용하여 소프트웨어를 통해 작동할 수 있는 일종의 기계적 잠금 장치가 있어야 한다는 것만 지정하면 됩니다. 실제 구현 방법은 아직 관련이 없습니다.

올바른 방식으로 시스템 설계하기

아키텍처 단계에서는 두 가지 중요한 측면이 있습니다. 첫째, 시스템을 합리적으로 분할하는 것이고, 둘째, 시스템 요소 간의 상호 의존성에 대한 추상적 초기 정의를 제공하는 것입니다. 이 맥락에서 "합리적"이라는 것은 하위 시스템 간의 의존성을 최소화해야 한다는 것을 의미합니다. 왜냐하면 기업 내 개별 개발 팀 간의 모든 조정이 이 시점부터는 이러한 의존성을 중심으로 이루어질 것이기 때문입니다. 프로젝트의 진행에 따라 제품이 완전히 정의될 때까지(또는 MVP로 정의될 때까지) 인터페이스를 점점 더 자세히 설명하는 것이 중요합니다.

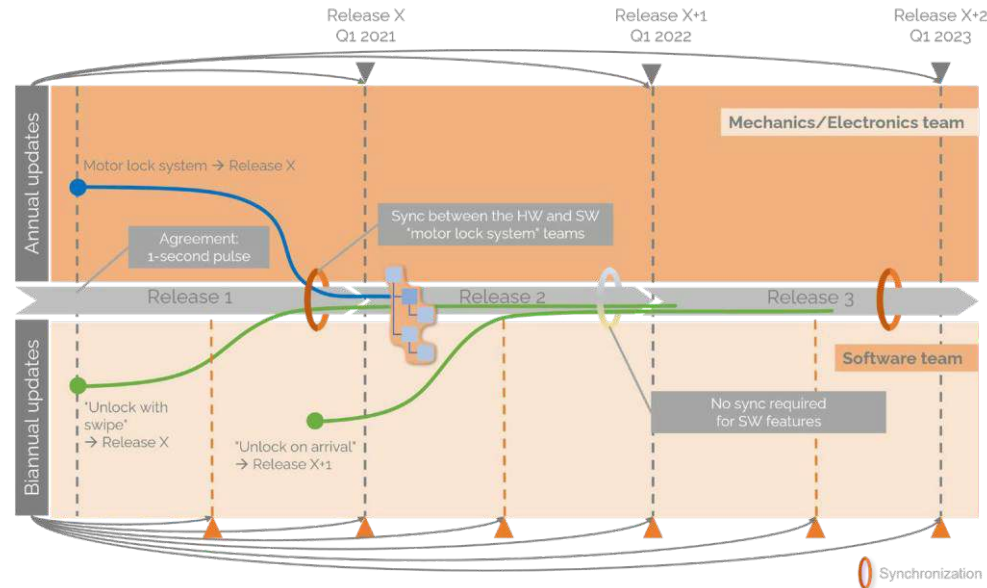
초기 개발을 넘어서, 시스템 분할 방식과 종속성에 대한 설명도 지속적 인 제품 유지 보수에 중요합니다. 이 접근 방식은 각각의 개별 부문이 성공하고 각자의 속도로 민첩하게 혁신을 추진할 수 있게 해줍니다. 단, 하위 시스템의 종속성에 의해 부과된 한계를 초과하지 않는 한에서만 가능합니다. 이는 소프트웨어 기반 제품의 경우 하드웨어와 기계가 제약을 가하지 않는 한 소프트웨어 측면에서의 가능성은 무한하다는 것을 의미합니다.

동기화 메커니즘 생성

더 포괄적인 추가 개발이 발생하는 경우, 소프트웨어만으로는 모든 새로운 요구 사항을 구현할 수 없기 때문에 여러 하위 시스템과 분야를 고려해야만 하는 경우가 거의 전부라고 할 수 있습니다. 즉, 하드웨어에도 변경이 필요합니다. 따라서 주요 기능 혁신과 하위 시스템 인터페이스의 확장을 명시한 개발 로드맵을 수립해야 합니다.

전체 시스템이 속도를 설정하면 관련된 모든 하위 시스템이 이를 따라야 합니다. 예를 들어, 전기 자전거 제조업체는 매년 업데이트된 모델을 시장에 출시할 수 있습니다. 개발팀은 이러한 연간 업데이트를 위해 계획된 주요 기능을 적시에 구현해야 하며, 이는 물론 하위 시스템 간의 인터페이스도 아키텍처의 맥락에서 정의되어야 함을 의미합니다. 예를 들어, 전자 자전거 잠금 기능은 다음과 같을 수 있습니다.

그렇지 않으면 두 하위 시스템이 서로 다른 속도로 독립적으로 개발될 것입니다. 예를 들어, 기본 변형 및 기계/전자 장치의 개조 수리는 각 모델 연도별로 계획하지만 새로운 소프트웨어 버전은 애자일한 방식으로 매월 또는 더 짧은 간격으로 출시할 수 있다는 것을 의미합니다.



맺음말 & 요약

시스템을 기계/전자 및 소프트웨어 하위시스템으로 나누는 것은 물론 하나의 예시일 뿐입니다.

더 복잡한 제품은 수많은 기타 하위 시스템, 나아가 동일한 유형의 여러 병렬 하위 시스템(예: 여러 소프트웨어 하위 시스템)까지도 포함할 수 있습니다.

모든 것을 단순하게 유지하려는 노력에도 불구하고, 이러한 하위 시스템 간의 상호 의존성은 매우 다양하고 복잡해질 수 있습니다. 따라서 IT 환경이 이러한 모든 상호 의존성을 최대한 잘 추적할 수 있도록 지원하는 것이 중요합니다.

이러한 유형의 방법 모델을 구축하면 개발 과정 전반에 걸친 소프트웨어 기반 제품의 신뢰성 있는 개발이라는 과제를 해결할 수 있는 기반을 마련할 수 있습니다.

미래에는 소프트웨어로 구동하는 제품은 점점 더 많아지고 그 소프트웨어의 구동 수준 또한 점점 더 높아질 것입니다.

비록 제품 복잡성의 증가는 불가피하더라도, 이러한 복잡성을 신뢰성 있게 관리할 수 있는 방법은 존재합니다. 특히, 엔드 투 엔드 시스템 엔지니어링은 이 부분에서 중요한 지원을 제공합니다.

하지만 이는 기업이 변화의 문화를 확립하고, 학제적 사고에 개방적이며, 오래된 습관을 버릴 준비가 되어 있다는 것을 전제로 합니다. 오늘날 대부분의 기업이 이미 이러한 개념의 많은 부분을 어떤 식으로든 실천하고 있기 때문에 미지의 것이라고 두려워할 필요는 없습니다. 많은 경우, 최적화된 목표 지향적 조직화가 부족할 뿐이며, 이는 적절한 사고방식 전환, 특정 담당자 배치 및 변화의 노력을 지원하는 적절한 도구의 사용 등으로 해결할 수 있습니다.





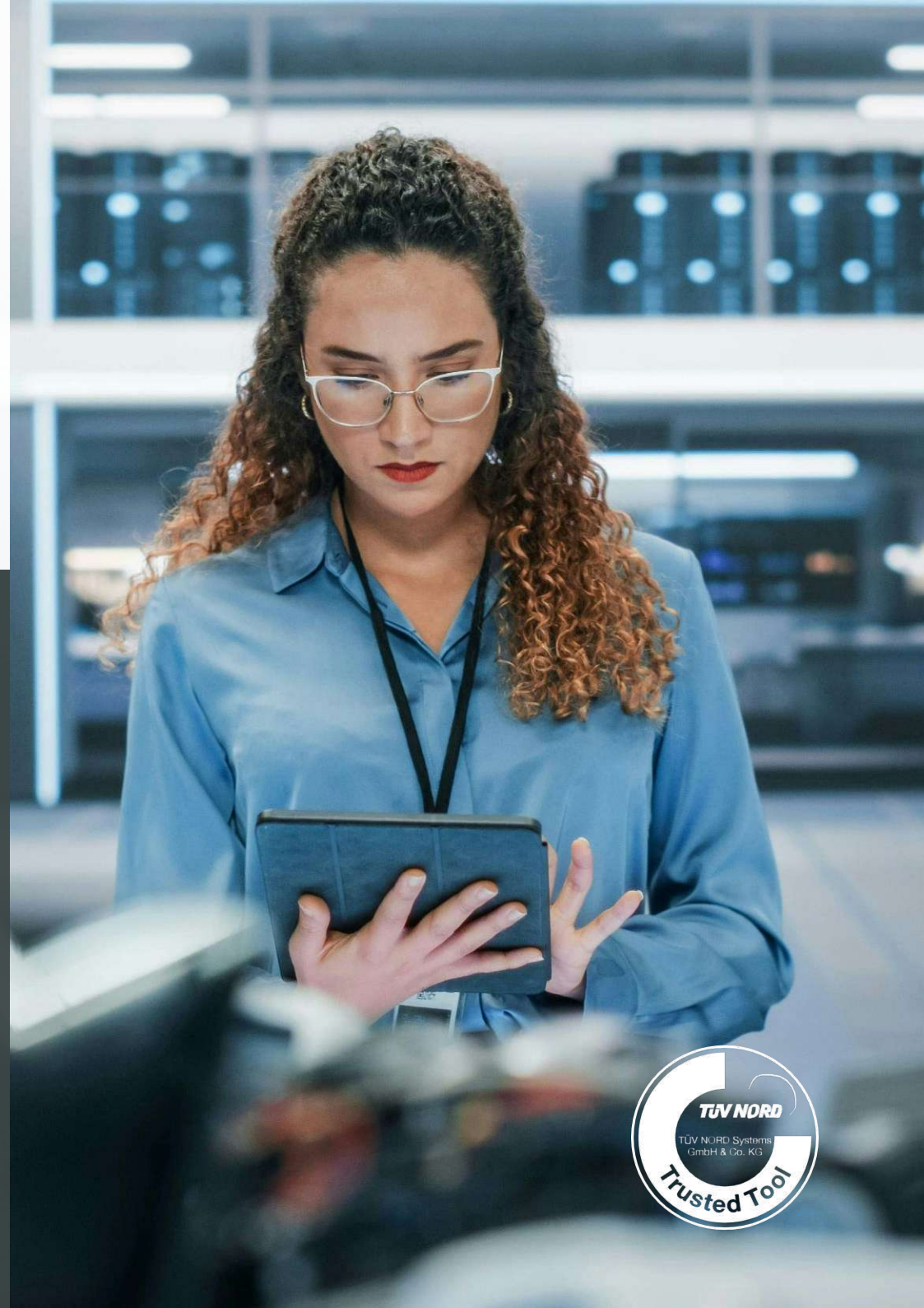
BHC GmbH는 **PROSTEP AG**의 일원으로 자동차 산업, 기계 공학 및 플랜트 엔지니어링 분야에서 제품 및 애플리케이션 수명 주기 관리에 대한 IT 관련 컨설팅을 전문으로 하고 있습니다. BHC GmbH의 핵심 역량은 메카트로닉스 및 소프트웨어 개발 분야의 기업을 위한 일관된 프로세스, 방법 및 IT 시스템 개발 컨설팅입니다. 자세한 내용을 알아보려면 [Linkedin](#)에서 **BHC GmbH**를 팔로우하세요.



Codebeamer는 복잡한 제품 및 대규모 소프트웨어 엔지니어링을 단순화하는 업계 선두의 소프트웨어 도구를 제공합니다.

PTC의 엔터프라이즈급 플랫폼은 기술 제품 개발을 가속화하고 규정 준수를 간소화하는 데 도움이 됩니다. 전 세계 자동차, 의료, 제약 및 생명 과학 개발 부문의 수많은 선도 기업이 혁신적이고 규정을 준수하는 제품 엔지니어링 프로세스를 관리하기 위해 PTC의 솔루션을 사용하고 있습니다.

자세히 알아보기





DIGITAL TRANSFORMS PHYSICAL

121 Seaport Blvd, Boston, MA 02210 : [ptc.com](https://www.ptc.com)

© 2023, PTC Inc. (PTC). All rights reserved. 본 문서에 기술된 내용은 정보 제공 목적으로 사전 통지 없이 변경될 수 있으며 PTC의 보증, 약속, 조건 또는 제한으로 해석되어서는 안 됩니다. PTC, PTC 로고 및 기타 모든 PTC 제품명과 로고는 미국 및 기타 국가에서 PTC 및/또는 그 자회사의 상표 또는 등록 상표입니다. 기타 모든 제품명 또는 기업명은 각 소유자의 재산입니다.

O20-integrating-hardware-software-development-digital-product-delivery-02-13