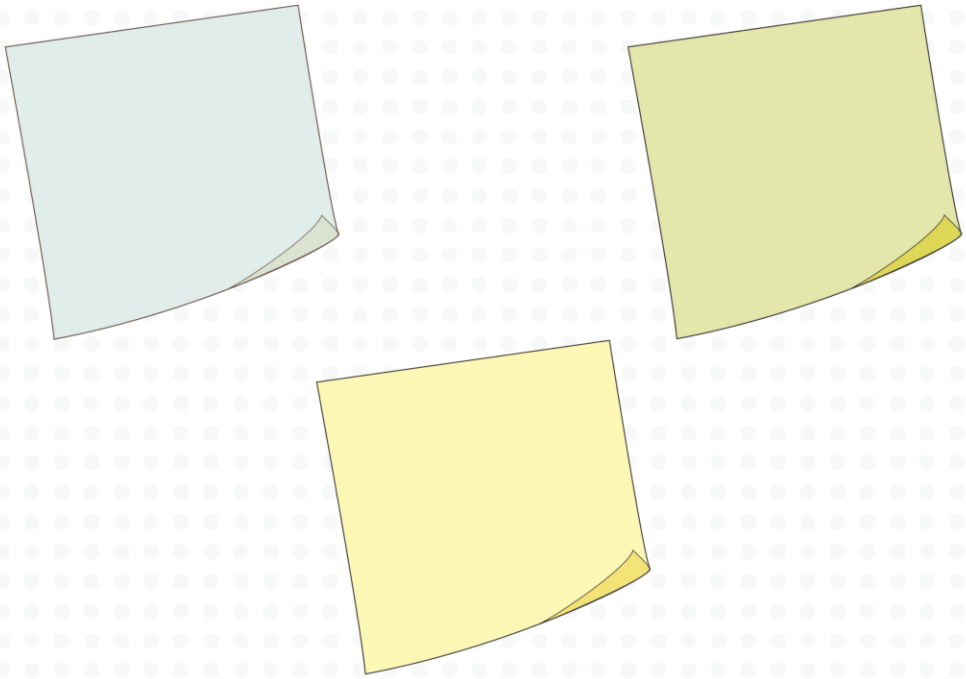

이벤트 소싱 이해

이벤트 모델링과 이벤트 소싱으로 확장 가능한 시스템 설계 및 구현

마틴 딜거 지음
임진호 옮김

아담 디미트룩, 가브리엘 N. 쉔커 추천사

이벤트 소싱 이해

Version: 1.0.0-sample (2025-06-26)

본 출판물은 Leanpub을 통해 출판 및 배포됩니다.

오타 혹은 문의사항은 jinhoyim@outlook.com 으로 보내주시기 바랍니다.

첫 번째 에디션

Table of Contents

옮긴이 서문

이벤트 소싱이라는 개념을 본격적으로 접한 건 2016년 무렵이었습니다. SNS와 블로그에서 간간이 언급되던 이 개념은, 이듬해 CQRS와 이벤트 소싱을 주제로 한 오프라인 모임에 귓동냥이라도 해보자는 마음으로 참석하면서 조금 더 가까워졌습니다. 이후 스프링 캠프, ASP.NET Korea 같은 개발자 행사에서도 점차 다뤄지기 시작했고, 이제는 다양한 온라인 세미나와 블로그, 도메인 주도 설계 관련 도서에서도 종종 등장하는 주제가 되었습니다.

자료가 많아졌지만 대부분 단편적이거나 인프라 구현을 깊게 다룹니다. 구현을 통해 동작 방식을 이해하는 데에는 도움이 되지만, 복잡한 코드를 따라가다 보면 흐름을 놓치곤 했습니다.(하지만 좋은 자료들이고 이 책에서 다루지 않는 내용을 포함하므로 시간을 내어 살펴보시기를 권장합니다.)

이 책은 다릅니다. 저자는 이벤트 소싱 인프라를 세세하게 구현하기 보다 Axon Framework를 활용하여 비즈니스를 구현합니다. 그리고 친숙한 장바구니 도메인을 바탕으로 '이벤트 모델링'이라는 도구를 활용해 분석, 설계, 구현, 테스트까지 하나의 일관된 흐름을 보여줍니다. 덕분에 독자는 이벤트 소싱 인프라 측면 보다는 핵심적인 흐름과 구조에 집중할 수 있습니다.

특히 인상 깊었던 부분은 이벤트 모델링의 도입입니다. 이벤트 모델링은 단순한 설계 도구를 넘어, 협업과 문서화, 시스템 분석 등 다양한 소프트웨어 공학적 목적에 활용될 수 있는 도구입니다. 해석의 차이로 인한 오해가 발생하기 쉽고 관리와 학습 부담이 큰 기존 방식들과 달리, 직관적이고 실행 친화적이며 빠른 피드백을 제공합니다. 이벤트 스토밍의 장점을 수용하면서도, 코드와 밀접하게 연결되어 있어 가볍게 시작하고 빠르게 실험해볼 수 있는 구조를 갖추고 있습니다.

이 책은 이벤트 소싱을 중심에 두고 사고의 전환을 유도하며, 어떤 상황에서

적합한지, 어떤 선택이 실용적인지를 함께 고민하게 만듭니다. 흔히 "CRUD를 기본으로 하되, 필요한 곳에만 이벤트 소싱을 쓴다"는 접근을 넘어서, 처음부터 이벤트 소싱을 고려하고, 필요한 경우에 CRUD 방식을 선택하라고 주장합니다. 이는 초기 설계의 불확실성, 변화에 대한 유연성, 데이터의 장기적 가치에 대한 깊은 고민에서 비롯된 관점입니다.

물론, 이 책에서도 이벤트 소싱의 가장 복잡한 구성을 모든 시스템에 적용하라고 주장하지는 않습니다. 저자 역시 복잡성을 경계하며 단순하게 시작하기를 권장하며, 필요한 경우 더 작고 제한된 범위에 적용할 수 있는 실용적인 대안을 함께 제시합니다. 중요한 것은 도구 자체가 아니라, 상황에 맞게 판단하고 적용하는 것이라는 점을 거듭 강조합니다.

"망치를 든 사람에겐 모든 게 못으로 보인다."는 말이 있습니다. 이벤트 소싱을 배우고 나면 모든 문제를 이 방식으로 풀고 싶어질 수 있습니다. 하지만 이 책은 오히려 RDB와 CRUD가 가지고 있는 유일한 망치는 아니었는지 돌아보게 합니다.

저 역시 과거에는 "시스템이 작아서", "아직 때가 아니라서" 같은 이유로 이벤트 소싱을 선택도, 준비도 하지 않았습니다. 정확히 말하면 하지 못했습니다. 개념은 알고 있었지만, 애플리케이션에 어떻게 적용할지 감이 잡히지 않았고, 충분히 이해하지 못한 것에 대한 두려움도 컸습니다. 그런데 이 책을 읽고 나서 어렵게만 느껴졌던 기존 자료들이 새롭게 보이기 시작했습니다. 흐름이 보였고, 의도가 읽혔고, 한 걸음 더 나아갈 수 있게 되었습니다.

이 경험을 다른 분들과도 나누고 싶어 이 책의 번역을 결심하게 되었습니다. 익숙하지 않음에서 오는 두려움을 줄이고, 정보 시스템을 바라보는 새로운 관점과 도구를 갖추는 데 이 책이 좋은 출발점이 되기를 바랍니다.

참고로, 이 책에서는 커맨드Command, 애그리거트Aggregate, 핸들러Handler 등

일부는 번역하지 않고 음차 표기를 유지했습니다. 실제 코드나 모델, 그리고 다른 원문을 참고하며 읽는 분들께 도움이 되기를 바랍니다.
또한 다이어그램이 포함된 도서의 특성 상 일부 그림은 확대가 필요한 점을 양해 부탁드립니다.

임진호

이 책을 쓰게 된 이유

처음 이 장은 계획되지 않았습니다. 하지만 많은 사람들로부터 실제로 제가 관심을 갖는 이유가 무엇인지에 대한 질문을 받았습니다. 저는 왜 이런 노력을 하는 걸까요? 저는 무엇에 이끌렸을까요?

저는 2005년쯤부터 소프트웨어 개발을 시작하여 이 분야에서 20주년을 앞두고 있습니다. 20년이나 지난 것 같은 느낌은 전혀 들지 않았습니다. 그러나 한 가지는 변함 없이 그대로였습니다. 소프트웨어 개발은 그 당시에도 어려워 보였고 오늘날에도 마찬가지입니다.

반면 그렇지 않은 프로젝트들도 경험할 수 있었습니다. 잘 진행된 프로젝트들이 있었습니다. 계획대로 잘 진행되었고 마무리도 성공적이었습니다. 저는 가치 전달을 방해하는 많은 요소들을 무시하고 중요한 것에 집중할수록 우리가 더 나은 결과를 얻을 수 있다고 생각합니다.

예전에 한 고객이 이런 말을 한 적이 있습니다. "소프트웨어 개발자는 피자 배달원 같아요. 항상 늦고 약속한 것의 절반만 가져다주며 화려한 광고에 비해 만족스러웠던 적은 없어요."

맞습니다. 제 생각에도 소프트웨어 개발은 지난 20년 동안 크게 변하지 않았습니다. 지난 40년 동안도 마찬가지일 겁니다. 애자일, 마이크로서비스, 클라우드, AI는 이러한 사실을 근본적으로 바꾸지 않았습니다. 소프트웨어 개발은 여전히 누구도 원하지 않는 피자 같습니다. 하지만 주문했기 때문에 먹어야 합니다.

2021년 저는 모든 것을 바꿔놓을 이벤트 모델링에 대해 알게 되었습니다. 드디어 고객이 이해할 수 있는 언어로 대화할 수 있게 되었습니다. 이해할 수 없고 관심조차 없는 기술 용어나 세부 사항으로 혼란을 유발하지 않으며, 그들이

진정으로 걱정하는 문제에 대해 논의할 수 있었습니다.

우리는 해결해야 할 진짜 문제에 전적으로 집중할 수 있었습니다.

여러 문제의 근본 원인이 많은 개발자가 생각하는 것처럼 성급한 최적화는 아니었습니다. 공통 언어의 부족, 즉 성공적인 의사소통을 위한 체계화된 방법이 부족해서였습니다.

이는 새로운 것이 아닙니다. 에릭 에반스의 저서 '도메인 주도 설계Domain-Driven Design'는 IT와 비즈니스 간의 공통 언어에 대해 자세히 설명합니다. 그는 이것을 '유비쿼터스 언어Ubiquitous Language'라고 명명했습니다. 8장에서 이에 대해 더 자세히 논의하겠습니다.

이벤트 스토밍EventStorming[1]의 고안자인 알베르토 브란돌리니Alberto Brandolini는 "소프트웨어가 되는 것은 비즈니스 지식이 아니라 개발자의 이해이다"라고 말했습니다.[2]

하지만 개발자가 단순히 잘못 이해했다면 어떨까요? 공통 언어가 없고 용어도 명확하지 않으며 문제를 제대로 논의할 시간조차 없어서 비즈니스와 개발 간에 지속적인 오해가 쌓여간다면 어떨까요? 지난 20년 동안 다양한 기술로 해결하고자 했던 문제가 이것이 아닐까요?

그 당시 이벤트 모델링은 마술처럼 보였습니다. 사용할 때마다 효과가 있었고 심지어 처음 만난 사람들과 일할 때에도 마찬가지로 효과가 있었습니다. 갑자기 모두가 이해를 했습니다. 우리는 비로소 서로 대화를 나눌 수 있게 되었습니다. 저는 가능한 한 그것을 사용하기 시작했습니다. 처음에는 제대로 하고 있는지 확신할 수 없었지만, 실제로는 너무 간단해서 점점 자신감이 생겼습니다.

마침내 명확한 의사소통이 이루어졌습니다. 우리는 앞으로 해결하고자 하는

사업과 문제에 대해 자신 있게 논의할 수 있었습니다. 더 나아가, 이러한 문제를 근본적으로 해결할 수 있는 '가치 있는 솔루션'의 비용을 이야기할 수 있게 되었다는 점입니다. 우리는 계획한 시스템에 대해 큰 신뢰를 얻을 수 있었고 남아 있던 불확실성, 즉 중요한 것이 간과되고 있다는 찝찝한 느낌을 없앴습니다.

저는 최악의 상황들을 본 적이 있습니다. 엄청난 양의 돈이 헛되이 타버렸고 시간과 노동력, 그리고 지식이 낭비되었습니다. 끝없는 회의에서 결과 없는 토론만 진행되었습니다. 누구도 제대로 이해하지 못한 요구사항들이 수개월, 수년 앞을 내다보는 견적과 계획에 사용되었고, 결국 제품은 초기의 계획보다 훨씬 더 오래 걸렸습니다.

그때부터 저는 다른 길을 추구하기로 결심했습니다. 우리 업계가 더 잘할 수 있는 방법이 있다고 판단했습니다. 그리고 저는 그것에 도달하기 위해 모든 것을 바꿀 필요는 없다는 것을 깨달았습니다. 몇 가지 사소한 조정만 하면 됩니다. 2024년 1월 웨비나[3]에서 저는 소프트웨어 개발이 향후 2년 내에 근본적으로 바뀔 것이라고 말했습니다. 그리고 저는 우리 업계가 올바른 방향으로 나아가고 있다고 생각합니다.

이 책은 지난 15년간 제가 배운 모든 것을 담고 있습니다. 또한 2021년에 발견한 내용이기도 합니다. 프로세스의 작은 변화, 진짜 문제에 대한 집중, 모두가 이해하는 언어로 소통하는 것이 진정한 차이를 만들어냅니다.

우리는 이를 '가속화한다accelerate'라고 표현하며, 이러한 아이디어를 사용하는 팀은 가속화되는 것을 빠르게 느낄 수 있습니다. 소프트웨어 개발은 기름칠이 잘 된 기계처럼, 계획부터 구현까지 끊임없이 순환하며 진행됩니다. 속도를 유지하면서 변화를 받아들여야 합니다. 이 개발 프로세스에 대한 설명과 제가 일반적으로 작업하는 방식은 가속화와 관련된 부록에서 확인할 수 있습니다.

저는 거인의 어깨 위에 서 있습니다. 이 책에 제시된 아이디어 중 어느 것도 저만의 것이 아닙니다. 저는 아담 디미트룩, 그렉 영, 지미 보가드, 에릭 에반스, 반 버논처럼 저보다 훨씬 똑똑한 사람들이 생각해 낸 것들을 모두 합쳤습니다.

이것은 저를 움직이는 원동력이며 정말 기대됩니다. 우리가 이런 방식으로 일하기 시작하면 어떤 변화가 나타나는지 보고 싶습니다.

마틴, 2024년 7월에

PS: 한 가지 더 말씀드리겠습니다. 저는 이 책에 온 마음을 담았고, 많은 사람들의 도움으로 3개월 만에 이 책을 쓸 수 있었습니다. 훌륭한 책이기를 바라지만 아직은 완벽하지 않습니다.

문제가 발견되면 개선할 수 있도록 도와주세요. 여기 깃헙에서 이슈를 열어주세요.

https://github.com/dilgerma/eventsourcing-book/issues

이 책을 최고의 책으로 만들어 보겠습니다.

[1] https://www.eventstorming.com/

[2] https://x.com/ziobrando/status/1347126001340358656

[3] https://nebulit.de/bessere-software-2024

1장 관심을 가져야 하는 이유

이벤트 소싱이란 무엇이며 개발자, 팀 관리자 또는 회사로서 관심을 가져야 하는 이유는 무엇입니까? 이벤트 소싱은 새로운 기술이 아닙니다. 2006년경 소프트웨어 산업에 등장했습니다. 이 책을 쓸 당시 소프트웨어 개발의 주류에 등장한 지 약 15년이 지났습니다 (기원으로 거슬러 올라가면 더 오래되었지만요).

그래서 왜 관심을 가져야 할까요?

구현 패턴으로서의 이벤트 소싱은 다소 간단합니다. 데이터를 테이블과 관계로 저장하는 대신, 시스템에서 실제로 발생한 사실들을 순서대로 저장합니다.

이는 최근에 일어난 일에 대한 스토리가 시스템의 데이터와 같다는 의미입니다.

먼저 고객이 생성된 다음, 새 주소로 이전했습니다. 그리고 큰 구매를 했습니다. 그로 인해 고객은 프리미엄 상태로 승격되었습니다. 이러한 모든 사실(이벤트 소싱에서는 이를 '이벤트'라고 함)을 합치면 프리미엄 상태의 새 주소를 가진 고객을 얻을 수 있습니다.

시스템이 데이터를 정규화된 관계형 스키마에 저장하는 전통적인 CRUD 기반 접근 방식에서는, 동일한 고객을 가져오더라도 시간이 지남에 따라 고객이 어떻게 프리미엄 고객이 되었는지에 대한 기록을 잃을 수 있습니다.

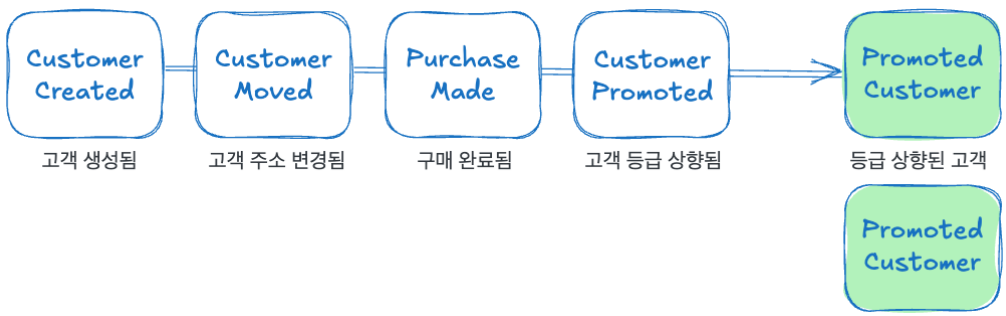

그림 1-1. 이력 기반 데이터(위) / 평면화된 데이터(아래)

결론적으로는 같은 결과를 얻습니다. 이것이 왜 중요할까요?

회사에서 데이터를 활용하려고 할 때 중요해집니다. 고객의 프리미엄 등급이 특점 시점 이전인지 이후인지 알아야 할 필요가 있다면 어떨까요? 만약 새 주소로 이사한 고객이 2주 동안 가구를 구매하는 데 소비한 금액을 알고 싶다면 어떻게 해야 할까요? 고객의 주소가 언제, 그리고 얼마나 자주 변경되었는지 알아야 한다면 어떨까요?

이벤트 소싱을 사용하면 시스템에 시간의 차원을 추가합니다. 시스템은 타임라인에 따라 발생하는 사실을 기록합니다. 데이터가 관계형 데이터베이스 스키마로 평면화되면 이 차원이 사라집니다.

물론 데이터 웨어하우스(있는 경우)를 조회하여 일부 데이터를 얻을 수 있습니다. 데이터 웨어하우스는 쉽게 말해, 시간 경과에 따른 모든 관련 데이터 변화를 저장하는 대형 데이터베이스입니다. 만약 이 모든 데이터를 자유롭게 활용하여 비즈니스 부서에서 어떤 아이디어를 내놓든 미래에 활용할 수 있다면 어떨까요?

지난 10년 동안 수집한 데이터를 AI 모델에 공급할 수 있다면 어떨까요? 그리고 만약 이 모델이 막연한 추측이 아닌 확실한 근거를 바탕으로 새로운 시장을 개척하는 데 도움이 된다면 어떨까요? 회사가 과거로부터 진정으로 배우고 이익을 얻을 수 있는 유일한 방법은 상세한 기록을 유지하는 것입니다. 이를 위해 새로운 사고와 다른 접근 방식이 필요합니다. 그리고 유연한 형식으로 정보에 접근할 수 있어야 합니다.

이벤트 소싱을 사용하면 순전히 기술적인 용어로만 생각하는 함정에 빠지지 않고 시스템을 더 쉽게 추론할 수 있습니다. 시스템이 실제 동작하는 방식으로 모델링되면 시스템을 이해하는 것이 더 쉬워집니다.

"고객에게 차단 플래그를 설정해야 합니다"라고 비즈니스 담당자가 이해할 수

없는 말을 하는 대신 "고객 차단됨"이라는 새 이벤트를 만드는 것이 훨씬 더 자연스럽습니다. 이 이벤트는 특정 시점에 특정 이유로 고객이 차단되었음을 나타냅니다.

하지만 안타깝게도 이런 사고방식은 학교 또는 대학교에서 거의 가르치지 않습니다. 그래서 많은 개발자가 소프트웨어 업계에서 수년간 일한 후에도 이 간단한 개념에 어려움을 겪습니다.

'정보를 잃지 않는 것'은 이벤트 소싱의 토대입니다. 몇 년 후에 어떤 용도에 유용할지 알 수 없기 때문에 항상 정보를 손쉽게 사용할 수 있도록 합니다.

이벤트 소싱을 배우는 것은 새로운 언어를 배우는 것과 유사합니다. 이 접근 방식을 사용하여 작업이 수행되는 방식을 찾아봐야 하는 경우가 많습니다. 그러나 인터넷 검색을 시도하거나 ChatGPT에 문의하면 이러한 개념에 대한 정보를 찾는 것이 예상만큼 간단하지 않습니다. 정보는 있지만 여러 다른 출처에 흩어져 있습니다. 게다가 이벤트 소싱에 대한 책은 기존 정보 시스템을 다루는 책에 비해 훨씬 드뭅니다.

그럼 이제, 다시 질문하겠습니다. 왜 관심을 가져야 할까요?

정보는 새로운 금광이기 때문입니다. 데이터는 정보를 바탕으로 비즈니스 결정을 내리는 데 도움이 됩니다. 그리고 데이터를 항상 활용할 수 있게 유지하는 것은 이벤트 소싱을 사용할 때 자연스럽게 얻어지는 결과입니다.

이 책에서는 이벤트 소싱과 그 주변의 모든 것들에 대해 더 깊이 파고들 것이며, 주로 구현 측면에 초점을 맞추겠습니다. 우리에게 가장 절실히 필요한 것은 실제 사례를 구현하는 방법에 대한 명확한 가이드입니다.

물론 관련된 이론과 이러한 방식으로 일하는 것이 어떤 의미인지도

논의하겠습니다. 하지만 다른 책과 정보와는 다르게 이벤트 소싱 시스템 구축의 진정한 의미를 이해할 수 있도록 실제 구현을 더 깊이 파고들고자 합니다.

그 뿐만이 아닙니다.

익숙할 수도 그렇지 않을 수도 있는 협업 모델링 기술을 사용하여 이벤트 소싱 시스템의 계획 단계에 대해 자세히 알아보겠습니다. 지난 10년 동안 아담 디미트룩이 개발하고 다듬어온 기술인 이벤트 모델링에 집중적으로 초점을 맞춰보겠습니다.[1] 이벤트 모델링은 정보 시스템을 명확한 언어로 계획하고 토론하는 데 도움이 됩니다. 이벤트 소싱 시스템에 한정된 것은 아니지만 특히 잘 어울린다는 것을 알 수 있습니다.

그렇다면 이 책에서 배우게 될 내용은 무엇일까요?

제가 15년 동안 터득한 것들을 생각보다 빠르게 배울 수 있습니다.

올바른 접근 방식과 확실한 계획만 있다면, 한 번에 그치는 것이 아니라 지속적으로 계획과 확장이 가능하며 잘 구조화된 정보 시스템을 구축할 수 있습니다. 이 책에서는 이러한 계획을 어떻게 세울 수 있는지 공유하겠습니다. 일부는 저의 개인적인 관점을 반영하고 있으며 일부 내용에 동의하지 않을 수 있습니다. 이해합니다. 이 책은 추상적이지 않습니다. 이벤트 모델링, 이벤트 소싱, 그리고 구현 측면에 대한 세부 사항까지 자세히 살펴보겠습니다. 책의 내용이 다소 어려울 수 있으니, 이 점 감안해 주시기 바랍니다.

우리는 이벤트 소싱 패턴에 대해서도 자세히 알아볼 것입니다. 여기서는 널리 공유되지 않는 몇 가지 기술을 배울 것이며, 확장할 수 있는 방식으로 이벤트 소싱 시스템을 구현하는 데 도움이 됩니다.

제가 정보 시스템을 개발해오는 동안 아키텍처와 시스템을 어떻게 모듈 단위로

구성했는지 알려드리겠습니다. 이것이 '최고' 또는 '완벽한' 접근 방식은 아닐 수 있지만 제게는 효과적이었습니다.

가능한 한 간결하고 집중적으로 설명하겠습니다. 여러분의 성공에 중요하지 않은 내용은 제외하겠습니다.

믿기 어렵겠지만 분명한 로드맵과 안내해 줄 사람이 있다면 이벤트 소싱 시스템을 구현하는 것은 생각보다 간단합니다. 제가 처음 시작할 때 이런 자료가 있었다면 좋았을 것 같습니다. 그래서 여러분이 오늘 바로 시작할 수 있도록 여정을 돕는 것이 저의 목표입니다.

[1] https://eventmodeling.org/