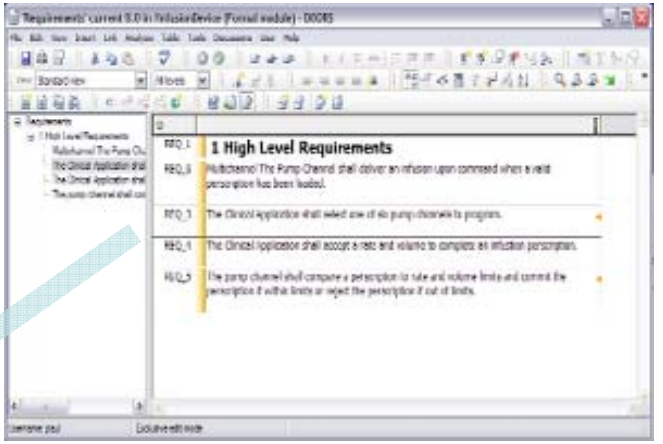
# 향상된 모델링 기능으로 스마트한 Android 서비스 개발하는 법

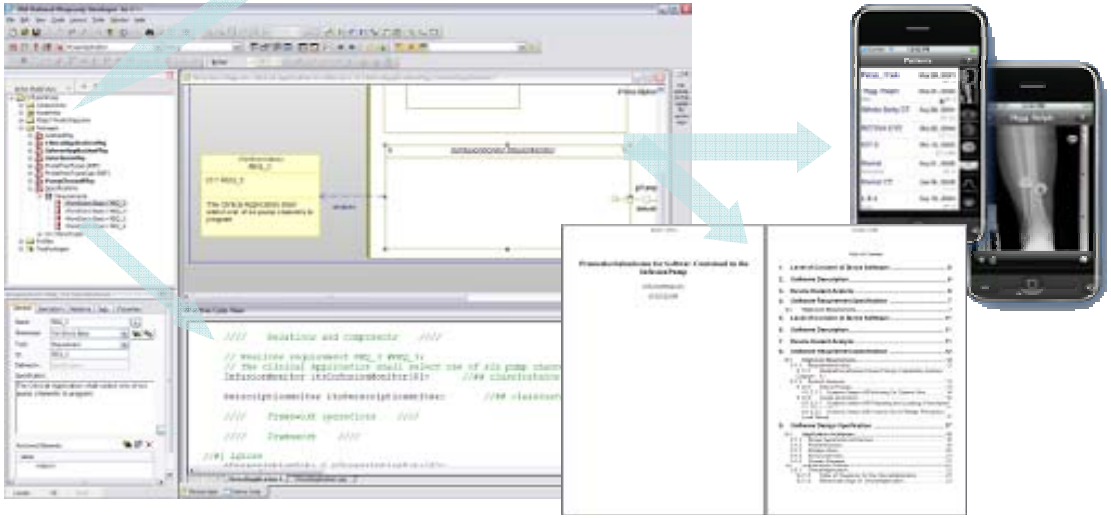
한국IBM Software group, Rational사업부
오세종 차장 (sejong@kr.ibm.com)

Let' build a smarter planet.

# Agenda

- Smart Device 란?
  - 무엇이 다른 걸까요?
  - 어떻게 대처해야 할까요?
- 간단한 데모
  - 명확한 역할 분할
  - 모델 기반 협력 (PIM/PSM)
  - 멀티플랫폼향 어플리케이션
- Smart Device 개발을 위한 IBM의 제안
  - 지역에 관계없이 Agile하게 협력할 수 있는 통합 개발 환경
  - 모델 기반 협력



# Smart device란?

- 빠른 속도 ?
- 다양한 활용도 ?
- 저전력, 고효율 ?
- 저가, 고성능, 휴대성…?

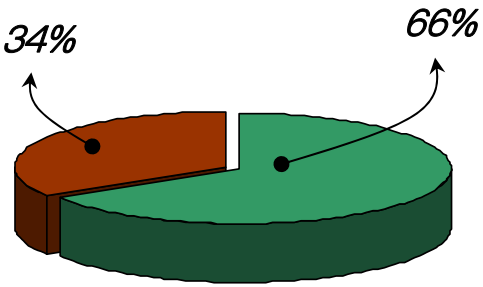
## 여러분의 의견은?

## 제가 생각하는 smart device란?

- Adaptable services :
  - ✓ 상황에 따라 최선의 결정을 내릴 수 있는…

# Smart의 핵심은 결국은 소프트웨어

Aberdeen Group
A Harte-Hanks Company




*개발된 모든 제품의 2/3 는 임베디드 소프트웨어 컴포넌트를 갖고 있다.*

*Source: Aberdeen Group, November 2008*

"의료 장비 영역은 특히나 소프트웨어 의존도가 높다. 소프트웨어는 우리의 진보된 환자 관리 시스템의 핵심이다."

MERGE Healthcare

"최신 자동차의 다른 모든 부품들처럼, 수력겸용 하이브리드 엔진 시스템은 매우 지능적인 소프트웨어 시스템이다."

EATON
Powering Business Worldwide

*"소프트웨어는 이제 단순히 제품의 기능만을 전달하는 숨겨진 파트가 아니라, 제품의 차별성과 사용자의 만족도를 결정하는 가장 중요한 파트로 부각되고 있다.*

- VDC Research

# Smart device의 미래를 보여주는 service 예 (1/2)

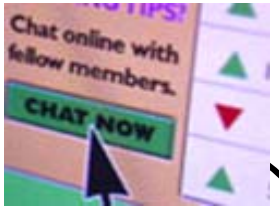
## [1] 개인별 Interface 서비스 제공

실제 사물과 Web 상의 정보 교환을 smart device가 중재하기 때문에 가능한 서비스

**[활용예-1]** 오프라인 상에서 Smart phone 끼리 인식을 통해 상대방 blog 방문/친구 신청 등

**[활용예-2]** 몸에 장착한 flexible chip을 통해 수집된 건강 정보를 수집하고, 이를 health care 센터에 보고하여, 개인 건강 관리를 종합적으로 제공하는 서비스

**[활용예-3]** 내가 소유/등록한 모든 기기의 상태를 분석하고, 원하는 동작을 수행시킬 수 있음




③ 자신에게 보여지는 상대방의 허용된 Interface상 메뉴를 통해 Interaction 요청 (예:블로그방문, 채팅요청 등)

② 개인의 허용된 Interface가 상대방 폰에 보여짐

① Bluetooth 또는 무선랜을 통한 폰간 인식

# Smart device의 미래를 보여주는 service 예 (2/2)

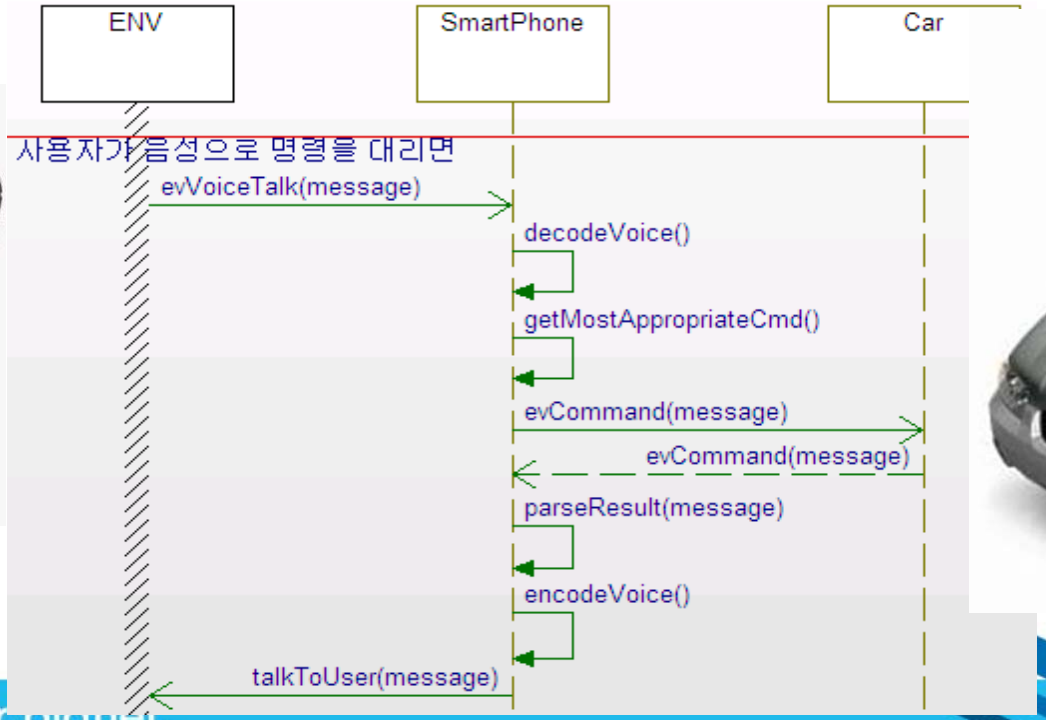
## [2] 사람과 사물과의 대화

Smart phone 은 소유주의 음성 정보를 가장 잘 파악할 수 있는 도구

따라서, 음성 인식 기반으로 상상할 수 있는 모든 서비스 어플리케이션의 시작점

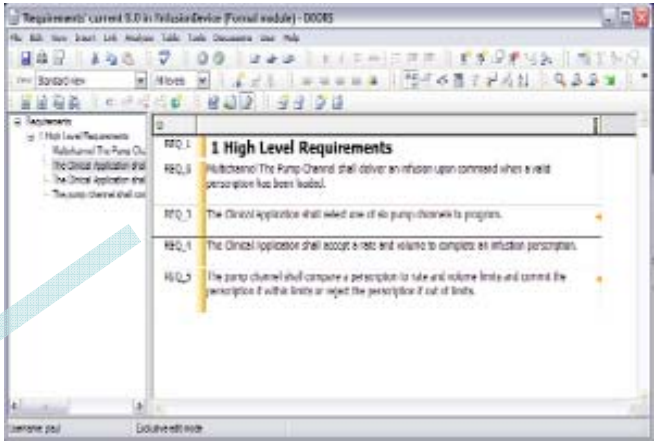
[활용예-1] H 자동차는 구매시 M-phone을 드림. M-phone에는 자신의 자동차와 대화할 수 있는 기능이 있음.

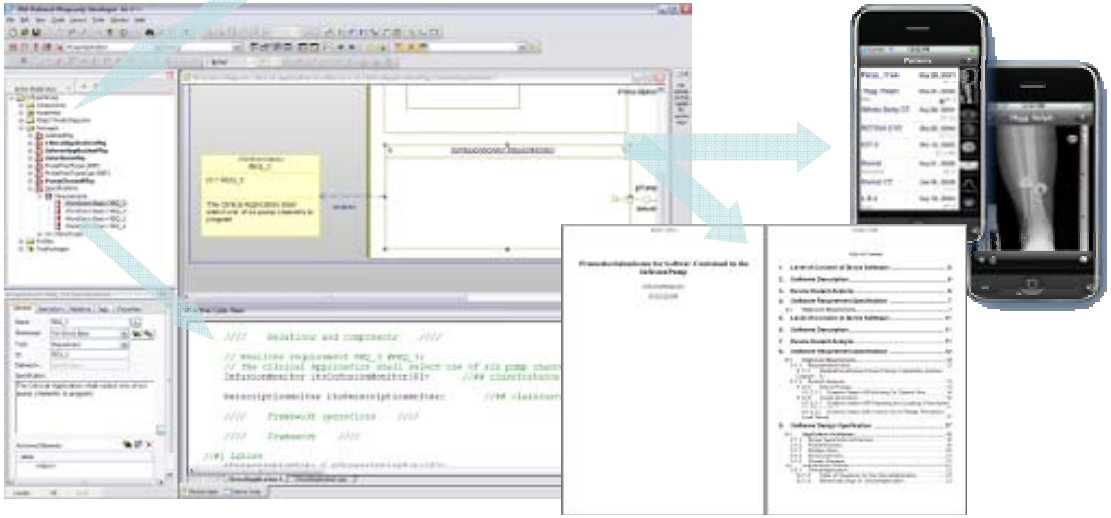
[활용예-2] S 전자는 자사의 모든 전자제품을 음성으로 조작할 수 있는 어플리케이션을 자신의 app-store 에서 판매함




# Agenda

- Smart Device란?
  - 무엇이 다른 걸까요?
  - 어떻게 대처해야 할까요?
- 간단한 데모
  - 명확한 역할 분할
  - 모델 기반 협력 (PIM/PSM)
  - 멀티플랫폼향 어플리케이션
- Smart Device 개발을 위한 IBM의 제안
  - 지역에 관계없이 Agile하게 협력할 수 있는 통합 개발 환경
  - 모델 기반 협력



# Rhapsody Android profile 소개

## [1] 개발된 Android app을 정적/동적으로 분석하는 데 도움

- Code visualization: class간, class 와 Android framework간 구조 분석용
- Animation : 실행 중 sequence diagram 형태로 동작을 보여줌.

## [2] Android framework의 분석과 개선에 도움

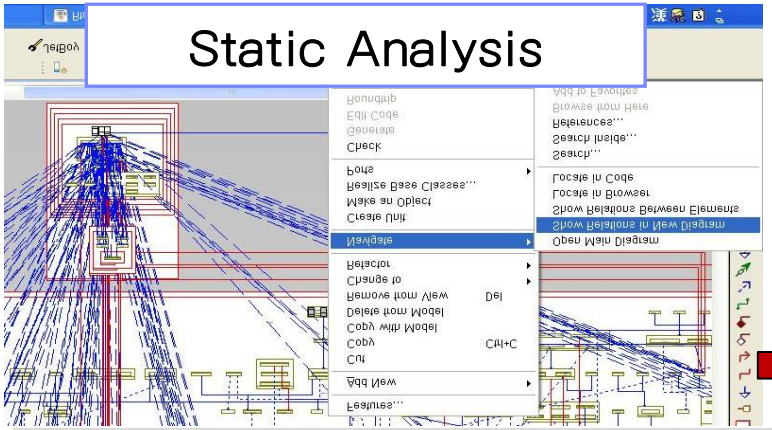
- Android framework 역시 [1]의 방법대로 정적/동적 분석이 필요함.
- 고객 입장에서 [1]의 검증이 만족스럽다면, [2]는 약간의 시간과 인력이 더 투여될 뿐 같은 작업이라고 봄

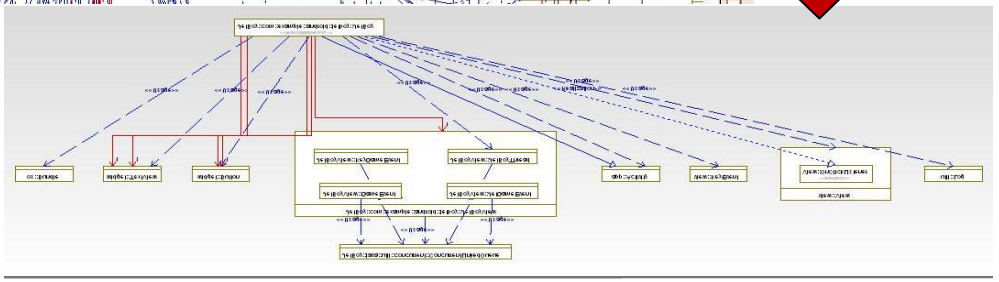
## [3] Android를 포함한 새로운 서비스를 개발할 때 설계와 검증에서 외부와 협력 도움

- 예를 들어, 단말로는 Android를 사용하고 연동 대상은 자동차나 가전 제품, 또는 서버가 되는 서비스들.
- 이는 RTC와 연계하여 사내.외 모델 기반 협력 도구로 제안할 것이며, 협력의 효율을 높여 줄 수 있는 솔루션이라고 봄

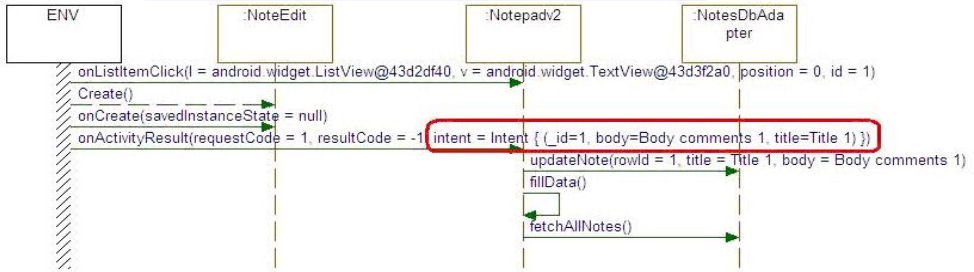
# Use case [1][2] : Android app/framework에 대한 정적/동적 분석

## Static Analysis



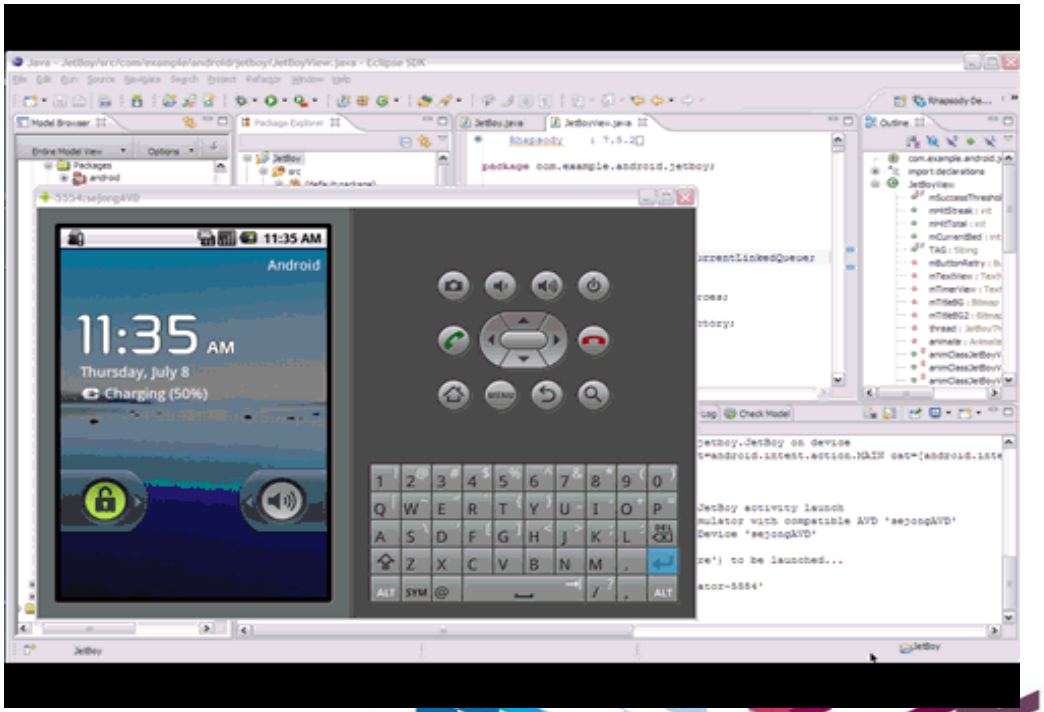


## Intent Analysis



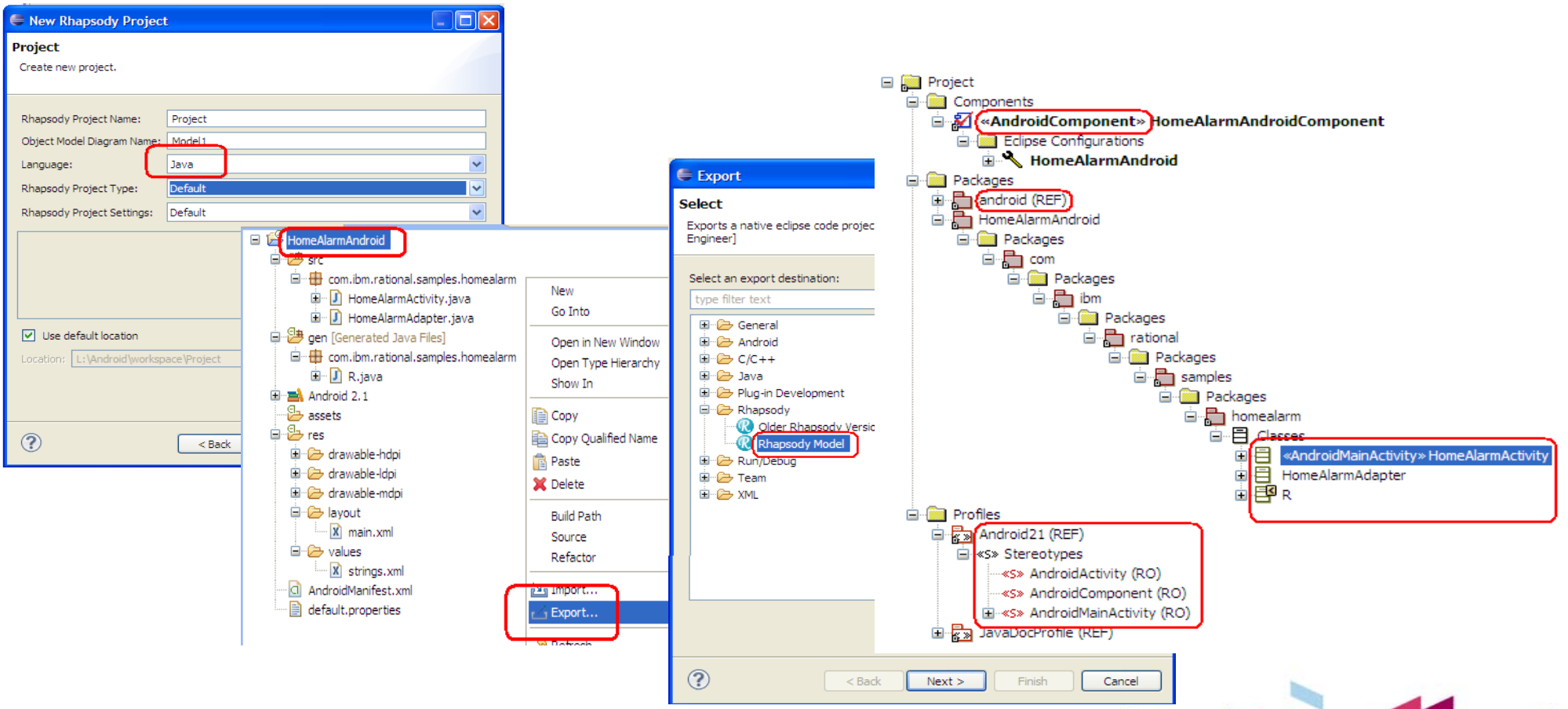

- Android profile 기반으로 Android app을 정적/동적으로 분석할 수 있습니다.
- 이는 개발자 그룹이 시각적인 자료를 통해 정보를 공유하고 협력할 수 있는 기반을 제공합니다.

## Runtime Analysis



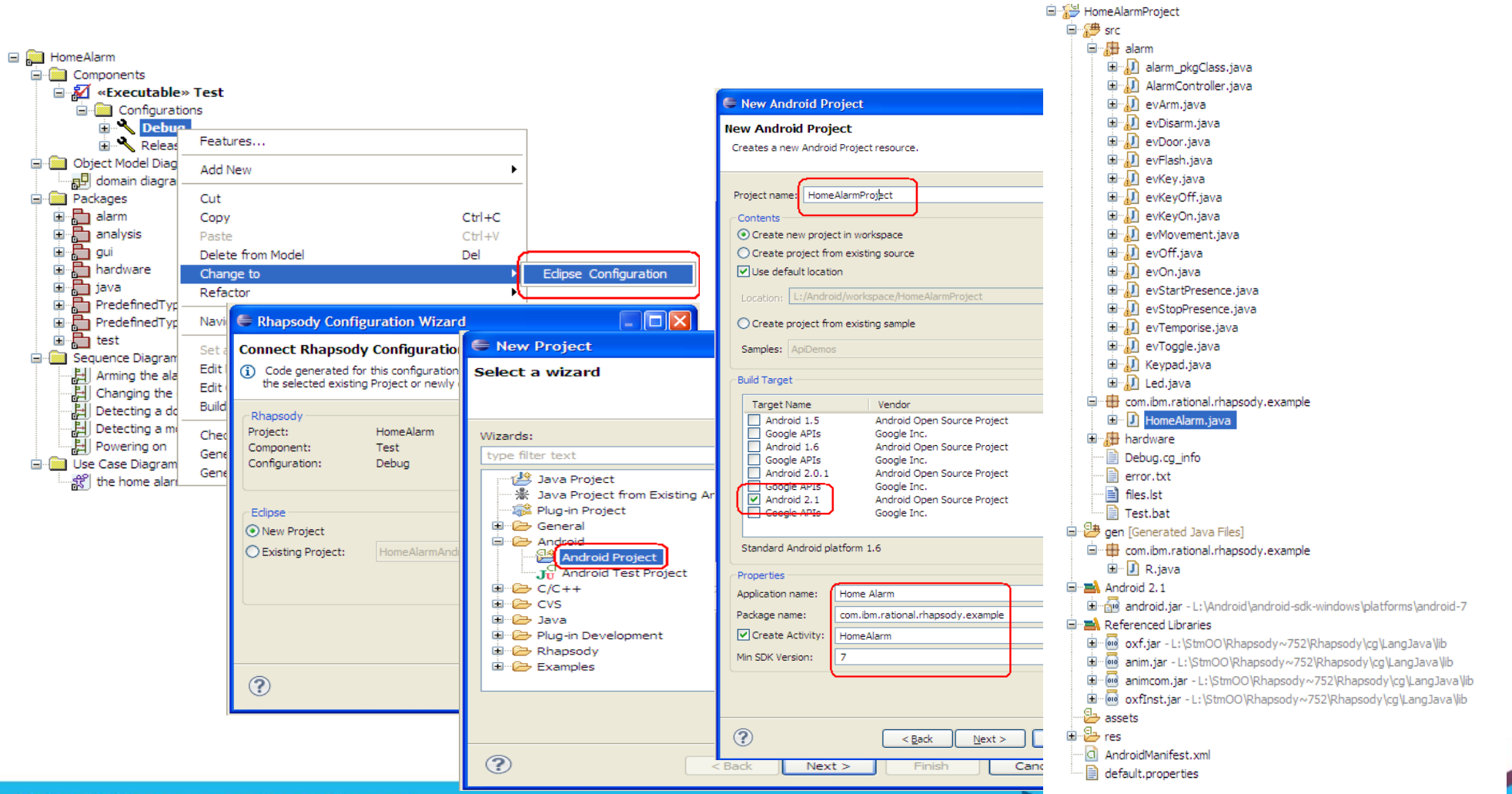
# Use case[1][2]를 위해 Android profile을 사용하는 예

- Android app을 Rhapsody 프로젝트로 export 하여, model 과 code를 동기시킨 후, 설계와 개발을 병행하는 목적으로 사용합니다.



# Use case[3]을 위해 Android profile을 사용하는 예

- Rhapsody로 설계되어 검증된 어플리케이션을 Android profile을 적용한 후, android 향 어플리케이션으로 타겟팅할 수 있습니다.



# Android profile 에 대한 질문 모음

## [질문 1] Rhapsody는 Android app을 개발하기 위한 도구입니까?

- Rhapsody는 UML 도면 (class diagram, state chart diagram, sequence diagram)으로 검증하고 코드 생성

-Android app의 개발에 UML이 적합한가요?  멀티플랫폼향 어플(단말, 중계, 서버 연동)의 경우 ?

## [질문 2] UML의 state chart는 큰 시스템에서만 적용 가능한가요?

- 훈련이 필요할 뿐, 일정 규모 이상의 로직의 설계/검증에 활용할 수 있음
- State chart는 모델 기반 자동 테스트의 시작점이며, 이는 UML 사용자의 향후 최대 장점이 될 것임
- 모든 소프트웨어의 가장 큰 문제는 어떻게 검증할 것이냐인데, 이는 State chart 가 해법이 될 것임

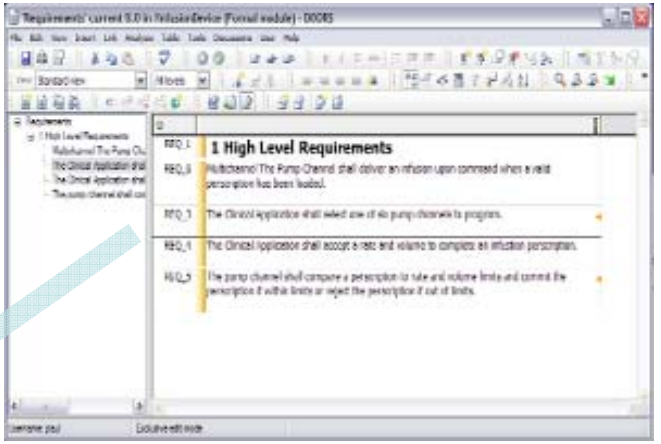
## [질문 3] 모델링을 시작하면 코드 개발자는 불필요해지나요?

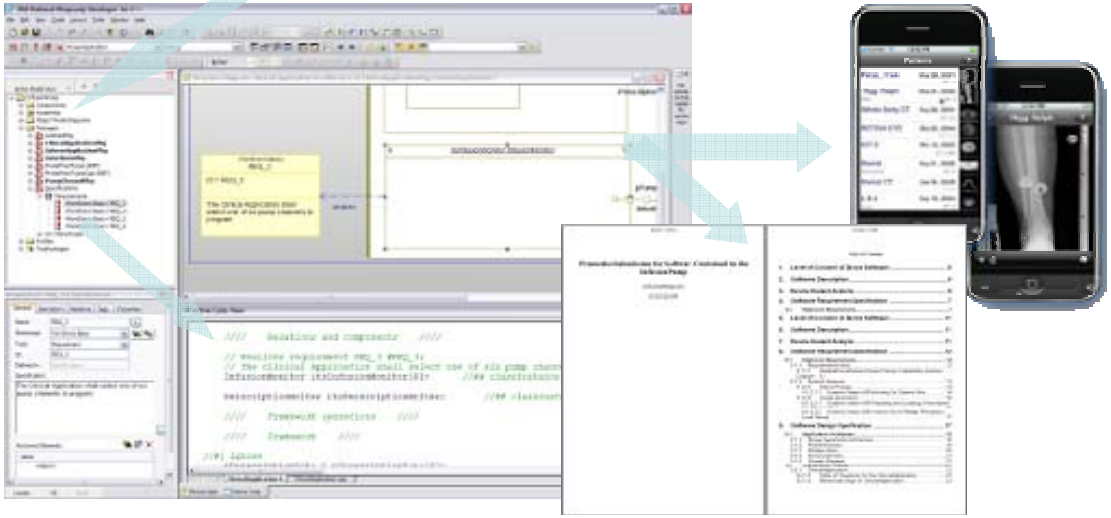
- 모델은 건물의 Executable framework 이고, 코드는 벽돌이고 목재이다.
- 잘 만든 framework은 재사용을 통해 소프트웨어 개발의 비용과 시간을 절감시킨다.

# Agenda

- Smart Device란?
  - 무엇이 다른 걸까요?
  - 어떻게 대처해야 할까요?

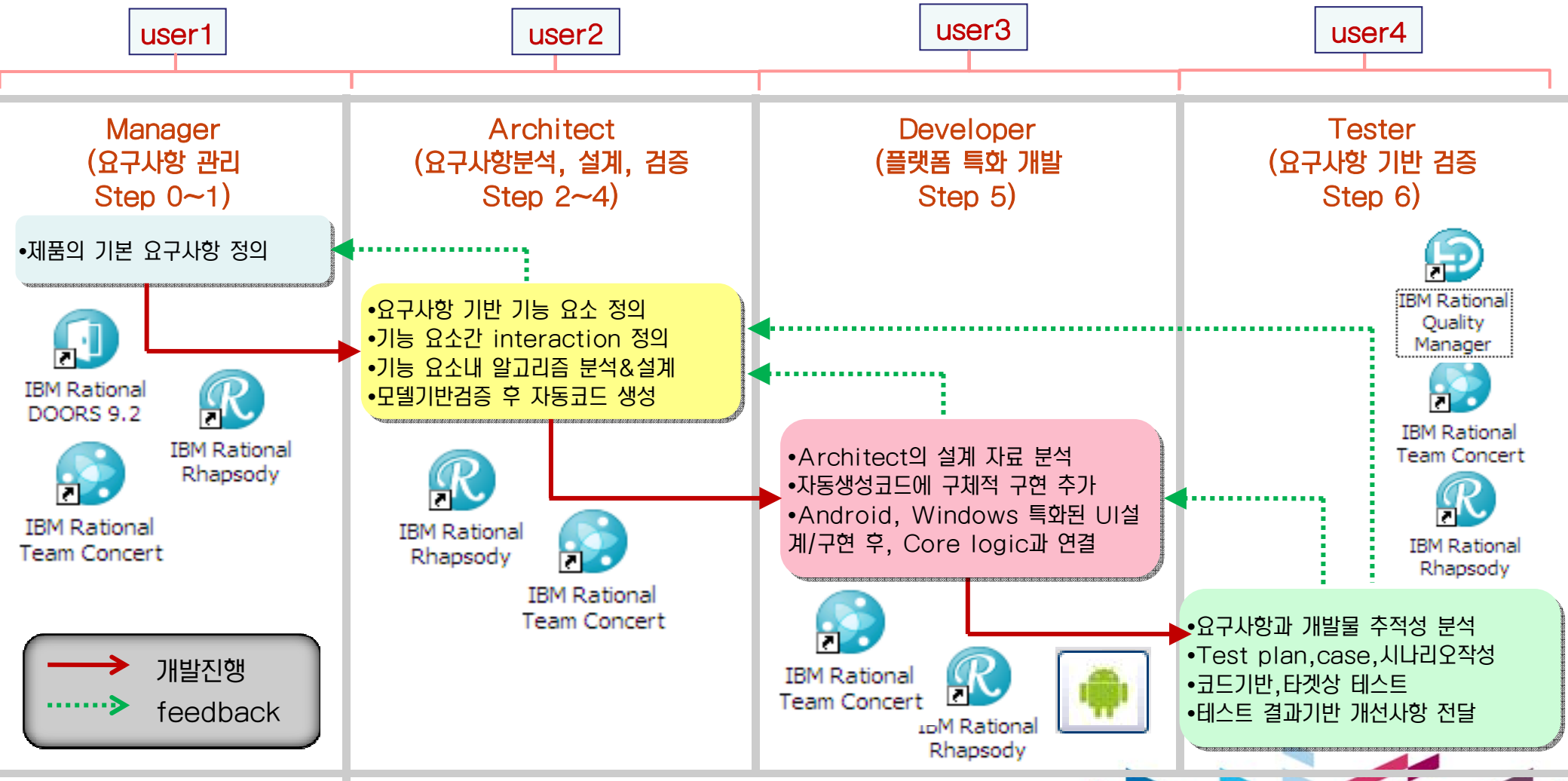
- **간단한 데모**
  - **명확한 역할 분할**
  - **모델 기반 협력 (PIM/PSM)**
  - **멀티플랫폼향 어플리케이션**



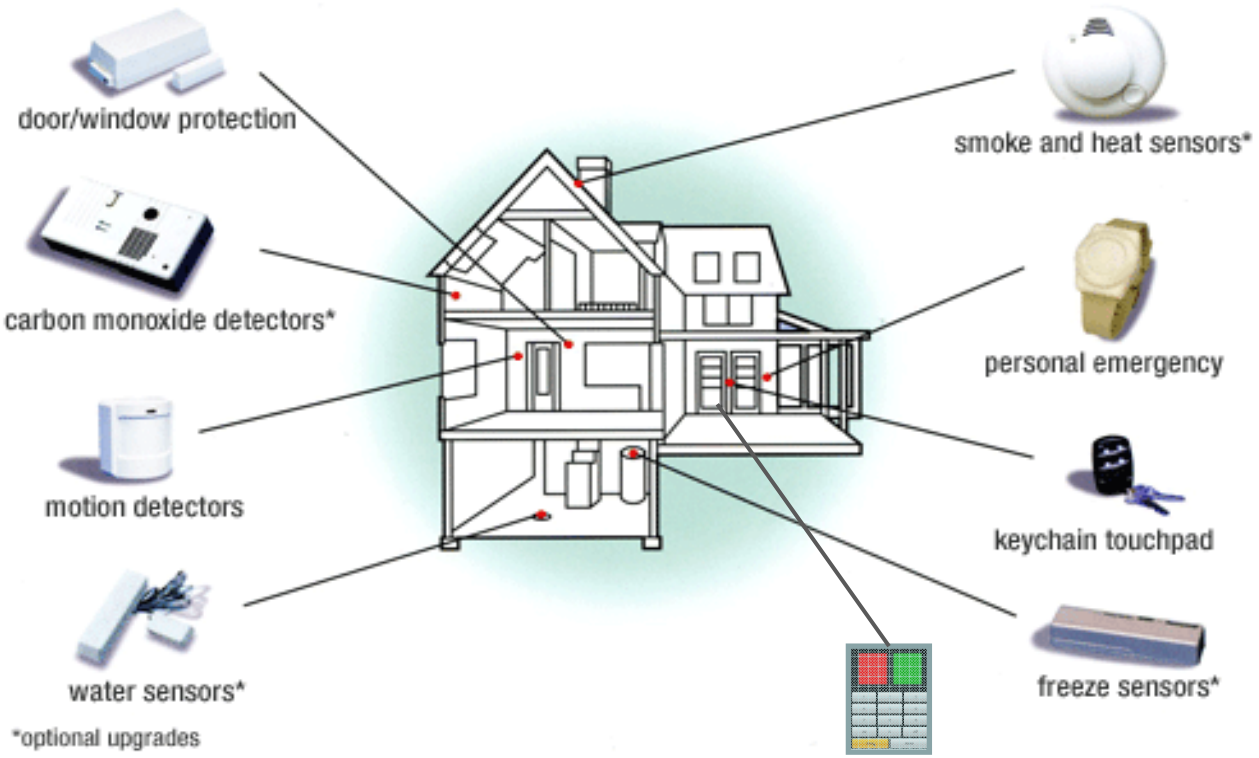
- Smart Device 개발을 위한 IBM의 제안
  - 지역에 관계없이 Agile하게 협력할 수 있는 통합 개발 환경
  - 모델 기반 협력

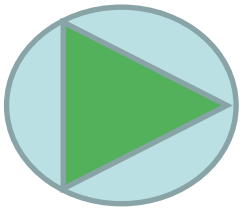
# 멀티플랫폼향 어플리케이션 개발을 위해 유용한 **툴체인 데모**



# [Step 0] 요구사항부터 시작

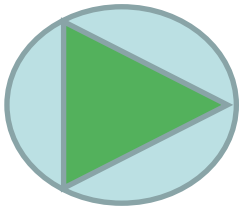
- Control panel과 Alarm, LED, Sound, 동작 검출 센서 등으로 구성된 홈 제어 시스템
- Arming 후에, monitoring 하고 있는 이벤트가 검출될 때 다양한 방식으로 경보
- 비밀번호 관리, 모니터링 관리, 경보 관리등이 원격 또는 로컬에서 가능



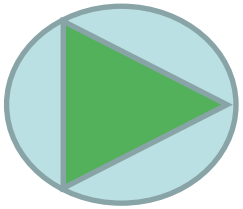
# [Step1] 아이디어의 도출 단계 (User1 영역)

- 모델 검증을 할 수 있는 플랫폼 결정 (Java? C++? C?)후 프로젝트 시작함
- 요구사항을 모델 요소로 import 함
- 시스템을 use case별로 분석하고, 개발 scope과 팀별 role을 결정함
- use case 와 요구사항간 추적성 연결함. 향후에 요구사항 coverage 분석에 사용할 것임



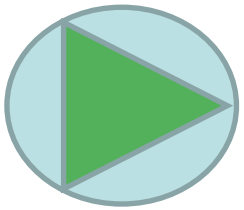
# [Step2] 프로젝트 공유 (User1과 User2 간)

- User1이 PM으로서, User2와 User3을 HomeAlarm 프로젝트에 추가함
- User1은 HomeAlarm 프로젝트의 기본 자료로서, 자신이 요구사항을 기반으로 작성한 Use case diagram을 전달함



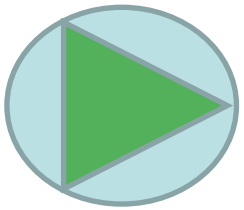
# [Step3] 아이디어의 도식화 (User2 영역)

- 모델로 import 된 요구사항을 기반으로 Use case 별 sequence diagram 그리기
- Sequence diagram 에서 자동으로 모델 element를 생성하기



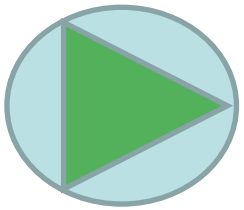
# [Step4] 아이디어의 구체화 (User2 영역)

- Statechart 와 같은 구체적 동작에 대한 모델링 부분을 수행
- Statechart가 채워지는 대로, 바로 visual simulation을 통해 검증을 수행함



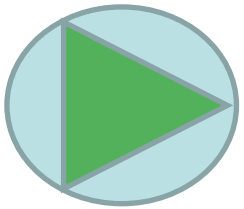
# [Step5-1] 플랫폼 특화된 개발 (User3 영역)

- Core logic이 검증된 모델을 받아서, 이를 Android UI 와 붙이는 작업을 수행함
- 이 때, Android에서 제공하는 서비스나 API를 Core logic의 Statechart와 붙이는 작업이 필요함



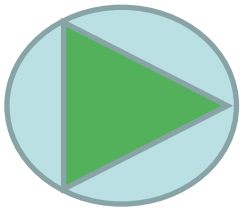
# [Step5-2] 플랫폼 특화된 개발 (User3 영역)

- Android emulator에서 검증된 Keypad와 달리, 실제 Home Server에 로딩될 Alarm 부분을 타겟(예: ARM core)에 맞게 구성하는 작업
- 이미 검증된 Alarm의 statechart에서 ARM core에 포팅할 C++ 코드를 생성하는 작업



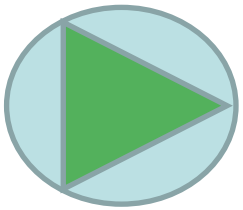
# [Step 6-1] 모델기반 TestCase와 Scenario 작성 (User2,3 영역)

- Test하려는 모듈을 선택하여 test architecture 생성
- Test 목적(BB, WB)에 따라 적당한 Test case를 생성
- 분석을 위해 작성했던 기능 요구사항별 Sequence diagram을 Test scenario로 변환
- Test case에 test scenario를 조합하고, test case를 시스템 요구사항에 연결



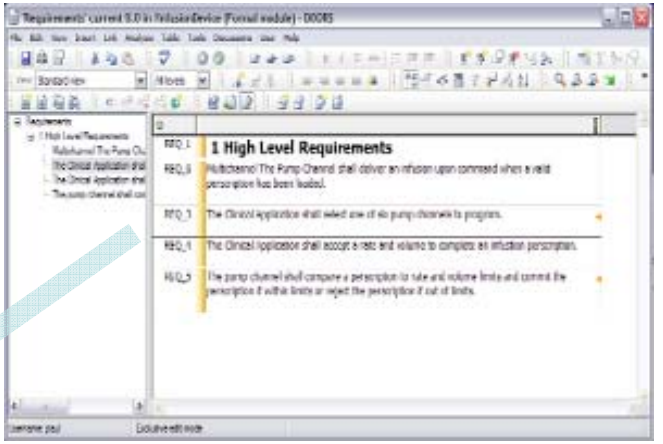
# [Step 6-2] 요구사항 기반 테스트 (User1 & User4 영역) iteration1 완료

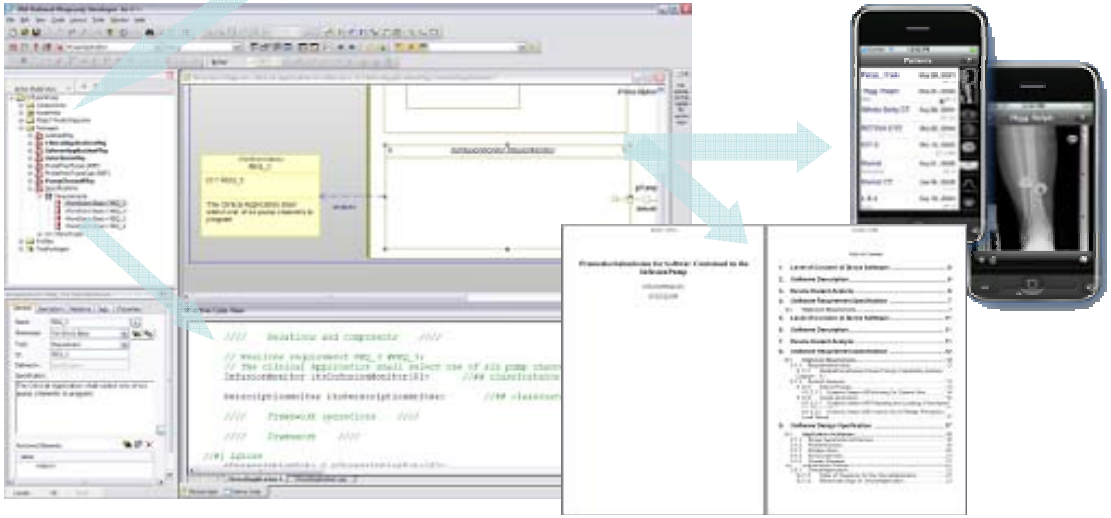
- DOORS의 요구사항을 Test 관리 툴(Rational Quality Manager)로 import
- Tester는 architect와 Developer에게 요구사항별 Test scenario를 요청함
- Test scenario 별 검증 후, 결과를 당사자에게 보고하여, 향후 보완 작업에 반영토록 함



# Agenda

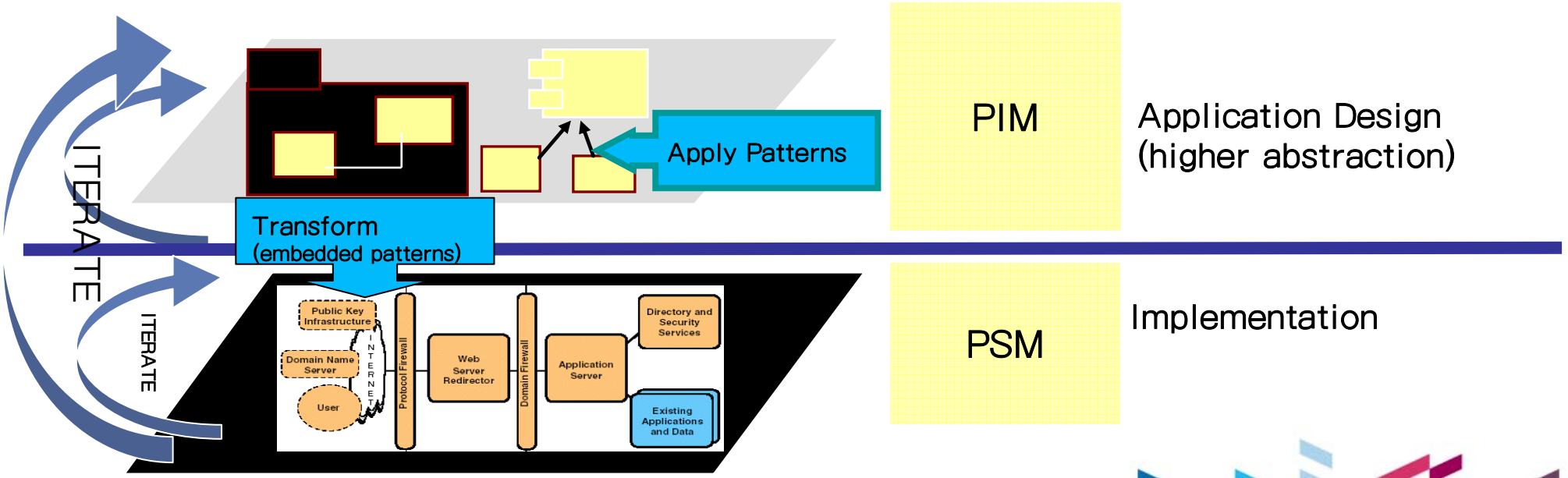
- Smart Device란?
  - 무엇이 다른 걸까요?
  - 어떻게 대처해야 할까요?
- 간단한 데모
  - 명확한 역할 분할
  - 모델 기반 협력 (PIM/PSM)
  - 멀티플랫폼향 어플리케이션
- Smart Device 개발을 위한 IBM 의 제안
  - 지역에 관계없이 Agile하게 협력할 수 있는 통합 개발 환경
  - 모델 기반 협력



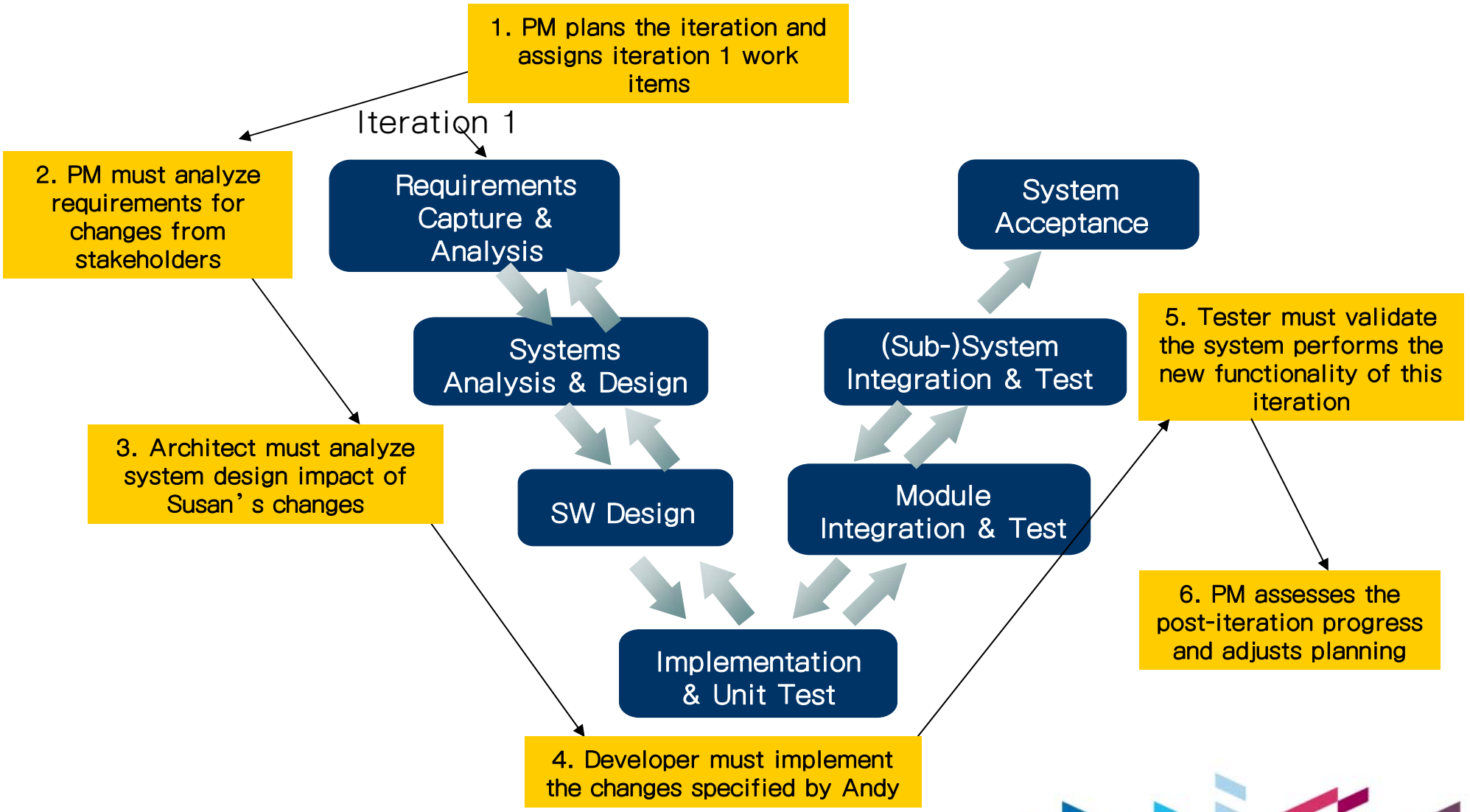



# PIM과 PSM 기반 Multi-platform 향 소프트웨어

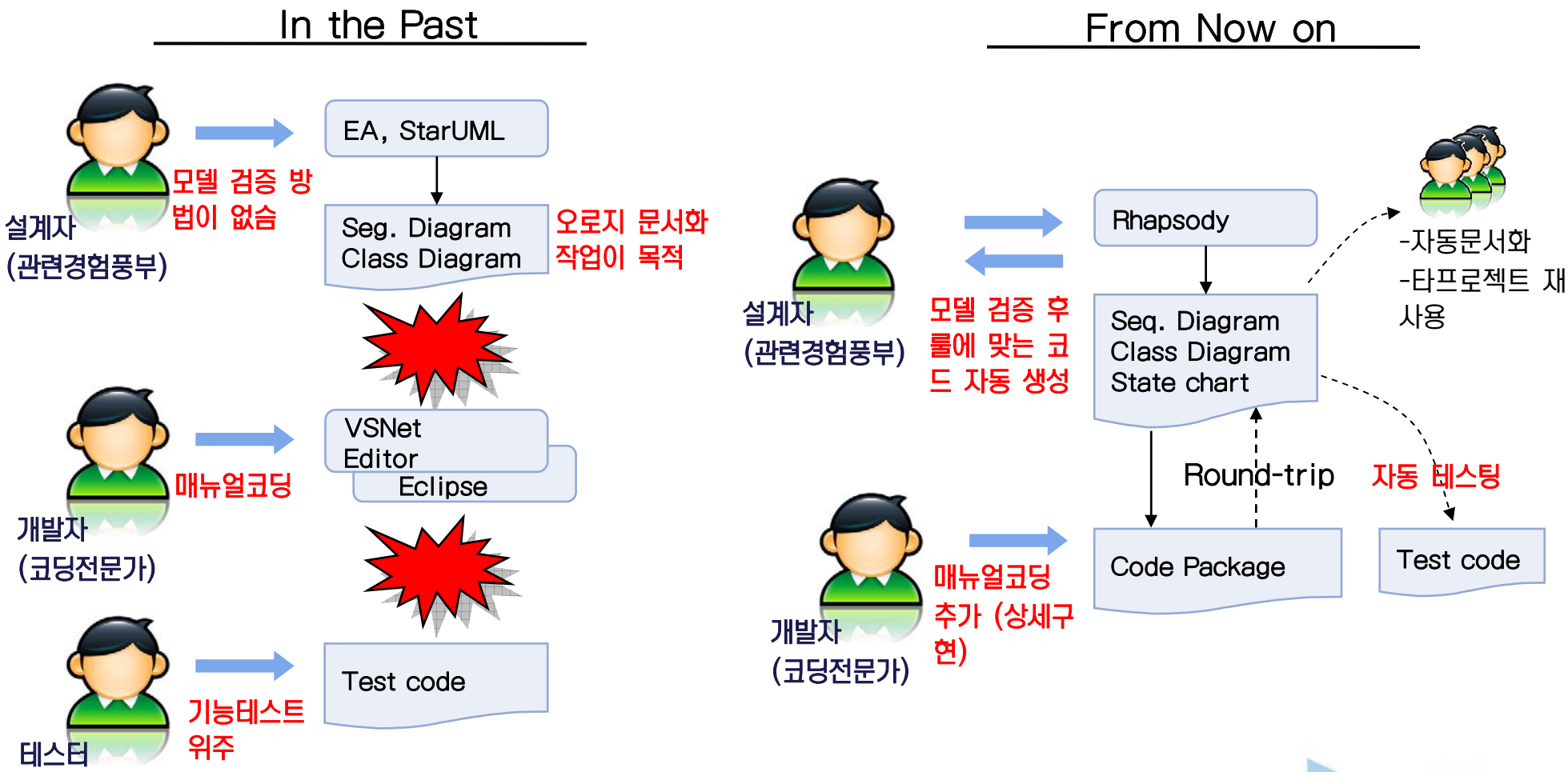
- PIM의 특징
  - platform 특화된 API와 관계 없이 시스템을 표현할 수 있음
  - 모델링을 통해 구현 언어(C,C++,Java)와도 독립적
- PSM의 특징
  - 구체적 platform API를 사용, 구현 언어에 의존적



# Smart한 개발을 위한 V 모델 개발 방법론



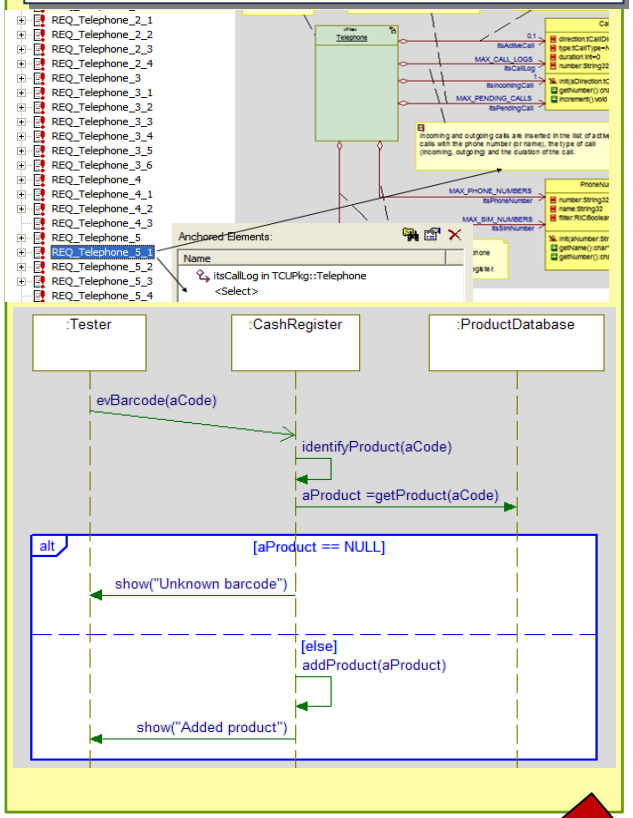
# 이제는 smart하게 코딩을 할 때입니다



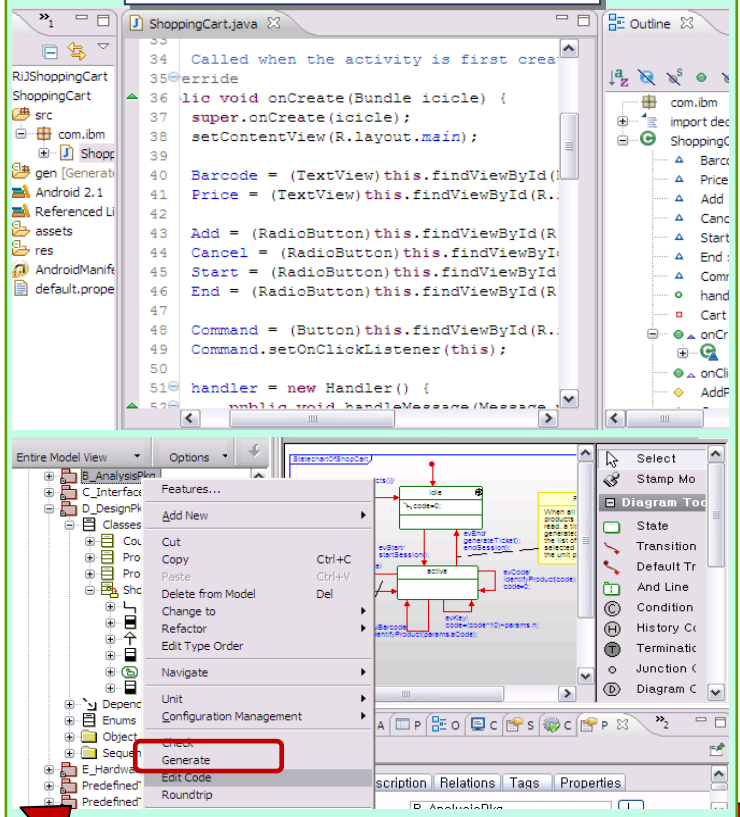
# Smart한 ALM 툴체인 활용법

## 아이디어를 시각적으로 공유하며 전세계적으로 협력할 수 있는 통합 개발 환경

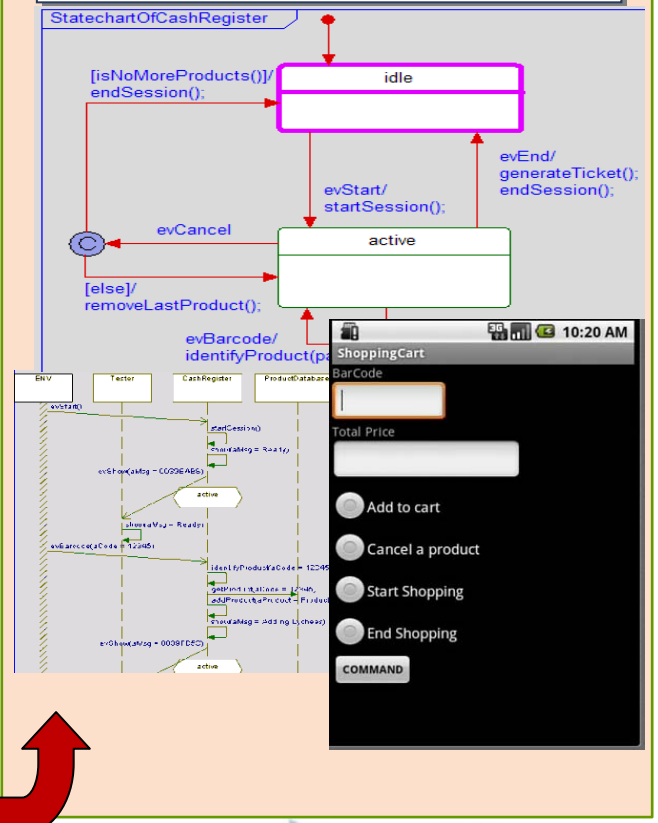
시스템의 요구사항과 분석/설계
(예: DOORS, RHP)




(2) 구체적 구현
(예: RTC, RHP, RSAR)



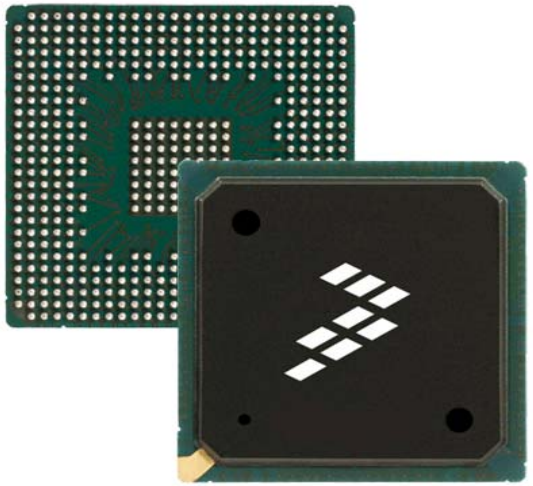

(3) 구현된 도면 기반 검증
(예: RQM, RHP, target SDK)



# 사례 1 : 통신 시스템용 DSP와 프로토콜 (Qualcomm)

W-CDMA 무선 접속망 (UTRAN)

- 25명의 개발자들
- 6개월 동안 170만 LOC
- 368 개의 컴포넌트들과 라이브러리
- 437 개의 configuration
- 186 개의 프로젝트

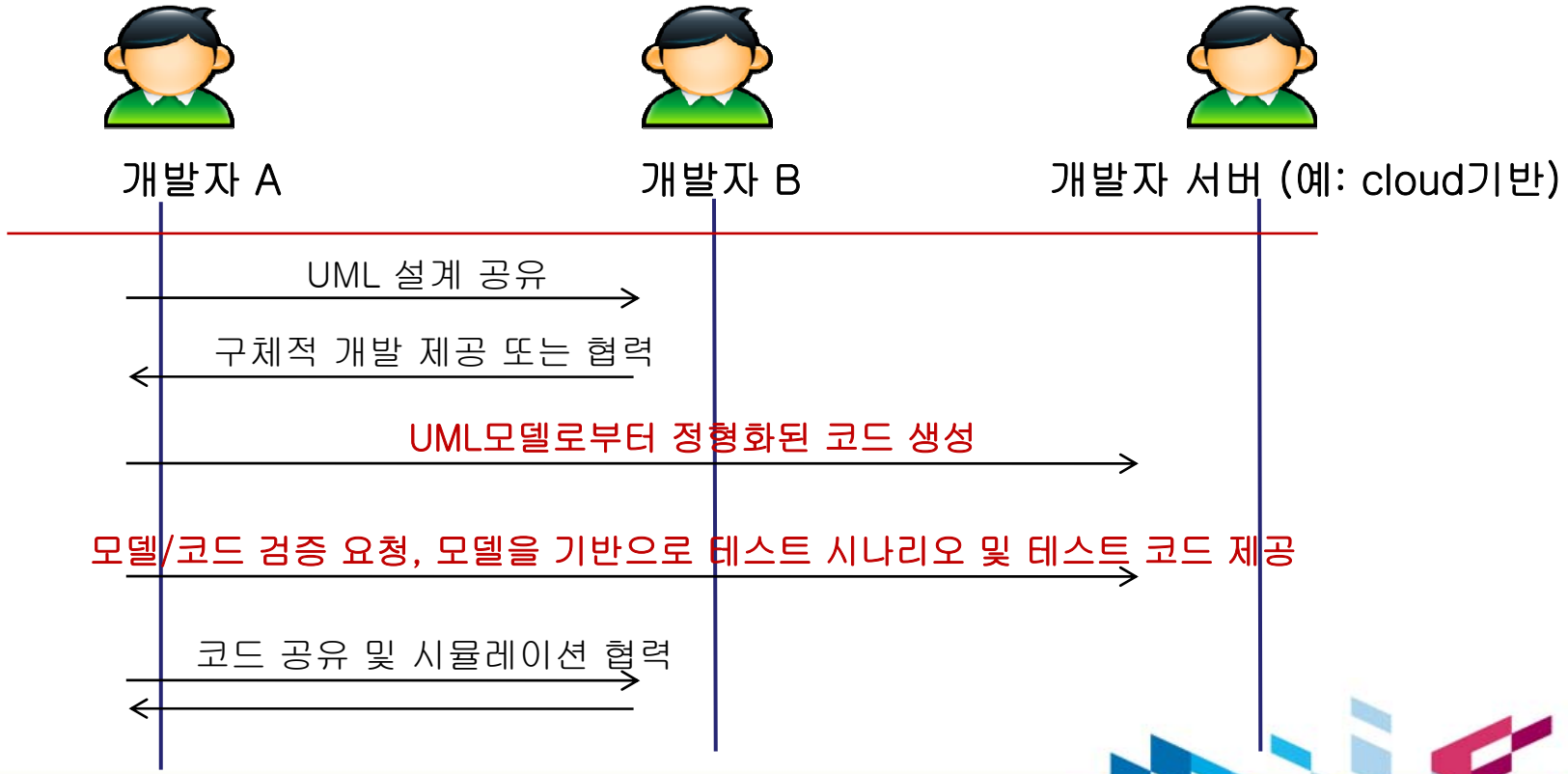


전반적 결과 수치들

- 시스템 연동(Integration) 에 소요되는 시간 80% 감소
- SE팀과 SW 팀간 협업에서 설계의 오류로 인한 defect이 40% 이상 감소하는 효과

# 모델을 기반으로 협력

- 아이디어와 구현이 분리될 수 있는 환경
  - Architect는 자신의 아이디어를 다양한 모습의 UML로 표현할 수 있음.
- PIM과 PSM의 지속적 iteration



# 사례 2 : 국내 가전 사업부 (차세대 세탁기 공정 제어)

*"We have to admit Rhapsody can help us to visually design and verify our requirements clearly. We also save lots of effort to setup middleware framework to be used in varieties of our products"*

## 무엇이 가장 큰 문제였나?

하드웨어와 직접 연관된 소프트웨어로 인해 재사용성이 낮음

Time-to-market을 위해, 재사용 가능한 공정 제어 로직이 시급함

## IBM 제안에 의해 나아진 점?

재사용성을 돕기 위한 Design pattern이 C로 구현하기는 참 어려움 !!!

모델링을 통한 시각적 설계를 통해, 유용한 Design pattern들을 활용할 수 있었음

실행 가능한 모델을 통해 target loading 없이 설계 내용을 신속하게 검증

**Framework pattern** 도입은 어플리케이션 내 요소들간의 의존성을 현저히 줄여줌


Questions








33