모바일 애플리케이션 개발 지침서

모바일 애플리케이션 프로젝트를 진행 중인 기업 실무자용 안내서

IBM

총괄 요약

모든 산업 분야에서 기업은 비즈니스 애플리케이션의 대상이 데스크톱 및 노트북 컴퓨터와 같은 기존의 PC 사용자에서 스마트폰과 태블릿을 통해 인터넷에 접속하고 원하는 정보를 찾는 사람들로 대다수 바뀌었다는 사실을 인지하기 시작했습니다. 애플리케이션의 사용 대상이 기업의 직접적인 고객(B2C 앱)이거나 직원 또는 비즈니스 파트너(B2E/B2B 앱)일 경우가 이에 해당합니다. 사용자 중심의 환경이 제공되고 사용이 편리하며 어딜 가든 휴대가 가능하기 때문에, 인터넷을 통해 정보를 얻고 서비스를 요청하는 주요 수단으로 모바일 디바이스를 사용하는 인구는 전세계적으로 계속 증가하고 있습니다.

일반 사용자 행동 양식의 중대한 변화로 인해 기업은 기존 비즈니스 애플리케이션을 위한 모바일 채널을 개발하고 시장에서 모바일 디바이스의 고유 특성을 활용할 수 있는 새로운 종류의 애플리케이션 개발 계획을 수립하고 있습니다. IT 산업 부문의 주요 혁신과 마찬가지로 변화의 초기 단계에서는 애플리케이션 개발 비용, 유지관리의 편의, 품질, 보안과 같은 전략적 고려사항에 대한 고찰 없이 수요를 충족하고 시장 진출과 점유율 확대를 위해 성급하고 대대적인 활동이 이루어집니다. 모바일 애플리케이션 시장이 성숙하고 초기의 흥분이 가라앉아 시장이 안정되면 장기적 계획 및 경제성을 중시하는 기업들로 인해 보다 포괄적인 소프트웨어 개발 이슈들로 관심이 집중됩니다.

IBM은 다양한 산업 부문에서 견실하고 책임감 있는 소프트웨어 개발 파트너로써 그 명성을 쌓아 왔습니다. IBM Global Services는 최근 발표한 "모바일 기업으로 거듭나기 위한 효과적인 애플리케이션 전략 수립(Establishing an effective application strategy for your mobile enterprise)"[1] 문서를 통해 기업의 모바일 애플리케이션 계획, 개발, 배포 및 관리 시 유용하게 활용할 수 있는 IBM의 권장사항을 폭넓게 소개하고 있습니다. 이 문서에서는 위 문서 내용 중 모바일 비즈니스 애플리케이션 개발을 위한 포괄적 접근법이라는 한 가지 주제에 대해 집중적으로 다룹니다. 문서에서 설명할 기법들은 협업 기반 소프트웨어 라이프사이클 관리에 대한 베스트 프랙티스와 모바일 애플리케이션 작성에 고유한 최신 요구사항을 함께 설명합니다. 이 문서의 내용은 모바일 엔터프라이즈 비즈니스 애플리케이션 개발 프로젝트에 참여하는 다음을 비롯한 모든 역할에 도움이 되는 가치를 제공하고자 기획되었습니다. 모바일 프로젝트 기획 설계자, 구현에 대한 의사결정을 내리는 개발 팀, 프로젝트 활동 세부사항을 관장하는 프로젝트 관리자, 새로 개발한 애플리케이션을 검증하는 테스트 조직, 새로운 모바일 앱이 기존 엔터프라이즈 애플리케이션 및 개발 프로세스에 어떠한 도움을 주는지 이해가 필요한 경영진 등 다양한 역할이 이 문서를 통해 정보를 얻을 수 있습니다.

모바일 애플리케이션 개발을 위한 고유 해결 과제

스마트폰이나 태블릿과 같은 최신 모바일 디바이스에서 실행할 애플리케이션을 작성하려면 고유한 요구사항과 과제들을 해결해야 합니다.

폼 팩터 및 사용자 입력 기술

모바일 애플리케이션의 제1요소이자 가장 명확한 고유 특성은 디스플레이 및 사용자 상호작용을 위한 폼 팩터가 기존의 소프트웨어와는 크게 다르다는 것입니다. 스마트폰은 보통 4인치 화면 안에 애플리케이션 컨텐트를 표시해야 하고 PC 화면과 비교할 때 화소수가 적어 화면 해상도가 떨어지기 쉬운데, 이로 인해 현재는 화면 크기를 늘리고 화소 수를 늘리는 추세입니다. 태블릿 디바이스조차도 PC보다 화면 크기가 작으며, 최신 데스크톱 PC에 사용되는 대형 평면 스크린과 비교하면 그 차이는 더욱 커집니다.

폼 팩터가 작다는 것은 해당 애플리케이션에서 일반 사용자에게 표시되는 데이터의 양과 레이아웃이 PC에서 실행되는 앱과 달라야 한다는 것을 의미합니다. 한 번에 표시할 수 있는 데이터의 양이 훨씬 적기 때문에 꼭 필요한 데이터 그리고 애플리케이션에서 사용자가 그 시점에 가장 필요로 하는 데이터만 표시해야 합니다.

모바일 애플리케이션의 또다른 물리적 큰 차이점은 사용자 입력 메커니즘입니다. 모바일 디바이스는 키보드 대신 효과적인 동시에 대중적인 사용자 입력 방식인 손짓 또는 동작을 사용하도록 발전해 왔습니다. 모바일 애플리케이션의 만족스러운 사용자 경험을 위해 손가락을 사용한 터치, 살짝 밀기, 모으거나 펴기 동작들이 계획되고 지원되어야 합니다. 촉각을 이용한 이러한 일반 사용자 입력 메커니즘이 크게 대중화되고 각광받으면서 이제는 애플의 Lion OS X 릴리스 및 Windows 8 Metro OS와 같은 데스크톱 PC 시스템에까지 도입되고 있습니다. 촉각을 통한 사용자 입력 다음 단계로, 모바일 디바이스에서 음성 기반 사용자 입력을 지원합니다. 사실 모바일 애플리케이션 사용자에게 있어 기존의 키보드 입력 방식은 가장 비효율적이고 가장 소외된 메커니즘일 것입니다.

모바일 디바이스는 일반 사용자의 직접적인 입력 외에도 GPS 구성요소의 위치 기반 입력 및 디바이스에 기본 탑재된 카메라에서 보내는 이미지 정보와 같이 다른 소스로부터 입력을 수신하는 기능도 갖추고 있습니다. 모바일 애플리케이션을 설계 및 개발할 때는 이러한 고유 입력 방식을 고려해야 합니다. 이들은 입력 방식이 제한적인 애플리케이션보다 모바일 앱의 기능을 더욱 강력하고 효율적으로 만들기 위한 새롭고 가치 있는 메커니즘을 제공합니다.

유용성 및 사용자 상호작용 설계

모바일 애플리케이션에서 유용성 및 사용자 상호작용 설계에 더욱 세심한 주의를 기울여야 하는 이유에는 여러 가지가 있습니다. 그 중 하나는 폼 팩터와 사용자 입력 방법의 차이입니다. 사용 가능한 데이터를 모두 표시하고 일반 사용자가 직접 원하는 데이터로 이동할 수 있도록 하는 것보다, 정확히 필요한 데이터만 표시하는 방법을 계획하는 것이 훨씬 어렵고 시간이 많이 소요되는 작업입니다. 문서에 비유하자면, 전체 내용을 쓰는 것보다 간략한 요약을 작성할 때 더 어려움을 느끼는 것과 마찬가지입니다. 모바일 앱 설계자는 화면의 크기를 고려해야만 합니다. 애플리케이션에서 여러 계층의 세부사항이 포함된 넓은 범위의 데이터를 표시해야 하는 경우, 일반적으로 드릴 다운 기능을 통해 단계적으로 구체적인 세부 항목에 접근할 수 있는 점진적 검색 방법을 사용하는 것이 좋습니다.

모바일 디바이스에서 사용할 수 있는 다양한 입력 방법도 초기 설계 작업에서부터 입력을 지원하는 더욱 효율적인 방법을 찾아 적용해야 하는 동기 요인으로 작용합니다. 양식에 입력해 넣는 단순한 설계는 기존의 웹과 PC 애플리케이션의 약점으로 작용합니다. 터치식 키보드의 크기가 작기 때문에 일반 사용자의 불편을 줄이기 위해서는 모바일 앱에서는 키보드를 통한 다량의 입력은 피해야 합니다. 키보드 이외의 방법으로 정보를 수집하고 모바일 앱에 이를 전달하는 방법을 찾기란 설계 시 큰 어려움으로 작용합니다.

명확히 드러나지는 않지만 설계에 더욱 주의를 기울여야 하는 다른 이유는 모바일 애플리케이션 설계에 소요되는 노력입니다. 일반 자용자가 모바일 디바이스 및 디바이스에서 실행되는 애플리케이션과 상호작용하는 방법은 데스크톱 PC 혹은 노트북 컴퓨터에서의 상호작용 방법과 다릅니다. 일반적인 모바일 디바이스 사용자는 보통 손에 디바이스를 쥐고 있는 상태에서도 주변 환경과 의사소통을 멈추지 않습니다. 따라서 사용자의 생활 환경에 집중하다 보면 오랜 시간 동안 모바일 디바이스에만 주의를 기울일 수가 없습니다. 사용자와 모바일 앱의 상호작용은 짧은 시간 동안만 가능하고, 방해 받기 쉬우며 다른 일을 하기 전에 잠깐 동안 신속히 애플리케이션 작업을 완료할 수 있어야 합니다.

이러한 모든 요인들로 인해 모바일 애플리케이션 설계는 개발 프로젝트 초기 단계부터 사용자 중심의 투자가 필요합니다. 유용성에 대한 고려사항과 설계 측면을 모바일 애플리케이션의 요구사항으로 코드화한 다음 사용자 상호작용 및 앱 사용자 경험이 만족스러운지 검증하는 테스트를 거쳐 후반 단계인 개발 산출물로 연결하는 것이 가장 이상적이라 할 수 있습니다.

구현 기술 선택

모바일 애플리케이션 구현을 위한 선택의 폭은 매우 넓습니다. 모바일 애플리케이션 구현을 위한 하나의 완벽한 정답은 없으며 모든 방법에 장점과 단점이 공존합니다. 따라서, 모바일 개발 팀은 각종 기술 간의 장단점을 파악하고 고유한 애플리케이션 요구사항에 따라 취사선택하면 됩니다. 앞서 언급한 IBM Global Services 문서에는 여러 구현 기술에 대한 구체적인 설명과 이에 대한 비교 표가 나와 있습니다. 이 문서에서는 이를 보충할 수 있는 추가적인 고려사항을 소개합니다.

모바일 프로젝트를 위해 특정 구현 기술을 선택하면 애플리케이션 개발과 관련한 다른 의사결정에도 영향을 미칩니다. 개발도구의 선택에 제약이 따를 수도 있습니다. 구현 팀의 역할 및 구성에도 영향이 있을 수 있습니다. 이에 따라 애플리케이션 테스트 및 검증 방법, 일반 사용자에 대한 배포와 전달 방법 등도 달라지게 됩니다. 그러므로 모바일 애플리케이션 구현 방법의 선택은 초기 단계에 신중을 기해 판단해야 하는 매우 중요한 의사결정입니다.

네이티브 애플리케이션 구현

네이티브 구현이란 특정 디바이스 유형의 모바일 운영체제에서 사용되는 프로그래밍 언어와 프로그래밍 인터페이스를 통해 애플리케이션을 작성하는 것을 의미합니다. 예를 들어 iPhone용 네이티브 구현은 애플(Apple)에서 공급 및 지원하는 Objective-C 언어와 iOS 운영체제 API로 작성됩니다.

네이티브 애플리케이션 구현의 장점은 해당 모바일 디바이스에 최적화되어 있다는 것입니다. 사용되는 API의 수준이 낮고 해당 애플리케이션 전용 디바이스에 종속되기 때문에 해당 디바이스에서 제공하는 모든 기능과 서비스의 장점을 애플리케이션에서 최대한 활용할 수 있습니다.

모바일 애플리케이션의 네이티브 구현은 다른 모바일 운영체제로 전혀 이식이 불가능합니다. 네이티브 Apple iOS 앱을 안드로이드(Android) 디바이스에서 실행하려면 완전히 새로 작성해야 합니다. 따라서 이러한 구현 방법을 선택할 경우 모바일 비즈니스 애플리케이션 구현을 위해 많은 비용이 소요됩니다.

웹 애플리케이션

최신 스마트폰과 태블릿들은 고급 웹 브라우저가 미리 설치된 상태로 출시되므로 모바일 비즈니스 애플리케이션이 표준 웹 애플리케이션인 경우 쉽게 실행할 수 있으며, 특수 스타일 시트로 모바일 폼 팩터를 수용하고 모바일 디바이스의 룩앤필(Look & Feel)을 구현할 수 있습니다. JavaScript 및 HTML5를 대다수의 웹 브라우저에서 지속적으로 지원하기 때문에 이러한 방식으로 구현된 모바일 애플리케이션이 가장 폭넓은 모바일 디바이스를 지원합니다. 여러 가지 상용 및 오픈 소스 웹 2.0 라이브러리도 이 접근법에서 활용할 수 있습니다. 이미 웹 애플리케이션 개발에 필요한 언어와 기술에 능숙한 개발자를 보유하고 있는 기업이라면 웹 프로그래밍 모델을 통해 모바일 애플리케이션을 구현하는 것이 유리합니다.

웹 애플리케이션 구현의 경우 카메라, 주소록 등과 같이 모바일 디바이스에서 직접 실행되는 기능에는 액세스할 수 없다는 단점이 있습니다. 하지만, 디바이스에서 실행 중인 로컬 서비스에 독립적으로 작동하는 애플리케이션이라면 일반적인 웹

애플리케이션 구현 방식으로 충분합니다. HTML5 사양이 높은 수준이고 모바일 웹 브라우저에서 폭넓게 지원되므로 모바일 디바이스에서 로컬로 실행되는 대다수의 서비스는 W3C 프로그래밍 표준을 통해 웹 애플리케이션에서 이용할 수 있습니다.

웹 애플리케이션과 네이티브 애플리케이션 선택의 기로에서 고려해야 할 다른 요소는 애플리케이션을 디바이스에서 배포 및 활성화하는 방식입니다. 네이티브 애플리케이션은 공개적으로 액세스할 수 있는 Apple iTunes나 Google의 Android Marketplace와 같은 앱스토어에서 다운로드 후 설치해야 합니다. 앱스토어 배포 메커니즘은 검색 알고리즘을 통해 모바일 앱을 쉽게 찾을 수 있다는 장점이 있습니다. 따라서 기업은 모바일 앱스토어에서 제공하는 일반 사용자 피드백을 참고하거나 시장의 상황을 파악할 수도 있습니다.

앱스토어를 이용할 때의 불편한 점이라면, 특히 유명 앱스토어의 경우 기업과 기업의 목표 고객의 중간자 역할을 한다는 점을 들 수 있습니다. 모바일 애플리케이션을 업데이트하려면 항상 해당 앱스토어를 거쳐야 하므로 앱스토어 메커니즘을 통해 전달된 모바일 앱을 원격으로 제어하거나 관리하기가 어려울 수 있습니다. 반면 웹 애플리케이션은 앱스토어를 통해 배포되지 않습니다. 최종 사용자가 모바일 웹 브라우저에 원하는 애플리케이션의 웹 주소를 입력하면 인터넷을 통해 해당 애플리케이션이 제공됩니다. 모바일 웹 애플리케이션 업데이트는 앱을 호스트하는 서버의 업데이트만큼이나 간단합니다. 다음 번에 사용자가 웹 사이트에 액세스하면 새로운 버전의 모바일 앱이 디바이스에 다운로드됩니다.

하이브리드 모바일 애플리케이션 구현

하이브리드 모바일 애플리케이션 구현은 전형적인 네이티브 구현과 전형적인 웹 구현을 혼합 및 절충한 형태입니다. HTML5 및 JavaScript와 같은 산업 표준 웹 프로그래밍 언어와 기술을 통해 모바일 앱을 작성하지만 네이티브 설치 형식으로 앱을 패키징하여 앱스토어 메커니즘을 통해 배포합니다.

하이브리드 앱의 경우 추가적으로 네이티브 라이브러리에 연결되므로 해당 앱이 단일 애플리케이션 코드 기반으로부터 네이티브 디바이스 기능에 액세스할 수 있습니다. 특정 디바이스에만 적용되는 고유 기술을 사용하여 구현되는 하이브리드 애플리케이션은 소수에 지나지 않으므로 대부분의 애플리케이션 코드를 여러 다양한 모바일 운영체제에 이식해 재사용할 수 있습니다. 하지만 일부 네이티브 코드도 하이브리드 앱에 통합될 수 있습니다. 따라서 애플리케이션 구현 중 얼마나 많은 부분을 공유할지, 공통 코드 기반을 사용할지 및 디바이스별 사용자 지정의 규모는 어느 정도로 할지를 개발자가 결정해야 합니다.

또한 앱스토어를 통해 전달되는 네이티브 설치 가능 앱으로 패키징할 코드의 규모와 네트워크를 통해 다운로드할 규모도 선택할 수 있습니다. 앱에서 화면에 표시할 첫 번째 요소는 사용자의 앱 실행 시 신속한 로드를 위해 디바이스에 직접 설치되도록 패키징할 수 있습니다. 기타 동적인 요소들은 서버에서 관리되고 애플리케이션에 액세스할 때 항상 최신 버전이 제공되도록 웹 페이지로 구성할 수 있습니다.

다수의 업계 애널리스트들은 일반적인 수준의 모바일 비즈니스 애플리케이션에서 코드 재사용 및 유연한 애플리케이션 개발이 가져다 주는 경제적인 효과가 장기간에 걸쳐 절충형 하이브리드 접근법 확산에 기여할 것으로 확신합니다.

모바일 애플리케이션 빌드 및 전달

모바일 애플리케이션은 단시간 내에 시장에 출시해야 유리하기 때문에 대개의 모바일 개발 프로젝트는 극도로 빡빡한 일정으로 진행됩니다. 착수에서 출시까지 몇 개월 내에 완료되는 것이 일상적입니다. 모바일 앱을 신속히 출시해야 한다는 부담은 대부분의 모바일 프로젝트에서 민첩성이 뛰어난 개발 방식 도입으로 이어집니다.

신속한 개발 과정에서 중요한 요소는 지속적인 통합과 구축입니다. 개발자가 애플리케이션을 변경하면 해당 애플리케이션을 실행해야 하는 모든 모바일 운영체제에서 이같은 변경이 즉시 처리되어야 합니다. 하이브리드 또는 네이티브 구현 모바일 애플리케이션의 경우, 개발자가 애플리케이션 변경사항을 제공할 때마다 여러 애플리케이션 빌드를 매번 트리거해야 합니다. 빌드 설정 및 구성이 지원되는 각각의 모바일 환경마다 서로 다르며, 대개의 경우 이러한 여러 운영체제용 모바일 애플리케이션 빌드를 처리하기 위해 소규모 빌드 서버 팜(Farm) 프로비저닝 및 활성화할 필요가 있습니다.

테스트

모바일 애플리케이션 개발 시 넘어야 하는 또 하나의 산은 테스트입니다. 모바일 애플리케이션 테스트는 기존 애플리케이션에 비해 훨씬 복잡하고 많은 비용이 소요됩니다. 일반적인 PC 및 웹 애플리케이션과는 달리 지원되는 모바일 디바이스의 범위가 넓고 버전도 너무나 많습니다. 모바일 프로젝트의 테스트 매트릭스에는 디바이스, 모바일 OS 수준, 통신사, 지역, 디바이스의 가로/세로 방향 조합에 따라 수백 수천 개의 순열이 포함되는 것이 보통입니다.

모바일 테스트 환경에서 고려해야 하지만 다른 종류의 소프트웨어에는 해당되지 않는 변수들은 여기서 그치지 않습니다. 동일한 모델의 디바이스도 통신사 네트워크가 다르면 조금씩 다른 방식으로 작동하며 네트워크 연결 품질도 모바일 애플리케이션의 동작에 큰 영향을 미칠 수 있습니다. 몇몇 애플리케이션은 디바이스의 자원을 많이 소비하기 때문에 모바일 디바이스 자체의 성능도 애플리케이션 작동을 위한 중요 요소일 수 있습니다.

대다수의 모바일 앱에는 다중 계층 아키텍처를 채택하고 있어 디바이스에서 실행되는 코드와 기존의 중간 계층 및 데이터 센터 백엔드에서 지원하는 서비스가 사용됩니다. 효과적이고 포괄적인 모바일 앱 테스트를 수행하려면 모바일 디바이스의 코드뿐만 아니라 모든 애플리케이션 계층도 다루어야 합니다. 테스트 버전의 중간 계층 및 백엔드 서비스를 설정하고 가용성을 확보하려면 막대한 비용이 소요될 수 있으며 모바일 애플리케이션 테스트의 복잡성도 크게 증가할 수 있습니다.

모바일 프로젝트는 수작업을 통한 테스트부터 시작하는 경우가 많습니다. 테스트를 쉽게 시작할 수 있는 가장 확실한 방법이기 때문입니다. 그러려면 앱을 실행하려는 모든 모바일 디바이스를 구매하고, 개인 또는 팀 전체에 인건비를 지급해야 합니다. 결과적으로 각각의 애플리케이션 빌드에 대해 해당 모든 디바이스에서 테스트를 수행하는 방법을 선택하면, 엄청난 양의 지침에 따라 천천히 테스트가 진행될 것입니다. 수동 테스트는 극도로 비효율적이고 비용 소모적인 방법이지만 앱의 유용성 피드백 확보에 필요한 메커니즘을 제공하기 때문에 나름의 중요한 목적이 있습니다.

모바일 디바이스를 실제로 구입하는 대신, 모바일 디바이스 시뮬레이터와 에뮬레이터를 통해서도 테스트를 수행할 수 있습니다. 이 방법을 사용하면 데스크톱 워크스테이션에서 실행되는 소프트웨어 프로그램이 실제 디바이스를 대신합니다. 개발자 단말기 테스트와 같은 작업의 경우 모바일 애플리케이션 테스트 시에 시뮬레이터와 에뮬레이터를 사용하는 것이 더 효과적일 수 있습니다. 일부 에뮬레이터는 성능이 우수하지만 실제 디바이스 복제에 그다지 적합하지 않은 것들도 있기 때문에 수동 테스트에서나 자동 테스트에서 모두 일정 부분은 항상 실제 모바일 디바이스에서 수행해야 합니다.

디바이스에서 자동 실행 방식으로 테스트 스크립트가 상호작용할 수 있는 에이전트 프로그램 실행에 의존하는 모바일 앱 테스트 솔루션도 있습니다. 이 방법은 실제 디바이스나 에뮬레이터를 통한 테스트 모두에서 유연하게 사용이 가능하며 자동화의

효율을 높일 수 있습니다. 하지만 테스트 조직에서 테스트를 수행할 디바이스를 설정하고 해당 디바이스에 테스트 에이전트를 설치하기 위해 소요되는 비용을 감당해야 합니다.

디바이스 클라우드(Device Cloud)를 사용해서 모바일 앱 테스트를 수행하는 방법도 있습니다. 클라우드가 범용 컴퓨터 대신 실제 모바일 디바이스를 나타내는 자원을 노출할 수 있습니다. 테스트가 진행되는 몇 시간 혹은 며칠 동안 Linux 가상 시스템을 대여하지 않고 특정 모델 및 릴리스의 모바일 디바이스를 대여할 수 있습니다. 이 방법을 사용하면 수백 대에 달하는 디바이스 구매 비용과 테스트를 위해 이 디바이스를 관리하는 비용을 아낄 수 있습니다.

모바일 개발과 다른 소프트웨어 개발의 유사점

모바일 애플리케이션만의 고유한 특징이 있긴 하지만 전체 개발 라이프사이클에서 보면 다수의 역할과 작업이 다른 엔터프라이즈급 소프트웨어 개발과 동일합니다. Walker Royce의 "Measuring Agility and Architectural Integrity"[2] 문서에는 민첩하고 테스트 위주의 원칙을 적용하여 효과적으로 소프트웨어를 출시하기 위한 핵심 기법과 과정이 설명되어 있습니다. Walker의 문서에 언급된 소프트웨어 출시 과정은 모바일 개발 프로젝트에 아주 적합합니다.

프로젝트 전체 라이프사이클

소프트웨어 개발 프로젝트의 라이프사이클은 일반적으로 작성 중인 소프트웨어의 종류에 관계없이 패턴이 유사합니다. 가장 먼저 분석을 기반으로 애플리케이션 투자 의사 결정이 이루어집니다. 애플리케이션에 대한 요구사항을 파악하고 세부적으로 발전시킵니다. 애플리케이션 요구사항은 추후에 사용자 시나리오와 기능 작업 항목으로 구분해 반복이 가능하도록 작업 계획을 수립한 다음 애플리케이션 출시를 위해 완료하여 릴리스됩니다. 팀 구성원들이 작업 항목에 다양한 역할로 지정되고 여러 도구를 사용하여 작업을 완료한 후 프로젝트를 구성하는 작업 결과를 전달합니다. 작업의 결과로 작성된 애플리케이션을 테스트하여 요구사항이 충족되었는지 검증합니다. 회사마다 소프트웨어 프로젝트의 세부 프로세스 및 라이프사이클은 특정 기업의 목표와 정책에 맞춰 조정됩니다.

이러한 라이프사이클은 모바일 애플리케이션에서도 마찬가지입니다. 모바일 애플리케이션 개발의 특징은 대부분 소규모 팀에서 기존 인프라 및 강력한 대화식 애플리케이션을 사용하여 수행된다는 것입니다. 이러한 시나리오에서는 민첩한 방법과 테스트 위주의 원칙이 가장 이상적입니다. 모바일 앱 개발을 위한 세부적인 요구사항이 다른 소프트웨어 개발과 다를 수는 있지만 요구사항 수집, 검토, 조정을 위한 도구와 프로세스는 동일합니다. 이러한 요구사항 구현을 지원하기 위해서는 코드를 변경해야 한다는 필요성도 다른 소프트웨어와 모바일 애플리케이션 개발의 공통점입니다. 다시 말해 프로젝트의 흐름과 프로젝트 전반에 걸친 통합과 추적 필요성은 모든 개발 프로젝트에서 동일합니다.

다양한 도구의 통합

한 가지 개발도구로 소프트웨어 개발을 완료할 수 있는 프로젝트는 거의 없습니다. 대부분의 프로젝트는 프로젝트 전체 라이프사이클 동안 특정 역할 또는 작업을 수행할 수 있도록 설계된 여러 벤더의 다양한 도구를 사용합니다.

예를 들어 모바일 애플리케이션의 코드를 개발하는 개발자는 애플리케이션을 실행할 모바일 플랫폼에 맞는 간단한 코드 설계 도구를 사용하면 충분히 작업 수행이 가능하다고 생각합니다. 하지만 이 도구는 애플리케이션 개발에 참여한 개발 팀의 협업과 업무 조율에 필요한 기능은 지원하지 않습니다.

개발자용 코드 설계 도구를 호환 가능한 협업 기반 팀 개발 플랫폼에 통합하면, 민첩하게 움직이는 팀의 효율성과 품질을 한층 개선할 수 있습니다.

팀내 및 팀간 협업의 필요성

모바일 애플리케이션은 통상 서로 다른 스킬과 전문 지식을 보유한 소규모 팀에서 개발합니다. 일반적인 팀 구성은 기본 비즈니스 로직 및 웹서비스 개발자 1~2명, UI 개발자 1~2명, 사용자 경험 디자이너 1명, 소수의 테스트 담당자 그리고 팀장 또는 관리자로 이루어집니다.

일반적으로 모바일 앱 출시 일정은 매우 빠듯하기 때문에 소규모 팀이라도 효율성을 극대화하여 운영해야 합니다. 팀 구성원 사이에 이해나 소통에 문제가 생겨 지연이 발생하면 전체 일정에 차질이 발생합니다.

여러 모바일 운영체제에서 지원되는 모바일 애플리케이션 프로젝트에서는 코드 공유 및 재사용이 필요합니다. Android 전문 개발자와 iOS 전문 개발자가 따로 있을 수 있습니다. 팀 구성원들은 수행할 작업과 완료 시점에 대해 명확히 이해하고 있어야 합니다. 프로젝트 요구사항, 진행 일정, 계획 등은 공유하고 소스 코드, 테스트, 빌드만 다를 수 있습니다.

통합된 변경 관리 및 소프트웨어 버전 관리

특정 작업 항목에 관한 모든 코드 변경은 특정 변경 세트(Change Set)로 한데 모으거나 한꺼번에 제공되는 변경된 소스 코드 파일 목록으로 묶어 전체 코드 변경을 하나의 단위로 추적할 수 있어야 합니다. 이상적인 환경에서는 개발자의 집중력을 분산시키거나 로직 개발에 방해가 되지 않도록 변경 세트를 모으는 프로세스가 최대한 자연스럽고 원활하게 진행됩니다.

버전 관리 및 병합/롤백 프로세스도 직관적이고 자동화되어 있습니다. 개발자가 작업 수행을 위해 업무 내용을 전환해야 할 경우가 발생하면 이 때가 개발 프로세스의 중단 지점 및 잠재적 속도 조절 지점이 됩니다.

프로젝트 전반에서 추적의 필요성

소프트웨어 개발 시 민첩한 팀 접근 방식은 보통 소규모 애플리케이션 기능 향상을 구현 및 검증하는 과정이 짧게 여러 번 반복되도록 정의하는 것입니다. 일반적으로 2~4주 단위를 민첩한 반복으로 간주합니다. 팀의 리더는 팀원들과 함께 백로그 목록에 있는 작업 항목을 특정 반복으로 맵핑하고 이러한 작업 항목을 개별 개발자에게 지정하는 작업을 수행합니다.

개발자가 작업 항목을 접수하고 진행을 시작하면 자동으로 기록이 이루어져야 팀장이 이를 추적 및 조회할 수 있어야 합니다. 자동으로 정보가 기록되면 대시보드를 통해 이미 완료된 항목, 현재 작업 중인 항목, 앞으로 작업이 필요한 항목에 대해 쉽게 조회하고 추적할 수 있습니다. 팀 구성원 모두가 반복이 진행되는 과정과 반복을 위해 계획된 작업 항목의 상태를 확인할 수 있어야 합니다.

팀의 테스트 담당자가 모바일 애플리케이션 기능 테스트를 시작하면 공유된 개발 프로젝트에서 테스트 과정 중에 발견한 결함에 대한 작업 항목을 작성해야 합니다. 실패한 테스트 케이스가 프로젝트 계획의 특정 변경 세트 또는 기능 항목에 연결되면, 변경된 코드 및 해당 테스트 케이스 실패와 관련이 있어 보이는 코드에 대한 정보가 결함 데이터에 자동으로 입력될 수 있습니다. 또한 변경 집합이 코드 변경을 유발한 최초 요구사항에 연결되면 최초 요구사항에서부터 해당 요구사항이 애플리케이션에 반영되었는지 검증하는 테스트 케이스에 이르는 전체 프로젝트 라이프사이클에 걸쳐 추적이 가능합니다.

이러한 총체적 프로젝트 보기(Whole Project View) 및 엔드 투 엔드 추적성은 모든 유형의 소프트웨어 개발 프로젝트에서 매우 중요하지만, 그 중에서도 짧은 일정을 소화하고 민첩한 개발 방법을 도입하는 모바일 애플리케이션 개발 팀에게 특히 더 필요합니다. 소규모 개발 팀은 특정 요구사항이 검증되고 적용되었는지 여부 및 적용 시기에 관한 세부적인 사항까지 추적할 시간적 여유가 없습니다.

모바일 애플리케이션 개발 솔루션 선택

모바일 엔터프라이즈 애플리케이션 개발을 위한 IBM의 접근법에는 일반적인 엔터프라이즈 소프트웨어 개발 프로세스 분야에서 수년간 축적된 경험과 모바일 디바이스 및 이를 위한 소프트웨어 기반을 지원하는 최신 도구와 기술들이 한데 녹아 있습니다.

각 업종별 엔터프라이즈 소프트웨어의 설계 및 배포에 관한 광범위한 전문지식과 기술을 보유한 IBM은 모바일 애플리케이션 프로젝트 개발 필요성 충족을 위한 맞춤형 솔루션을 지원합니다. IBM 내에서도 혁신을 통해 오랜 기간 수만 명의 개발자가 참여해 온 수천 건의 프로젝트에서 민첩한 방법들을 도입해 왔습니다.

총체적 프로젝트 가시성을 위한 협업 기반 라이프사이클 관리(CLM)

모바일 애플리케이션 출시 라이프사이클 전반에서 통합이 진행되지 않는다면, 개발 팀은 따로 고립되어 다른 구성원과는 단절된 채 운영될 것입니다. 고립과 단절이 형성되면 제품 출시가 능률적으로 진행될 수 없습니다. 변화하는 시장의 요구사항과 표준에 대처할 수 있는 강력한 모바일 엔터프라이즈 애플리케이션 솔루션을 출시하려면, 소프트웨어 엔지니어링 팀이 작업을 효율적으로 진행하고 전체 라이프사이클 동안 협업을 통해 작업 산출물을 관리해야 합니다.

모바일 비즈니스 애플리케이션 개발용 IBM Rational 솔루션은 협업 기반의 작업을 지원하고 제품 라이프사이클 과정에서 개발된 다양한 아티팩트 연결에 도움이 되는 통합 라이프사이클 관리 플랫폼을 제공합니다. 이 솔루션은 또한 출시 워크플로우의 진행을 지원하며, 모바일 애플리케이션 개발 프로젝트의 효과적인 실행에 도움이 되는 작업 관리 기능을 제공합니다. 프로젝트 라이프사이클 중 모바일 전용 기능이 필요한 시점에 해당 기능을 통합하여 확장이 가능합니다.

IBM Rational Team Concert 소프트웨어를 포함한 민첩한 팀 협업 기반 개발도구를 사용하면 전체 프로젝트 내에서 여러 단기적인 반복을 정의할 수 있으며, 프로젝트 백로그에서 특정 반복 계획으로 손쉽게 작업 항목을 이동할 수 있습니다.

개발자가 소프트웨어 버전 관리 시스템에 통합된 모바일 애플리케이션 코드 개발도구 내에서 파일을 편집하면, 변경 세트의 자동 업데이트 및 유지관리가 가능합니다. 개발자는 작업 진행에 필요한 파일을 편집하는 일 외에, 변경 세트를 만드는 작업을 직접 수행할 필요가 없습니다.

변경 세트를 주요 코드 스트림에 완전히 통합하기에 앞서 팀 구성원 간에 이를 공유할 수 있습니다. 따라서 웹서비스 개발자가 웹서비스에서 제공하는 데이터 형식을 변경함으로써 생성한 변경 세트를 새로운 데이터 표시 로직에 관한 작업을 수행하는 UI 개발자와 공유할 수 있으며 팀의 다른 구성원 업무에는 영향을 미치지 않습니다. UI 코드 변경과 웹서비스 코드 변경 사이의 일치가 이루어져 준비가 완료되면, 주요 코드 스트림에 하나의 동기화된 작업으로 통합이 가능하고, 팀의 나머지 구성원이 이를 가져와 사용할 수 있습니다.

IBM 모바일 엔터프라이즈 전략으로 모바일 애플리케이션 런타임 제공

모바일 애플리케이션 전용 코드 개발도구와의 통합을 통한 협업 기반 라이프사이클 관리 플랫폼 조합은 IBM의 포괄적 모바일 애플리케이션 솔루션에 포함된 중요 구성요소입니다. 하지만 일부 엔터프라이즈급 모바일 애플리케이션 개발 시에 해결해야 하는 일부 과제들은 개발도구 및 사례만으로는 해결할 수

없습니다. 단일 공통 모바일 애플리케이션 프로그래밍 모델용 기능을 지원하기 위해 IBM은 IBM Mobile Enterprise 소프트웨어 런타임을 제공합니다.

IBM 모바일 비즈니스 애플리케이션 개발 솔루션에서는 모바일 도구를 갖춘 IBM Rational Collaborative Lifecycle Management 를 통한 강력한 팀 개발 기능과 IBM Mobile Enterprise 솔루션에서 제공하는 런타임 기능이 결합되어 있습니다.

포괄적 테스트 접근법

포괄적인 다중 계층 모바일 애플리케이션 테스트는 둘 이상의 테스트 실행 기능을 사용하여 이를 하나의 애플리케이션 품질 테스트 결과로 통합해야 한다는 의미입니다. IBM Rational Quality Manager 소프트웨어는 모바일 테스트에 필요한 다양한 테스트 실행 엔진을 하나로 묶어 관리할 수 있는 탁월한 제품입니다. 이 테스트 환경에서는 라이프사이클 초기의 큰 문제점들을 해결하고 비용 절감 효과를 얻을 수 있도록 통합 테스트를 우선 수행하는 방식으로 진행됩니다. 가장 어려운 문제를 먼저 테스트하기 위해 필요한 기법에 대한 자세한 내용은 Walker Royce의 "Measuring Agility and Architectural Integrity" 문서에서 설명합니다.

앞에서도 설명했지만, 현재 시장에 다양한 모바일 앱 테스트 및 검증 기술이 출시되어 있습니다. 효과적인 개발 프로젝트의 관리자라면 각 기술마다 장단점이 있기 때문에 모바일 애플리케이션 개발 시에 적용 가능한 모든 기술을 도입해볼 것입니다. 모바일 앱 테스트에 완벽한 하나의 정답은 없으며 사용 가능한 여러 가지 기술들 중 하나를 도입하면 나머지 다른 기술을 사용할 수 없는 것은 아닙니다. 모든 형태의 모바일 테스트 수행 간에 균형을 유지하고 각 테스트 결과를 포괄적인 전체 모바일 애플리케이션 품질 기준(Quality Metric)으로 통합할 수 있는 전략이야말로 가장 효과적인 테스트 전략일 것입니다.

다음은 IBM에서 지원하는 모바일 앱 테스트 기법의 예를 소개한 것입니다.

수동 테스트

수동 테스트는 현재 업계에서 사용되는 가장 일반적인 모바일 테스트 접근법입니다. 하지만 동시에 가장 오랜 시간이 소요되고, 오류 발생 가능성이 높으며, 비용도 많이 드는 테스트 기술이기도 합니다. 수동 테스트 케이스를 구성하고, 테스트 담당자가 테스트를 수행할 수 있도록 안내하고, 테스트 결과를 저장할 수 있는 솔루션이 있을 경우 비용이 크게 절감될 수 있습니다. Rational Quality Manager 소프트웨어가 이러한 기능을 제공합니다.

에뮬레이터 및 시뮬레이터

에뮬레이터는 모든 네이티브 모바일 운영체제 개발 키트와 함께 제공됩니다. 시뮬레이터의 경우 IBM을 포함한 여러 출처를 통해 얻을 수 있으며, IBM은 IBM Mobile Enterprise 솔루션용 개발도구의 일부로 모바일 시뮬레이터를 제공합니다.

프로토콜 가상화 및 애플리케이션 계층 분리

모바일 애플리케이션의 구조는 여러 계층으로 이루어져 있기 때문에 모바일 디바이스에서 코드 실행 테스트를 지원하는 인프라 설정 프로세스에 많은 시간과 비용이 소요될 수 있습니다. IBM Rational Green Hat 솔루션을 도입하면 비용과 개발 지연을 최소화할 수 있습니다. 모바일 디바이스에서 실행되는 코드에 대한 테스트를 지원하는 복잡한 미들웨어 환경을 테스트 팀에서 설정할 필요가 없습니다. Rational 솔루션이 중간 계층과 백엔드 서비스 그리고 프로토콜을 대신 수행하므로 테스트 팀은 디바이스 자체에서 실행되는 모바일 앱의 클라이언트 계층 테스트에 집중할 수 있습니다.

디바이스에 내장된 도구 및 에이전트

근본적인 요구사항은 수동으로 이루어지는 기능 검증을 모바일 디바이스에서 수행되는 자동 코드 테스트 형식으로 바꿔야 한다는 것입니다. 여러 벤더에서 이를 위해 다양한 접근법을 시도해 왔습니다. 가장 전형적인 방법은 디바이스에 자동

테스트가 가능하도록 일종의 추가 코드를 배치하는 것입니다. 이 코드가 디바이스에 내장된(On Devicd) 로컬 에이전트의 역할을 수행하면서 사용자 입력을 애플리케이션에 자동으로 전달하고, 이 입력의 결과로 발생하는 애플리케이션의 동작을 모니터링합니다. 이렇게 모바일 기능 테스트를 자동화하는 기술은 이 문서에 소개된 다른 기술의 보완에 많은 도움이 되며, 다른 기술과 결합하여 매우 효과적으로 사용할 수 있습니다.

디바이스 클라우드

각종 모바일 테스트 디바이스의 다양한 조합을 모두 구매, 설정, 관리한다는 것은 아무리 충분한 예산이 확보된 프로젝트라도 불가능에 가까운 일입니다. 이 문제를 해결할 수 있는 기술이 바로 디바이스 클라우드 테스트 솔루션입니다. 이 접근법을 도입하면 현재 출시되어 있는 수많은 종류의 디바이스를 소유하는 비용을 효과적으로 절감할 수 있으며 모바일 앱이 출시된 이후에 사용자가 도입을 고려할 수 있습니다. IBM은 총체적인 모바일 테스트 관리 솔루션인 Rational Quality Manager와 디바이스 클라우드를 지원하는 여러 비즈니스 파트너를 통합할 수 있는 옵션을 제공합니다.

결론

산업 분야를 막론하고 기업 비즈니스 애플리케이션의 모바일 버전에 대한 필요성이 커질수록 모바일 앱 개발에 대한 엔터프라이즈급 접근 방식이 필요합니다. IBM은 이미 이러한 접근 방법을 보유하고 있습니다.

모바일 애플리케이션 개발을 위한 IBM의 접근법은 다음과 같은 5가지 항목을 중요시합니다.

- 모바일 앱 사용자 경험 간소화
- 모바일 앱 프로젝트의 경제성 향상을 위한 통합 우선 방식
- 요구사항 검증을 위한 테스트 시 해당 요구사항에 대한 추적성 보장
- 극도로 민첩한 방법 도입
- 신속한 개발과 변경 비용 절감을 위해 자동화된 회귀 테스트 모음(Regression Test Suites) 활용

이러한 접근법을 통해 다른 엔터프라이즈 애플리케이션 개발과 동일한 효율성 및 엄격한 기준을 그대로 모바일 도구 및 기술에 적용할 수 있습니다.

저자 정보

Leigh Williamson은 1988년부터 미국 Texas의 Austin 연구소에서 근무 중인 IBM 책임 엔지니어(Distinguished Engineer)로, OS/2, DB2, AIX, OpenDoc, Java, Component Broker 및 WebSphere Application Server를 비롯한 IBM의 주요 소프트웨어 프로젝트에 참여해 왔습니다. 현재는 IBM Rational Software Chief Technology Officer 팀의 일원으로써 Rational 브랜드 제품의 전략적 방향을 설정하고 모바일 애플리케이션 개발을 위한 솔루션 정의를 주도해 나가고 있습니다. Leigh는 Nova University에서 컴퓨터 사이언스 분야 학사학위와 University of Texas에서 컴퓨터 엔지니어링 분야 석사 학위를 취득했습니다.

추가 정보

모바일 애플리케이션 개발용 IBM Rational에 솔루션에 대한 자세한 내용은 IBM 담당자 또는 IBM 비즈니스 파트너에게 문의하거나 다음 웹사이트를 방문하시기 바랍니다. **ibm.com**/software/solutions/mobile-enterprise/

추가적으로, IBM Global Financing은 가장 비용 효율적 방법과 전략적 방법으로 비즈니스에서 필요로 하는 IT 솔루션을 취득할 수 있도록 도와줍니다. IBM은 신용 있는 고객과 협력하여 귀사의 비즈니스 목표에 적합하고 효과적인 현금 관리를 가능하게 하며 귀사의 총소유 비용을 개선하는 맞춤형 IT 재무 솔루션을 제공합니다. IBM Global Financing은 중대한 IT 투자에 자본을 투입하고 귀사의 비즈니스를 발전시키는 가장 현명한 선택입니다. 자세한 정보는 다음 웹사이트를 참조하십시오. **ibm.com**/kr/financing

IBM Corporation
Software Group
Route 100
Somers, NY 10589

Produced in the United States of America
2012년 4월

[1] IGS 문서, "Establishing an effective application strategy for your mobile enterprise," **ibm.com**/common/ssi/cgi-bin/ssialias?subtype=WH&infotype=SA&appname=GTSE_EN_OS_USEN_C&htmlfid=ENW03007USEN&attachment=ENW03007USEN.PDF

[2] Walker Royce의 "Measuring Agility and Architectural Integrity" 백서, www.ijsi.org/ijsi/ch/reader/view_abstract.aspx?file_no=i92

재활용하십시오

RAW14302-KRKO-00