새로운 혁신 구현

개발자를 위한 2019년 10대 예측

저자: Siddhartha Agarwal

developer.oracle.com

ORACLE

1 2020년경에는 사용자 정의 애플리케이션 10개 중 8개가 임베디드 인텔리전스를 최고의 기능으로 제공할 것입니다.

따라서 개발자는 데이터 사이언스의 원리를 이해해야 합니다. 데이터 사이언티스트가 되지는 않겠지만, 어떤 개발자는 코드를 쓰고, 어떤 개발자는 수학 및 모델을 고안합니다. 그러나 개발자는 점차 데이터 사이언스 방법론을 이해하고 데이터 사이언스 모델을 워크플로에 통합해야 할 것입니다. 데이터는 소프트웨어를 더욱 지능적으로 만들어 소프트웨어가 머신 러닝을 통해 결과를 예측하거나 사용자 요구를 예상할 수 있는 기능을 제공합니다. 따라서 개발자는 혁신적인 기술을 개발하기 위해 데이터 사이언티스트와 더욱더 긴밀히 협업해야 합니다. 개발자는 API를 통해 데이터 사이언티스트의 모델을 노출하고 이 모델을 도메인 특정 앱에 포함하여 실제로 변화를 구현할 수 있습니다. 소매업자가 전자상거래 주문을 이행할 오프라인 상점을 지능적으로 결정하려는 경우를 생각해 보겠습니다. 데이터 사이언티스트는 어느 상점에서 배송하는 것이 가장 적합한지 계산하는 모델을 생성할 수 있습니다. 예를 들어 추운 지역에서 매장을 방문한 쇼핑객이 구입할 가능성이 높은 스웨터보다는 따뜻한 지역의 상점에서 팔려나가지 않을 가능성이 높은 스웨터를 회사가 배송하는 것이 좋을 것입니다. 개발자는 이런 종류의 인텔리전스를 주문이행 앱에 끌어와서 올바른 결정을 내릴 수 있도록 직원에게 제공할 수 있습니다.

2 개발자는 플랫폼 엔지니어와 협력해야 하며, 이 파트너십이 클라우드 네이티브 개발을 위한 새로운 주요 역할로 부상할 것입니다.

이 새로운 플랫폼 엔지니어 역할을 클라우드용 시스템 관리자 같은 것이라고 생각해 보십시오. 플랫폼 엔지니어는 개발자가 클라우드 네이티브 애플리케이션을 구축 및 배포할 수 있는 기반이 되는 높은 성능의 클라우드 인프라를 확보하는 데 필요한 모든 리소스를 집결합니다. 플랫폼 엔지니어는 컨테이너를 오케스트레이션할 Kubernetes를 채택하는 방법, Istio 및 Envoy 프록시 같은 도구를 사용하여 서비스 메시를 실행하는(마이크로서비스 애플리케이션의 복합적인 서비스-서비스 통신에 점차 중요해지고 있음) 방법과 같은 요소를 이해할 것입니다. 또한 보안과 가용성은 모두 엔터프라이즈 내에서 대규모로 컨테이너를 사용하는 데 있어 매우 중요하기 때문에 컨테이너의 보안 및 서비스 사이의 통신의 보안을 보장하는 데에도 초점을 둘 것입니다. 개발자는 여전히 클라우드의 이점도 활용합니다. 즉, 컨테이너에서 개발 및 배포할 수 있으며 클라우드는 수요에 따라 앱을 서비스하기 위해 리소스를 확장 및 축소합니다. 그러나 이러한 모든 인프라를 끌어와 개발자가 사용할 수 있게 준비하는 것은 플랫폼 엔지니어입니다.

3 보안 요구와 성능 요구 사이의 균형을 위해 애플리케이션 배포에 있어 가상 머신과 컨테이너의 중간의 하이브리드 모델의 인기가 높아질 것입니다.

작년에는 보안이 잘못되면 다른 아무것도 중요하지 않다는 것을 다시 한번 개발자에게 상기시킨 심각한 보안 위협이 발생했습니다. 이에 대응하여, 올해에는 컨테이너에서 실행하는 최고의 성능과 효율성을, 그것도 멀티 테넌트 환경에서 더 나은 보안과 격리를 보장하는 초경량 가상 머신으로 가능하게 하는 애플리케이션 배포 접근 방식을 찾아보십시오. Kata 컨테이너가 이 아이디어에 대한 유효한 구현을 제공하지만, 이와 비슷한 개념이 다른 곳에서도 탄생할 것입니다. 클라우드 서비스 공급자가 이러한 혁신을 촉진하지만, 개발자는 이 옵션에 대해 이해하고 이 옵션이 가져올 수 있는 성능상의 손해와 이득이 무엇인지 파악해야 합니다.

4 서버리스의 경제학이 무시할 수 없는 수준으로 아주 강력해져서 다중 프런트의 서버리스 혁신을 촉진합니다.

우선, 서버리스에 대한 개방형 표준이 명확한 형태를 가지게 되어 개발자는 오늘날의 다수의 서버리스 옵션이 가진 종속 위험 없이 Kubernetes처럼 서버리스를 수용하게 될 것입니다. 그 형태가 정확히 어떻게 될지(단일 주도 플랫폼 또는 일반적인 표준 기반 구성 요소가 포함된 다수의 플랫폼)는 예측하지 못하지만 내년 이 시기 즈음이면 혁신적인 결과물을 확인할 수 있을 것입니다. 둘째, 서버리스 솔루션은 오늘날의 함수 위주의 솔루션(Oracle Fn, AWS Lamba, Azure Functions)을 Apache Spark, Flink 및 Kafka 에코시스템 기반의 대규모 데이터 처리를 위한 클라우드 우선의 서버리스 인프라와 같은 다른 비즈니스 도메인으로 신속하게 확장할 것입니다. 클라우드에서 빅 데이터 실행과 마찬가지로, 경쟁력 있는 경제성이 확보됩니다. 전체적으로, 올해에는 추가 플랫폼 종속성을 추상화하는 방식으로 서버리스에 크게 의존해 개방형 클라우드 개발을 가속화하는 최신 개방형 클라우드 아키텍처를 찾아보십시오.

5 봇이 급증하면서, 개발자는 작업에 적합한 봇을 찾는 '도우미'에 눈을 돌립니다.

개발자는 주문을 받거나, 인벤토리를 검색하거나, 배송 일정을 정하는 등 각각 모종의 기술을 표현하는 다양한 봇을 작성할 것입니다. 그러나 사용자가 모든 작업에 적합한 봇을 치밀하게 찾아낼 것이라고 기대할 수는 없습니다. 대신, 사용자는 가능한 각 봇을 살펴보고 작업을 완료하는 데 필요한 봇을 채용할 수 있는 지능형 인터페이스, 즉 디지털 도우미가 필요합니다. 예를 들어 HR 앱에 들어가는 경우 비용 보고서를 파일 처리하겠다고 디지털 도우미에게 알리면 이 도우미가 봇 자동화를 호출해 파일 처리를 실행합니다. 대화형 인터페이스의 인텔리전스는 계속해서 성장하고 있으며, 지능적이지 못한 대화형 인터페이스는 사라질 것입니다. 주문 받기 또는 인벤토리 검색과 같은 특정 사용 사례의 경우 지능형 인터페

6 개발자는 개방성을 기준으로 클라우드를 선택하고 클라우드 종속을 거부합니다.

개발자는 IT 운영이 아니라 창조에 집중할 수 있도록 하기 위해 클라우드 인프라로 이동합니다. 구축 기반이 될 클라우드를 선택할 때 개발자는 다양한 언어와 데이터베이스 및 컴퓨팅 형태에서 선택하기를 바랍니다. 그리고 워크로드를 임의의 클라우드로 이동할 수 있기를 원합니다. 클라우드 사이에 이동할 수 있는 능력 또는 분기되지 않는 브랜치에서 새로운 혁신을 채택할 수 있는 능력을 제한하는 분기된 오픈 소스에 대해 개발자가 조심하는 경향이 높아지고 있습니다. 이 때문에 개발자가 클라우드 종속을 피하는 클라우드를 선택할 때 오픈 소스 및 개방형 표준이 중심적인 역할을 차지합니다.

7 개발자는 하나의 클라우드로 충분하지 않다고 판단합니다.

초기의 클라우드는 단일 공급자에 의존했던 경우가 많은 반면, 현재의 확장은 특히 다양한 SaaS 애플리케이션을 사용할 수 있으며 다수의 공급자 사이에 워크로드를 분산하는 멀티 클라우드 전략이 될 것입니다. 클라우드로 이동은 단계별 여정이며, 많은 온-프레미스 워크로드는 앞으로 몇 년 동안은 유지될 것입니다. 개발자는 클라우드 및 온-프레미스 배포에서 동일한 도구 및 워크플로를 요구할 것입니다. 지금 온-프레미스 HR 앱에서 데이터를 끌어오는 API가 내년에는 Saas(Software as a Service) 앱에서 같은 데이터를 끌어와 퍼블릭 클라우드에서 실행하는 고객 애플리케이션에 데이터를 라우팅할 수도 있습니다.

8 자율 데이터베이스는 개발자가 시장 규모에 맞춰 애플리케이션을 가속화할 수 있게 합니다.

개발자는 신속한 프로토타입 생성 그 이상을 원합니다. 이 프로토타입이 목표에 도달했을 때 이 결과물을 최대한 빠르게 최대한 많은 사람이 사용하게 되기를 원합니다. 이 사용 규모가 성공을 검증하며, 개발자는 애플리케이션을 개선하기 위한 피드백을 얻을 수 있습니다. 클라우드 기반의 자율 데이터베이스는 속도와 규모 모두를 가능하게 합니다. 자율 데이터베이스를 활용하면 개발자가 데이터베이스 확장, 패치, 프로비저닝 또는 튜닝에 대해 생각할 필요가 없습니다. 대신, 개발자는 요구를 충족하기 위해 자동으로 확장되는 데이터 저장소를 확보한 상황에서 앱을 빌드하는 데만 집중할 수 있습니다. 개발자는 DBA의 도움 없이 몇 분 안에 고성능 데이터베이스를 시작할 수 있으므로 생산성을 얻습니다. 규모는 클라우드에서 비롯되어, 비즈니스는 필요한 용량을 얼마든지 늘릴 수 있으며 데이터베이스의 자체 튜닝 및 보안을 통해 효율적으로 실행합니다.

9 레거시 엔터프라이즈 애플리케이션이 클라우드 네이티브 개발 접근 방식으로 이동합니다.

이제, 클라우드 네이티브가 새로운 사내 개발 프로젝트를 위한 기본 옵션이라는 것이 명확해졌습니다. 하지만 더 어려운 점은 기술 스택의 80% 이상이 레거시이며 클라우드 네이티브가 아니라는 점입니다. 클라우드 네이티브 앱과 동일한 방식으로 배포, 실행 및 관리할 수 있도록 레거시 엔터프라이즈 앱을 현대화하고 컨테이너화하는 것을 가능하게 만드는 새로운 기술과 전술을 찾으십시오. 엔터프라이즈 앱에 클라우드 네이티브 접근 방식을 적용하는 두 가지 구체적인 예가 있습니다. 하나는 Kubernetes에서 WebLogic 도메인을 설정할 수 있는 새로운 운영자이고, 또 하나는 Java 애플리케이션을 마이크로서비스로 빌드하여 Java 모놀리스를 현대화하는 프레임워크인 Helidon입니다. 이러한 추세는 경력 많은 개발자에게 반가운 소식입니다. 이들은 자신이 보유한 기술에 기반하여 앱을 빌드하여 컨테이너 기반의 클라우드 네이티브 환경으로 확장할 수 있기 때문입니다.

10 규제산업 분야의 개발자에게 블록체인이 대세입니다.

외부 소스에서 데이터를 가져오는 애플리케이션을 만드는 개발자는 특히 식품 및 농업, 제약 또는 금융과 같은 규제산업에서 사용되는 애플리케이션의 경우 곧 이 데이터 중 일부를 블록체인 원장에서 쿼리하게 될 것입니다. 클라우드 애플리케이션 개발자는 이제 앱 외부에서 블록체인 클라우드 서비스를 호출하는 기능을 포함할 수 있습니다. 블록체인을 클라우드 서비스로 사용하면 개발자는 성능과 확장성을 보장하는 등 블록체인 기술 구현의 복잡성에 대해 걱정할 필요가 없습니다. 왜 데이터베이스가 아니라 블록체인일까요? 블록체인은 공급업체, 배송업체 등의 타사에서 나온 데이터와 관련하여 신뢰성과 투명성을 제공할 수 있으며, 이러한 신뢰할 수 있는 데이터는 규정 준수를 더 수월하게 만들 수 있습니다.