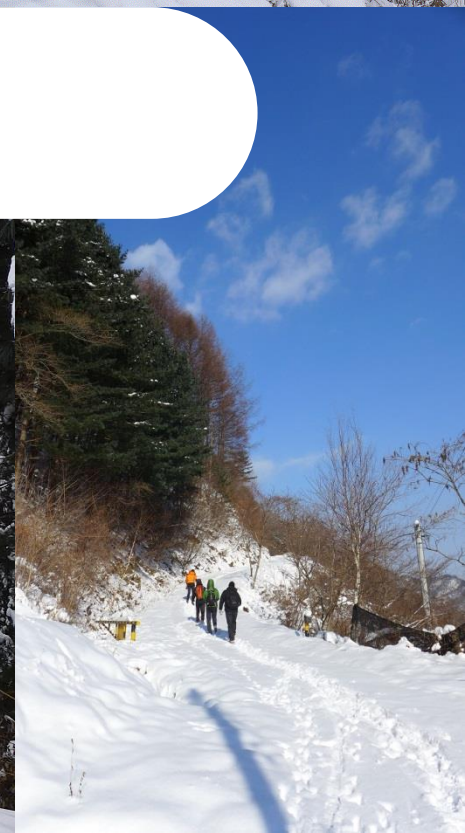



Not design, but choice

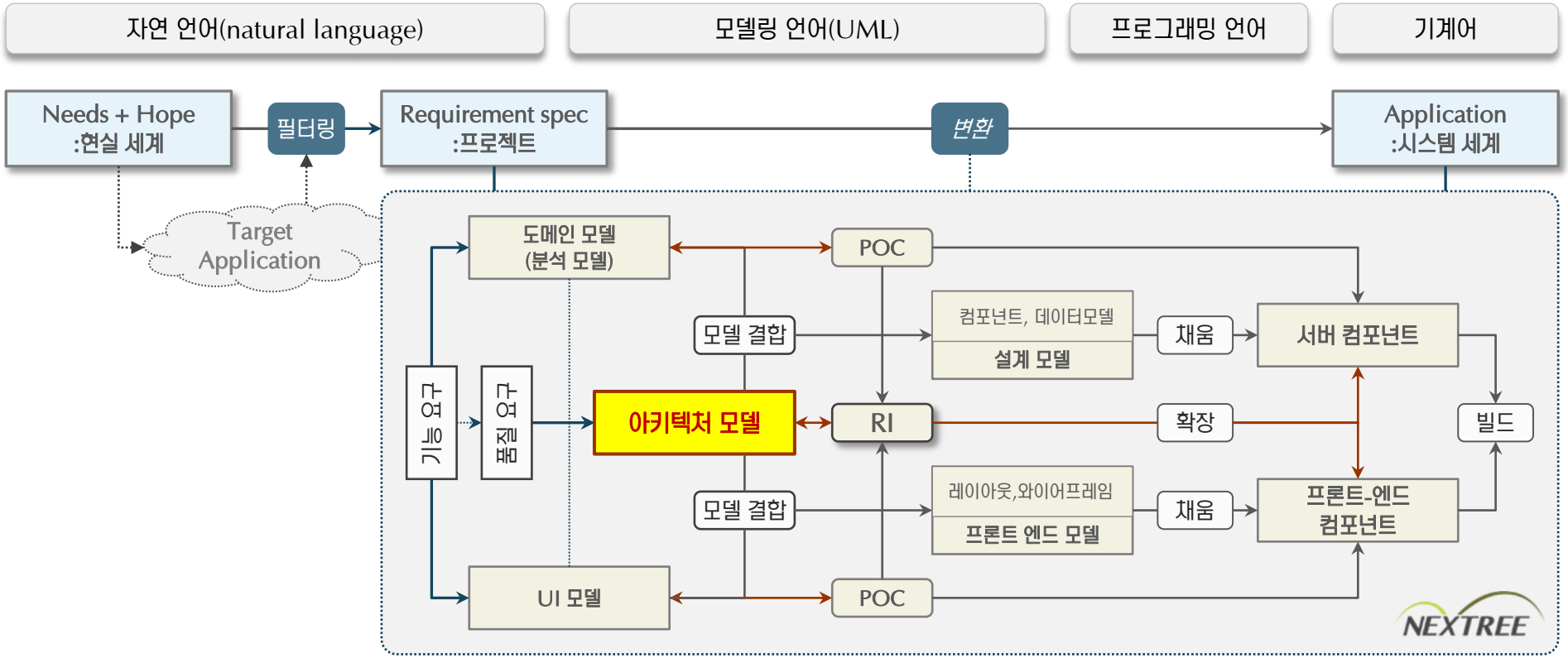




목차

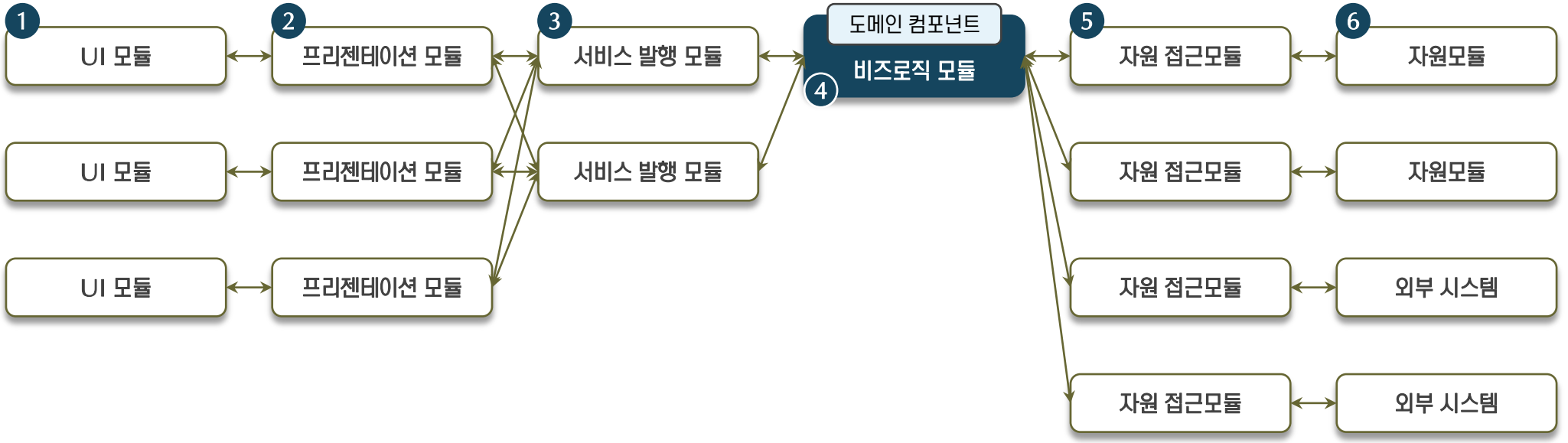
1. 시스템 개발 절차

✓ 개발 프로세스의 목표는 현실 세계의 필요를 충족하는데 필요한 시스템을 구축하는 것입니다.
✓ 개발이란 현실 세계에 존재하는 개념들을 필터링, 변환, 번역 과정을 거쳐서 작은 시스템 무대로 이동시키는 것입니다.
✓ 일련의 과정에서 서로 다른 언어(자연어 → 모델링 언어 → 프로그래밍 언어)로의 연속적인 변환이 이루어집니다.
✓ 개발 기술과 프로세스는 소프트웨어 개발의 특성을 반영하여야 합니다. ← 소통이 중요한 이유입니다.



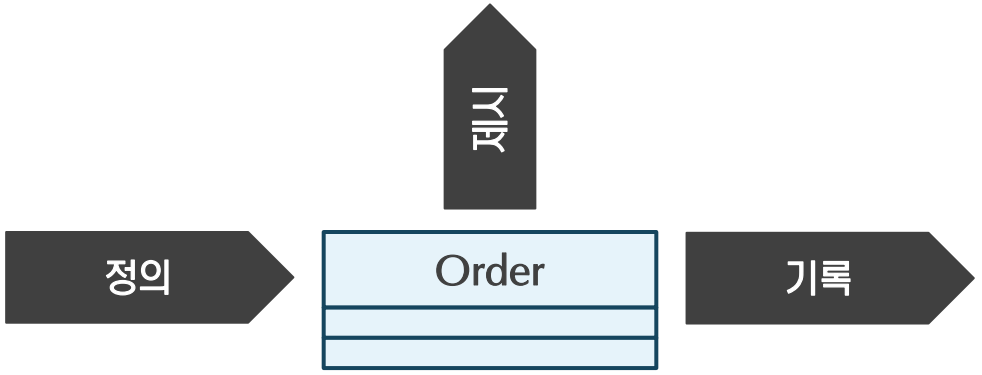
2. 아키텍처 설계의 목표

- ✓ 잘 설계된 애플리케이션은 Loose coupling, High cohesion 원칙을 잘 지켰을 겁니다.
- ✓ 서로 다른 시스템 구성 요소인 도메인과 기술이 잘 분리되었을 겁니다. 분리와 결합이 자유롭습니다.
- ✓ 기술 요소의 변화는 레이어를 기준으로 자유로우며, 도메인은 기술의 변화에 영향을 받지 않습니다.
- ✓ 도메인 컴포넌트는 각 레이어 기술 요소로부터 중립적이며, 의존성을 갖지 않습니다.



3. 정보와 정보관리 시스템

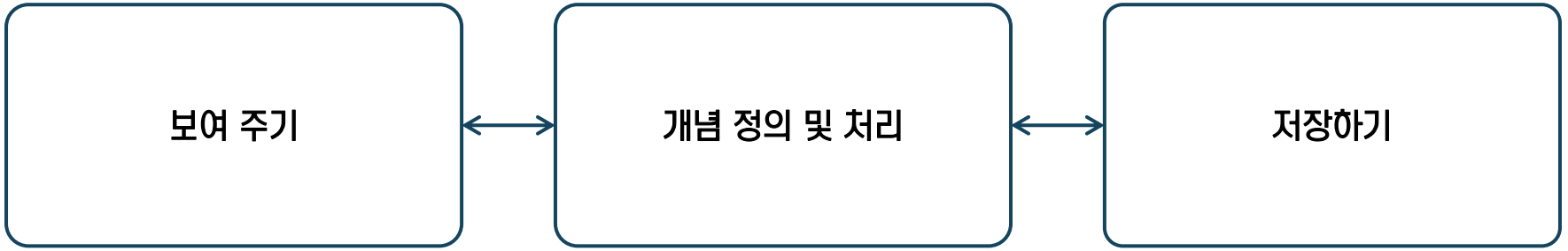
- ✓ 아키텍트 관점에서 정보의 정의는 무엇일까요?
- ✓ 정보에 대한 충분한 개념, 영속성을 갖기 위한 기록, 유용성을 갖기 위한 제시, 세 가지를 생각할 수 있습니다.
- ✓ 이런 정보의 특성에 대한 이해는 정보 시스템을 설계하는데 도움이 됩니다.
- ✓ 정보 관리 시스템은 이 세 가지 특성을 기반으로 설계합니다.



정보란 개념이 정의되어야 하고, 영속성을 가져야 하고, 필요할 때 제시할 수 있어야 한다.

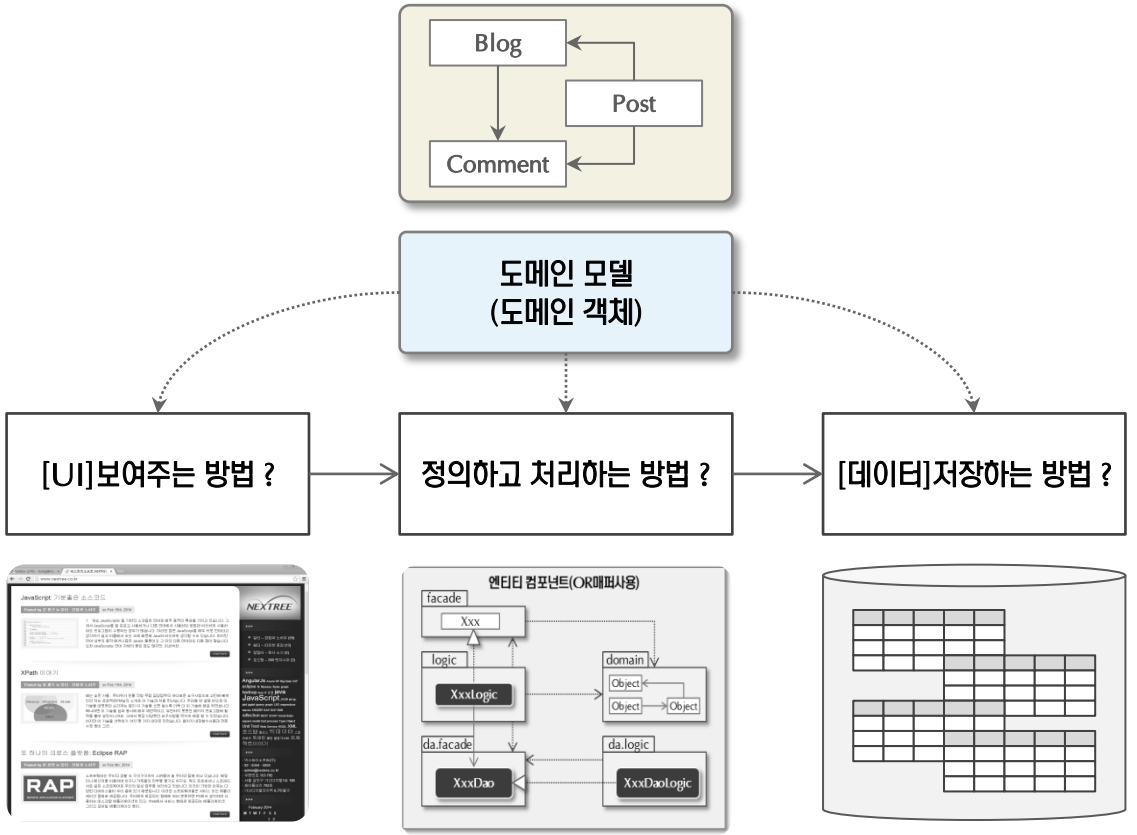
4. 아키텍처 설계

- ✓ 정보의 특성을 바탕으로 정보를 관리하거나 처리하는 시스템의 아키텍처를 설계해 봅니다.
- ✓ 시스템은 크게 세 부분으로 나뉘게 됩니다. ← 설계 원칙(Separation of concern)
- ✓ 시스템에서 가장 중요한 것은 정보 그 자체와 정보를 처리하는 부분입니다.
- ✓ 보여 주기와 저장하기는 기술의 특성을 많이 타는 부분입니다.



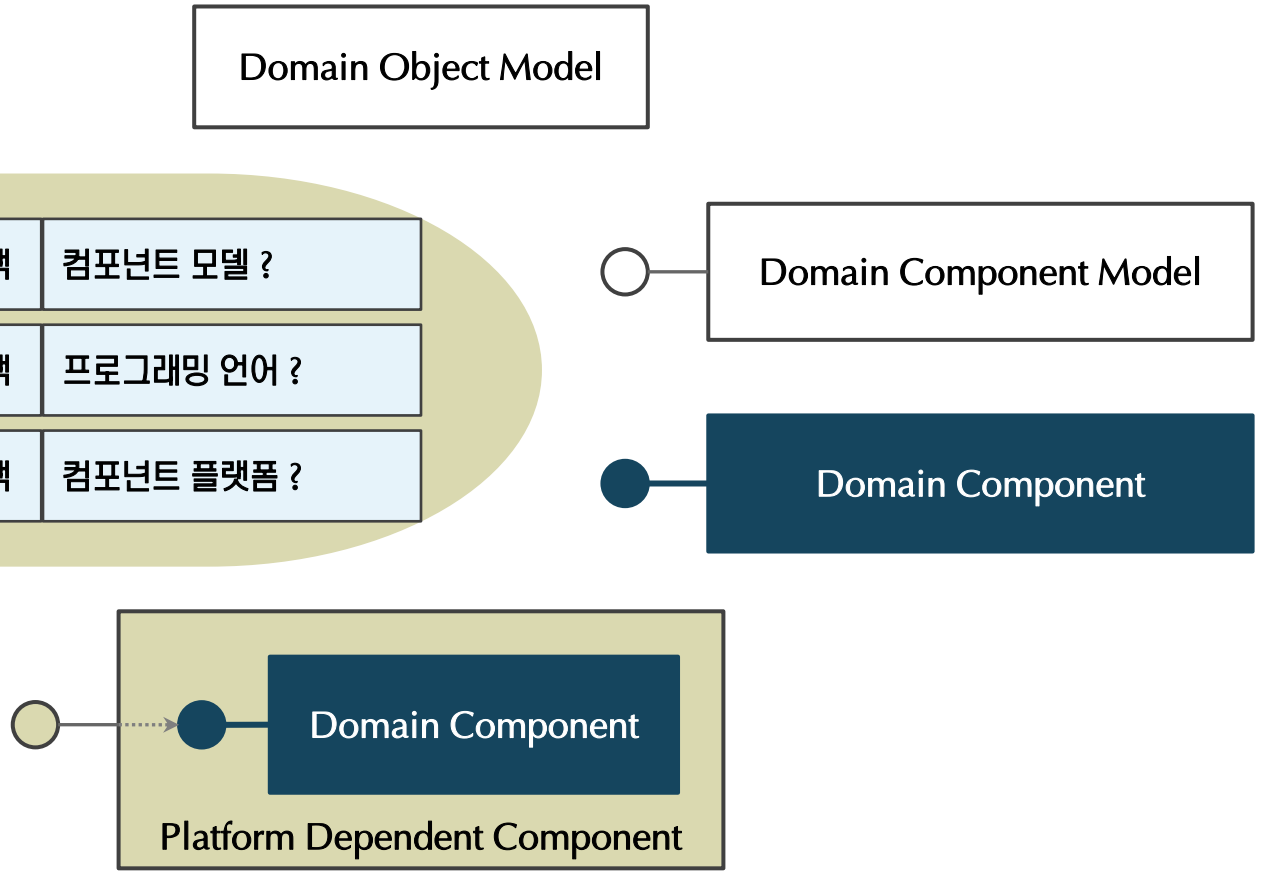
4.1 도메인 모델 보호

- ✓ 기술의 변화가 빠를 수록, 비즈니스 환경이 복잡할 수록 "도메인 모델"의 중요성은 더 커져 갑니다.
- ✓ 도메인 모델과 도메인 컴포넌트를 기술로부터 보호하도록 설계하는 것은 아키텍트의 의무입니다.
- ✓ 세 가지 영역은 서로 다른 설계 개념을 가지고 있습니다.



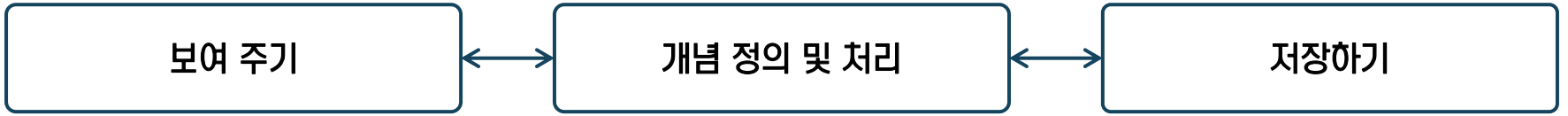
4.2 컴포넌트 구조 설계

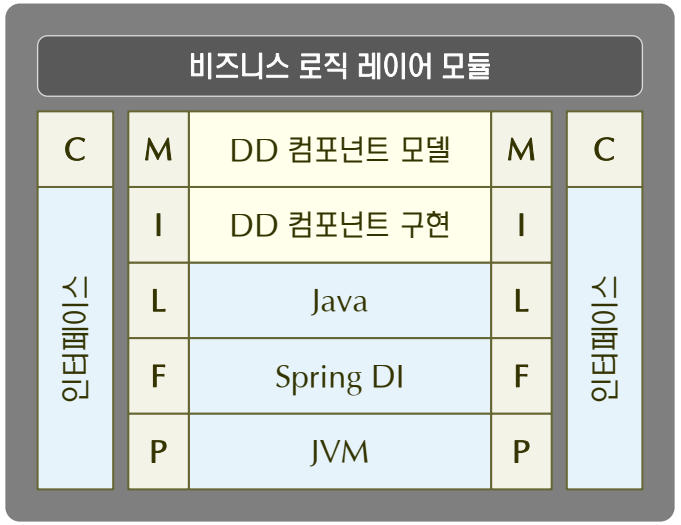
- ✓ 도메인 객체 모델을 담을 컴포넌트 모델을 어떤 구조로 할 것인지 선택해야 합니다.
- ✓ 목표 시스템이 어느 수준의 기술 독립성을 가져야 하는 지, 팀원의 기술 수준 등에 따라 컴포넌트 모델을 결정합니다.
- ✓ 컴포넌트를 구현할 언어와 컴포넌트 플랫폼을 선택해야 합니다.
- ✓ 도메인 모델링과 도메인 컴포넌트 구현은 분석가가 할 수도 있습니다.



4.3 비즈니스 로직 레이어 모듈 (1/2)

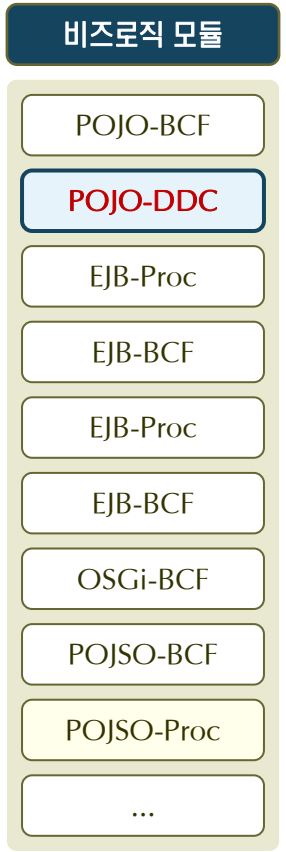
- ✓ 도메인 컴포넌트를 감싸고 있는 플랫폼 종속적인 컴포넌트는 비즈니스 로직이 됩니다.
- ✓ 각 레이어 모듈에서 선택 가능한 요소는 대략 여섯 가지 정도입니다.
- ✓ 플랫폼, 프레임워크/라이브러리, 표현 언어, 사용자 정의 모델, 모델 구현체, 외부로의 연결 등은 모두 선택 사항입니다.





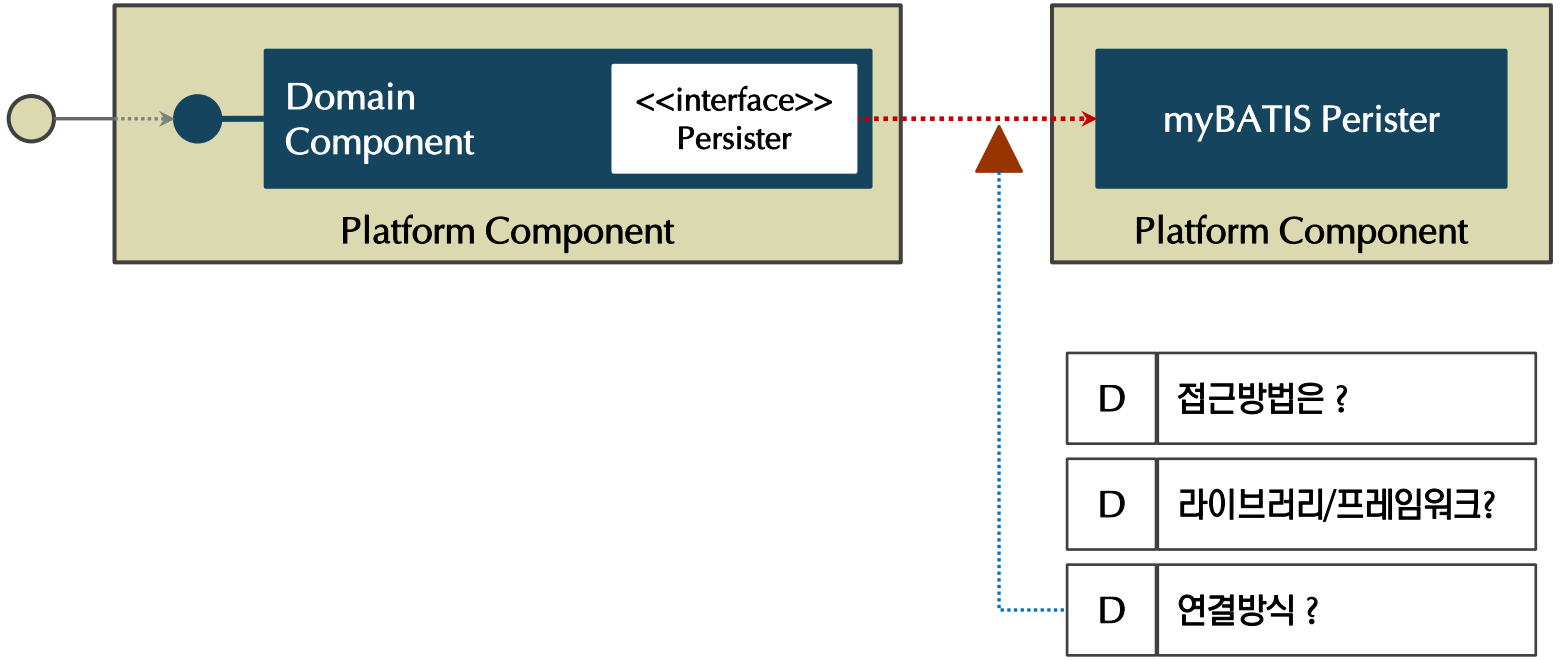
4.3 비즈니스 로직 레이어 모듈 (2/2)

- ✓ 여섯 가지 선택 요소는 결국 레이어 모듈의 특징을 결정하게 합니다.
- ✓ 비즈니스 로직 레이어를 구성하는 모듈은 컴포넌트 모델과 플랫폼 선택에 따라 다음과 같이 다양할 수 있습니다.
- ✓ 우리는 POJO 기반의 Domain [Driven] Component를 선택했습니다.



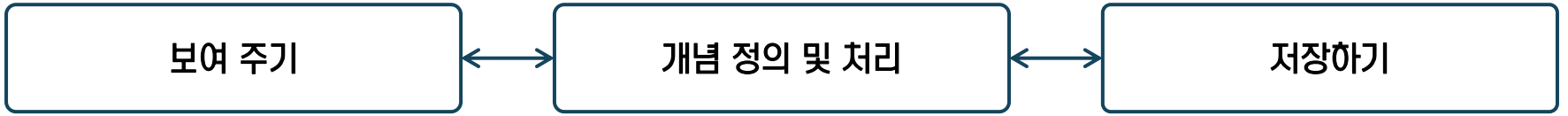
4.4 자원접근 레이어 모듈 (1/3)

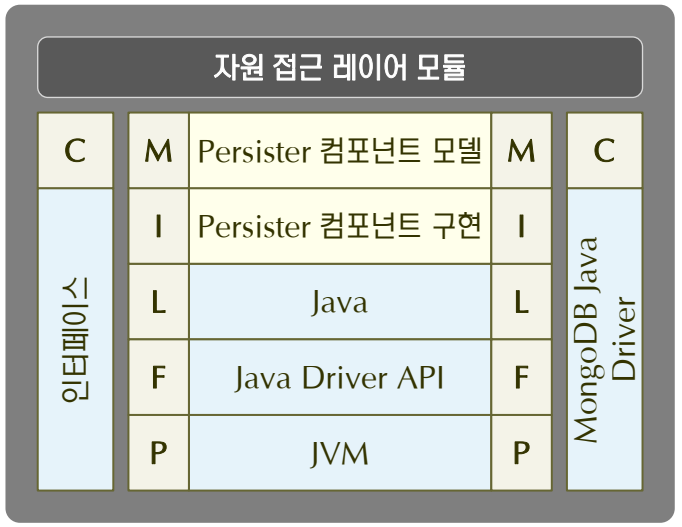
✓ 다음은 플랫폼 컴포넌트로부터 자원에 접근하는 구조를 설계합니다.
✓ 자원 접근 영역이 완전히 투명해야 하는가 아니면 어느 정도의 의존성을 허용하는가에 대해 고민을 합니다.
✓ 어떤 경우든 자원 접근 모듈을 대체 가능하도록 설계하는 것이 필요합니다.

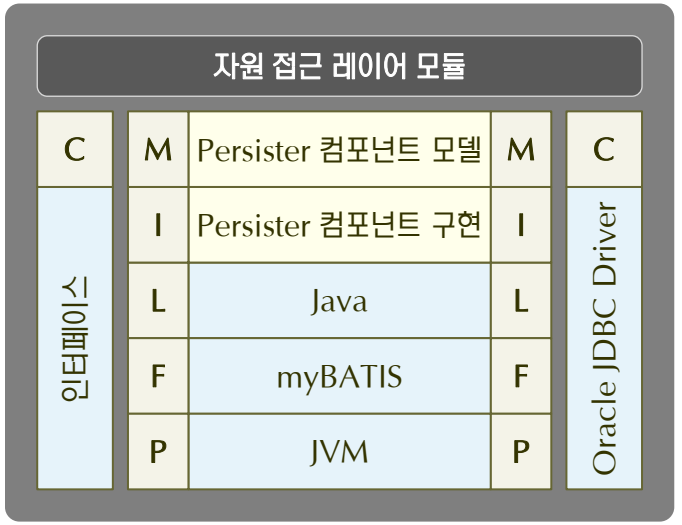


4.4 자원접근 레이어 모듈 (2/3)

- ✓ 고객의 요구와 시스템 환경 등을 고려하여 두 가지 접근 방법을 선택합니다.
- ✓ 읽기 쓰기는 오라클 DB로 읽기 전용은 MongoDB로 접근하도록 설계를 했습니다.
- ✓ 이 모듈은 동일한 Persister 인터페이스를 구현하므로, 필요할 경우 서로 대체할 수 있습니다.

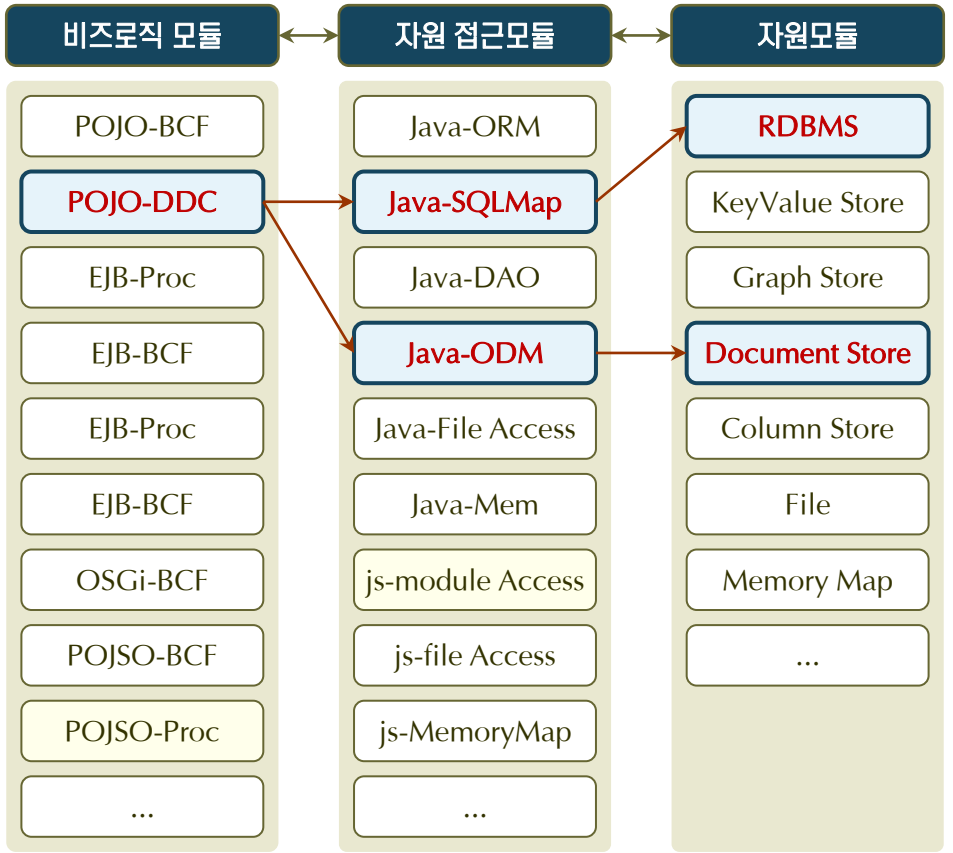






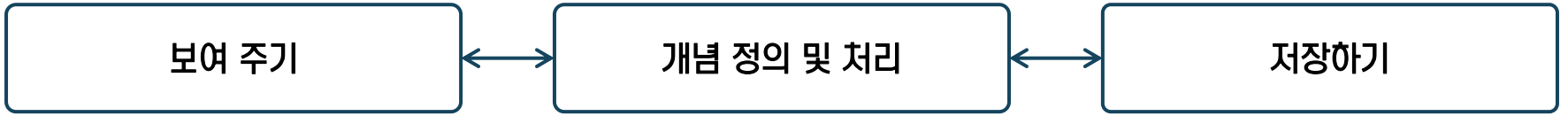
4.4 자원접근 레이어 모듈 (3/3)

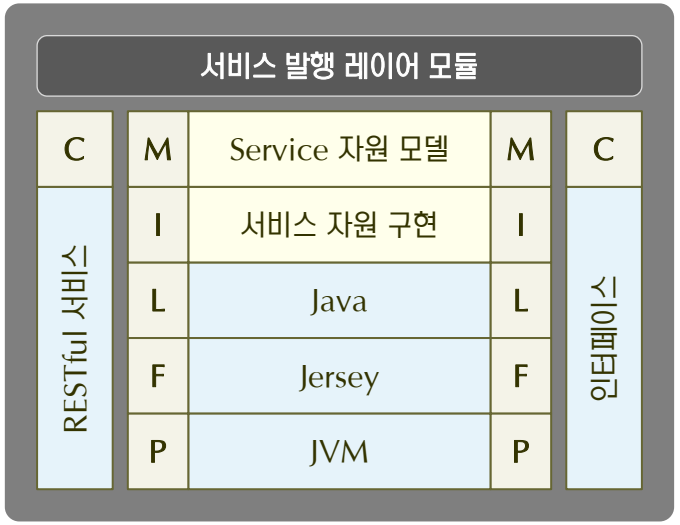
- ✓ 자원 모듈과 자원 접근은 매우 밀접한 관계가 있으므로 함께 선택합니다.
- ✓ 선택 가능한 자원 접근 방법 중에 두 가지를 선택했습니다.
- ✓ 트랜잭션 등에 대한 고민은 틀을 완성할 때까지는 뒤로 미룹니다.

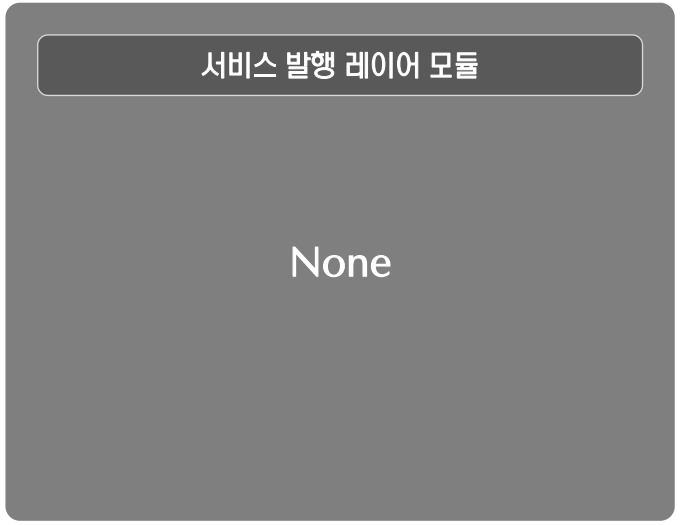


4.6 서비스 발행 레이어 모듈 (1/2)

- ✓ 정의한 정보나 처리된 결과 정보를 외부로 내보내는 레이어입니다.
- ✓ UI나 프리젠테이션 레이어로부터 직접 플랫폼 컴포넌트에 접근한다면 레이어 모듈은 “None”입니다.
- ✓ 웹 페이지의 Javascript 모듈에서 REST 서비스를 이용하여 데이터를 가져가는 경로도 준비했습니다.

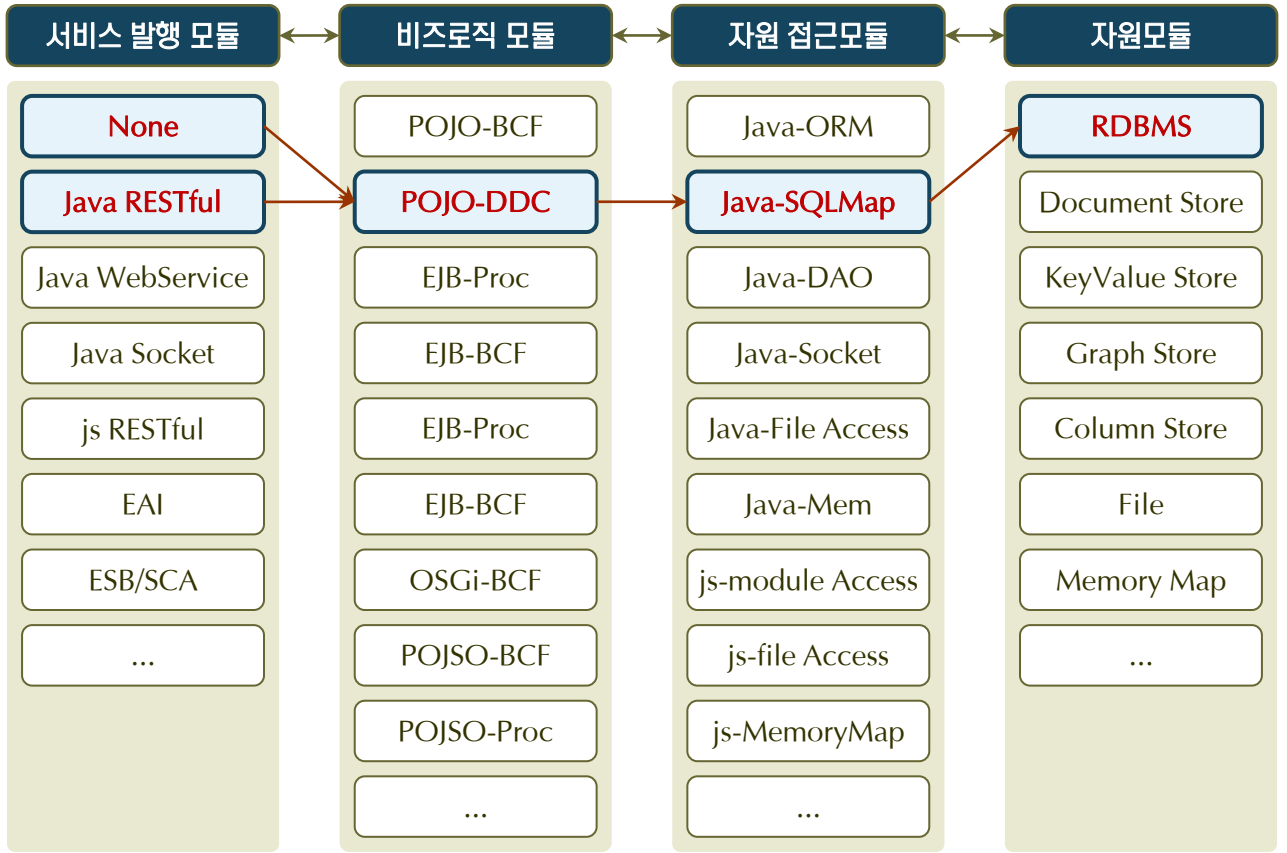






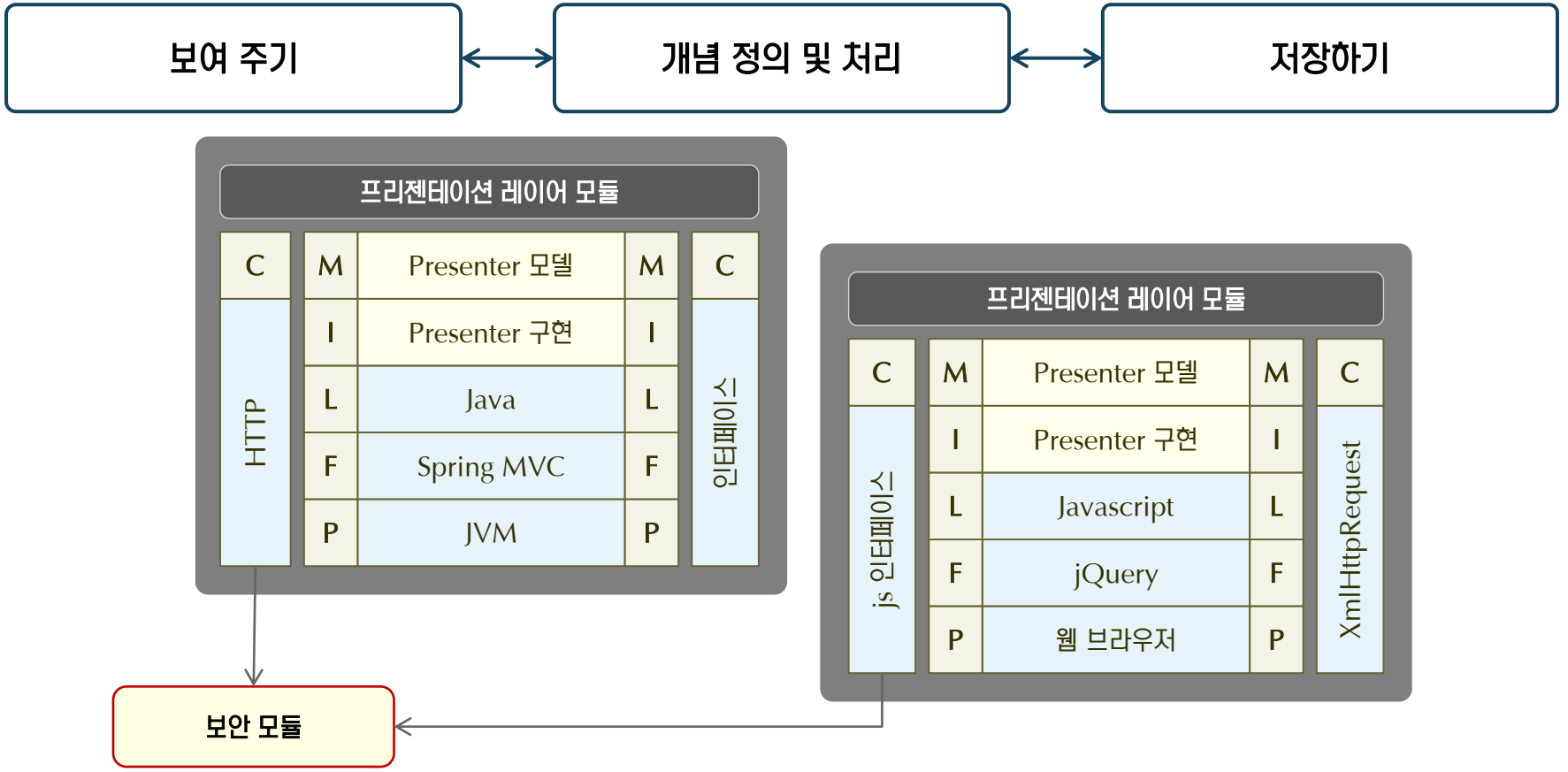
4.6 서비스 발행 레이어 모듈 (2/2)

- ✓ 프리젠테이션 모듈에서 컴포넌트로 직접 접근 경로와 RESTful 서비스로 접근하는 경로를 선택했습니다.
- ✓ 그 외 다양한 서비스 발행 방법이 있습니다.
- ✓ 요즘은 Node.js 기반의 Javascript RESTful 서비스 발행도 많이 선택되고 있습니다.



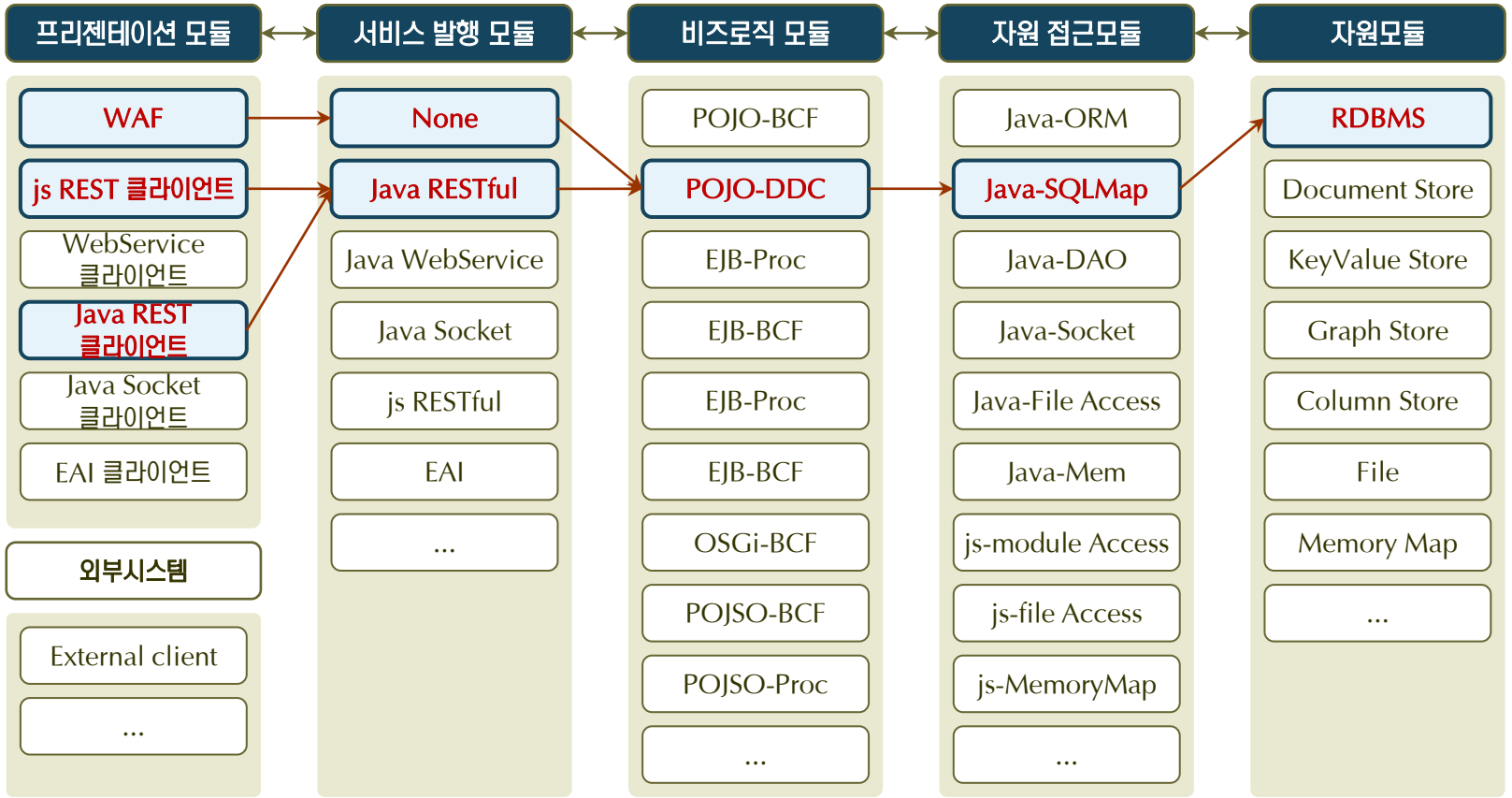
4.7 프리젠테이션 레이어 모듈 (1/2)

- ✓ 프리젠테이션 모듈은 두 가지를 선택하였습니다.
- ✓ Java와 Web Application Framework 위의 Presenter 모듈이 페이지 단위 UI 요청 처리를 합니다.
- ✓ 웹 브라우저에 내려가 있는 jQuery 모듈이 서버와 Ajax 통신을 하여 데이터를 가져 옵니다.



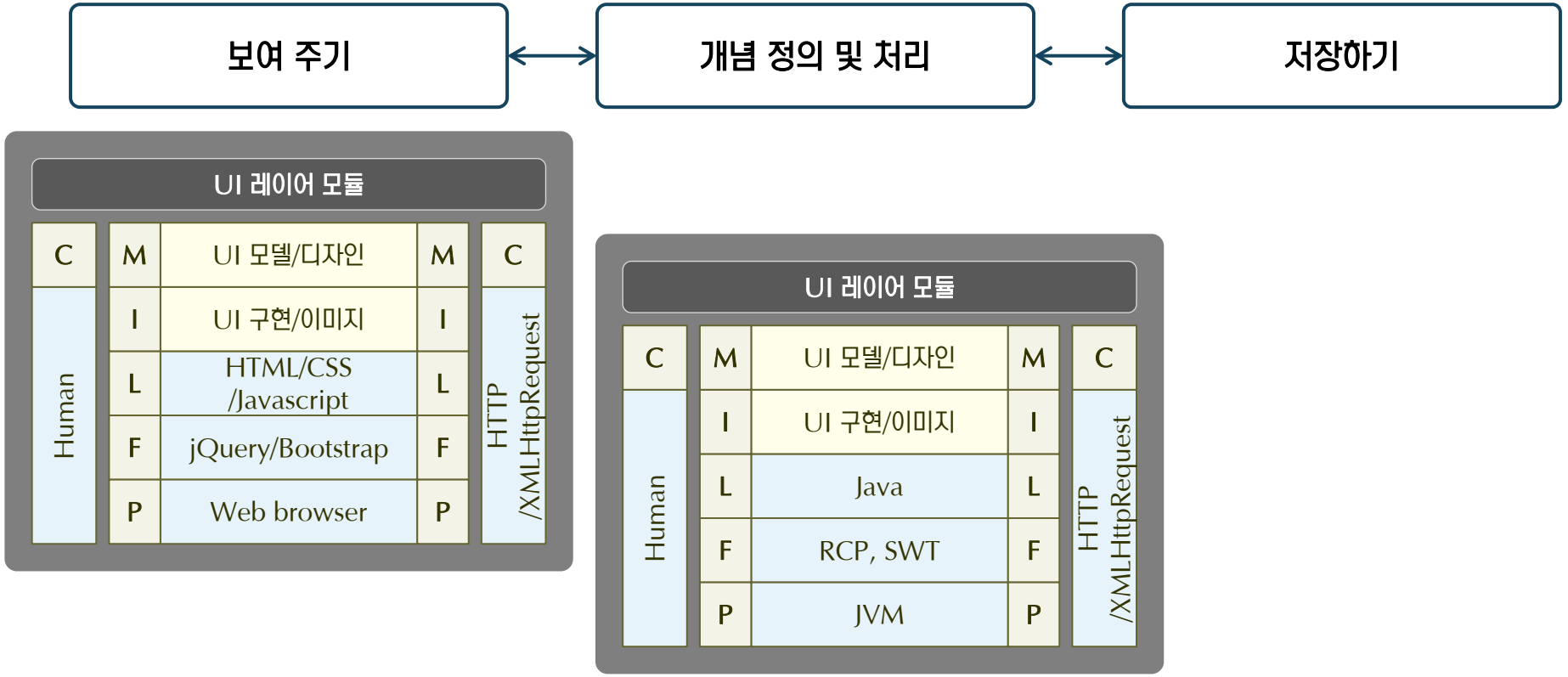
4.7 프리젠테이션 레이어 모듈 (2/2)

- ✓ 프리젠테이션 레이어는 UI 레이어와의 구분이 어려운 레이어입니다.
- ✓ 프리젠테이션 레이어에서는 두 개의 서로 다른 경로를 선택했습니다.
- ✓ WAF는 서버 사이드에서, Javascript RESTful 클라이언트는 웹 브라우저에서 동작합니다.



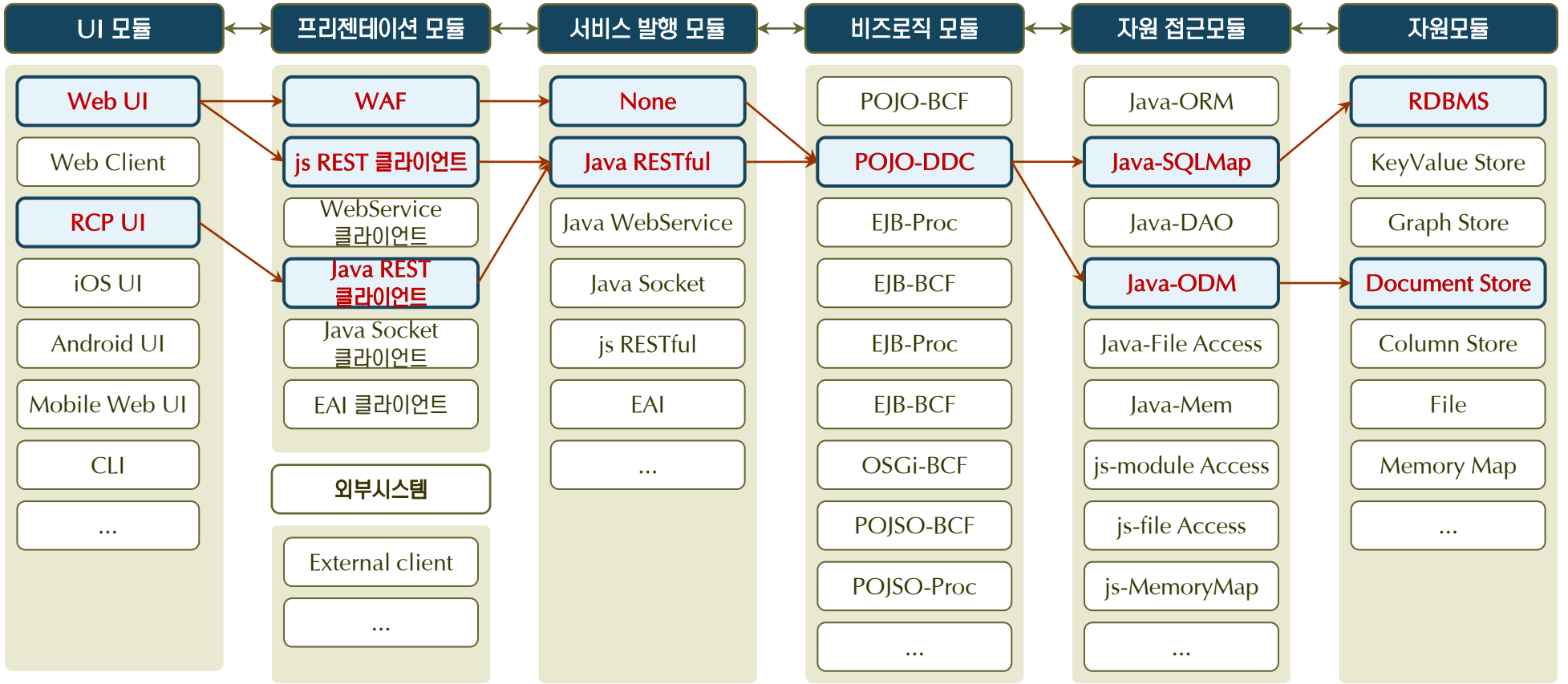
4.8 UI 레이어 모듈 (1/2)

- ✓ UI 레이어를 위한 플랫폼으로 웹 브라우저와 JVM을 선택했습니다.
- ✓ 하나는 전형적인 웹 아키텍처에 AJAX 기능 보완한 구조를 가지고 있습니다.
- ✓ 또 다른 하나는 관리자를 위한 모니터링 기능을 제공하기 위해 eclipse RCP를 선택하였습니다.



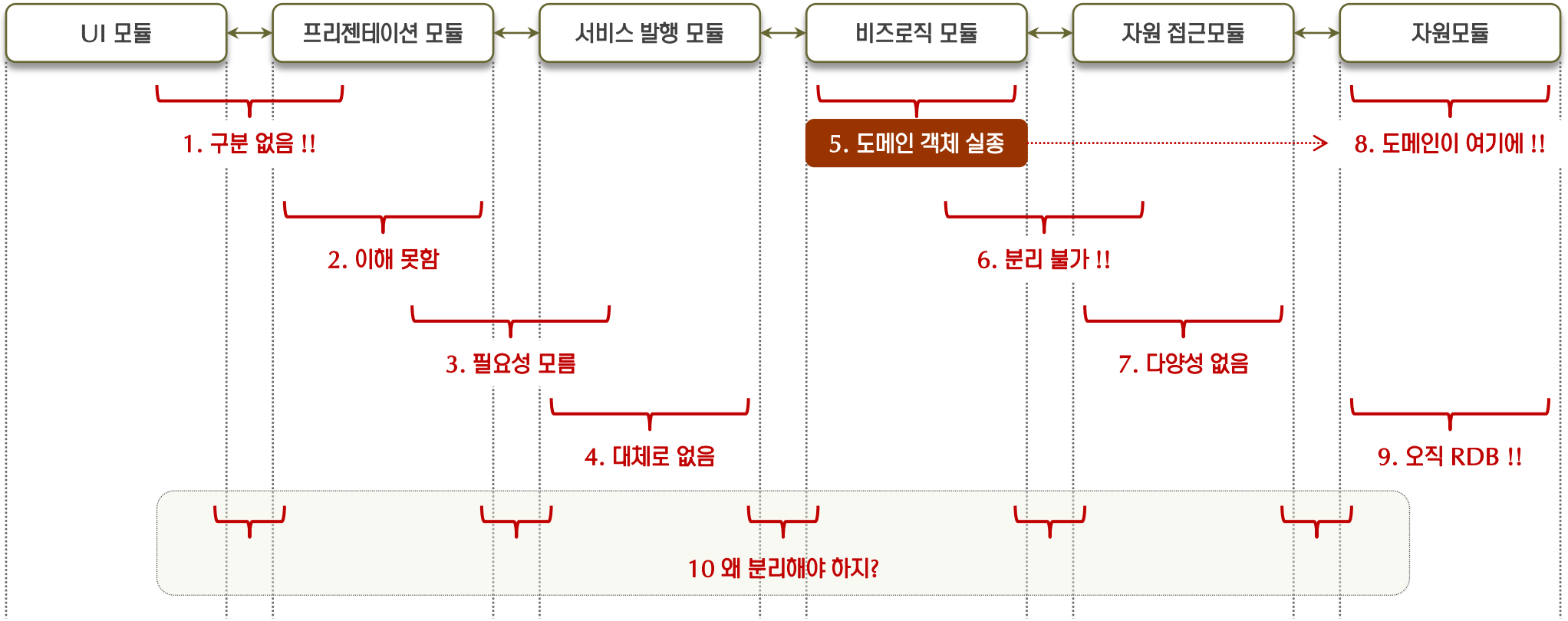
4.8 UI 레이어 모듈 (2/2)

- ✓ 다양한 UI 중에 두 가지를 선택하였습니다. 일반 사용자용 웹 UI와 관리자용 RCP UI입니다.
- ✓ 이제 여섯 개의 레이어의 모듈을 모두 [설계 | 선택] 했습니다.
- ✓ 서비스나 프리젠테이션 레이어에서 도메인 컴포넌트의 인터페이스를 사용할 것인가에 대한 선택이 남아 있습니다.
- ✓ 각 레이어 진입점(connector) 에 필요한 장치들에 대한 선택이 남아 있습니다.



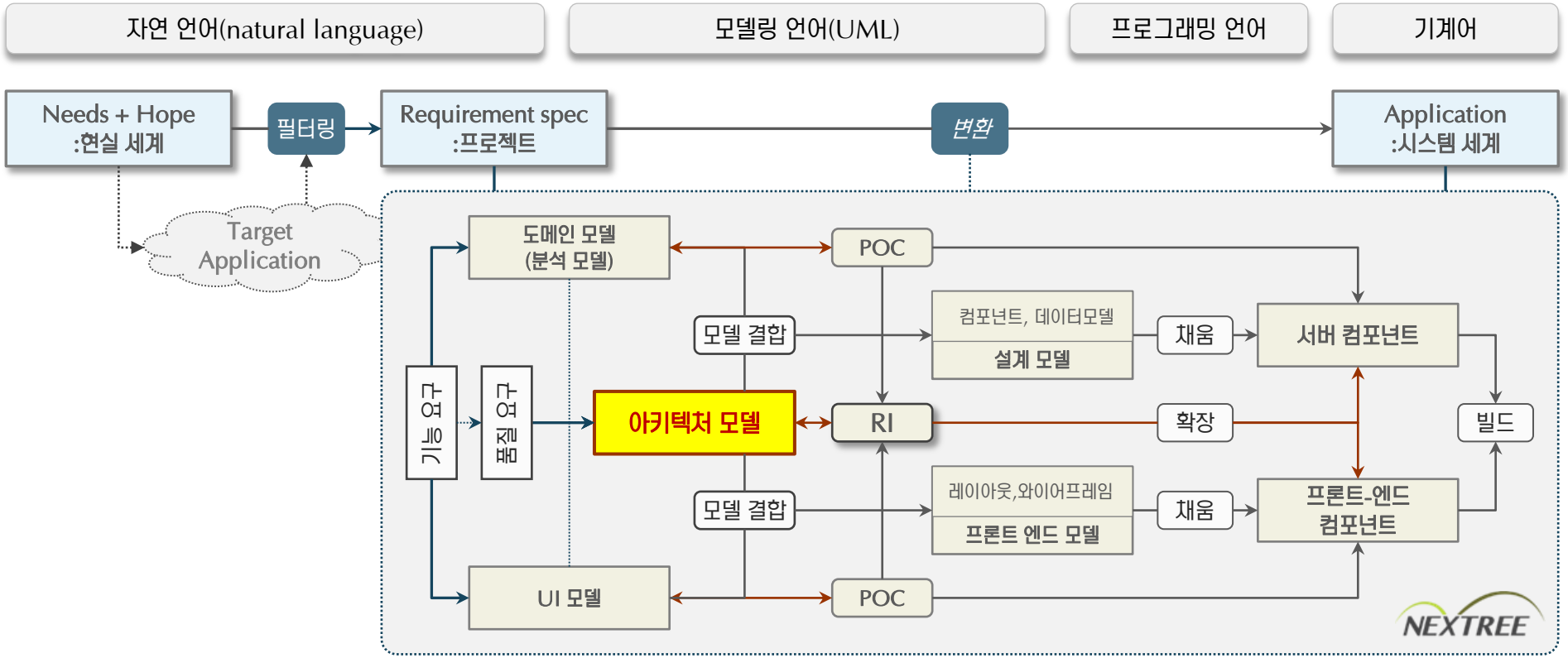
4.9 아키텍처 설계 회고

- ✓ 하지만, 우리는 레이어를 잘 분리해서 멋지게 설계한 시스템을 본적이 별로 없습니다.
- ✓ 체계적인 설계 역량이 부족하거나, 시간이 부족하거나, 관심이 부족하기 때문이겠죠.
- ✓ 아무렇게나 지어도 사람이 들어가서 살 수 있듯이, 아무렇게 개발해도 실행되기 때문입니다.
- ✓ 우리는 아키텍처 설계에 대한 고민을 많이 하지 않았으며, 그 결과로 유연성 없는 불량 시스템을 양산하고 있습니다.



5. 요약 (1/2)

- ✓ 정보를 정의하고 처리하며, 저장하고, 제시하기 위한 시스템에 대한 설계를 마쳤습니다.
- ✓ 기술 종속을 완전히 제거한 도메인 컴포넌트를 기반으로 선택하여 조합 가능한 레이어 구조로 설계를 하였습니다.
- ✓ 일련의 과정을 살펴보면, 현대의 아키텍처 설계란 것이 “Design” 보다는 “Selection”에 가까운 것 같습니다.
- ✓ SW 설계 기술이 발전할 수록 선택의 단위는 커질 것 같습니다.



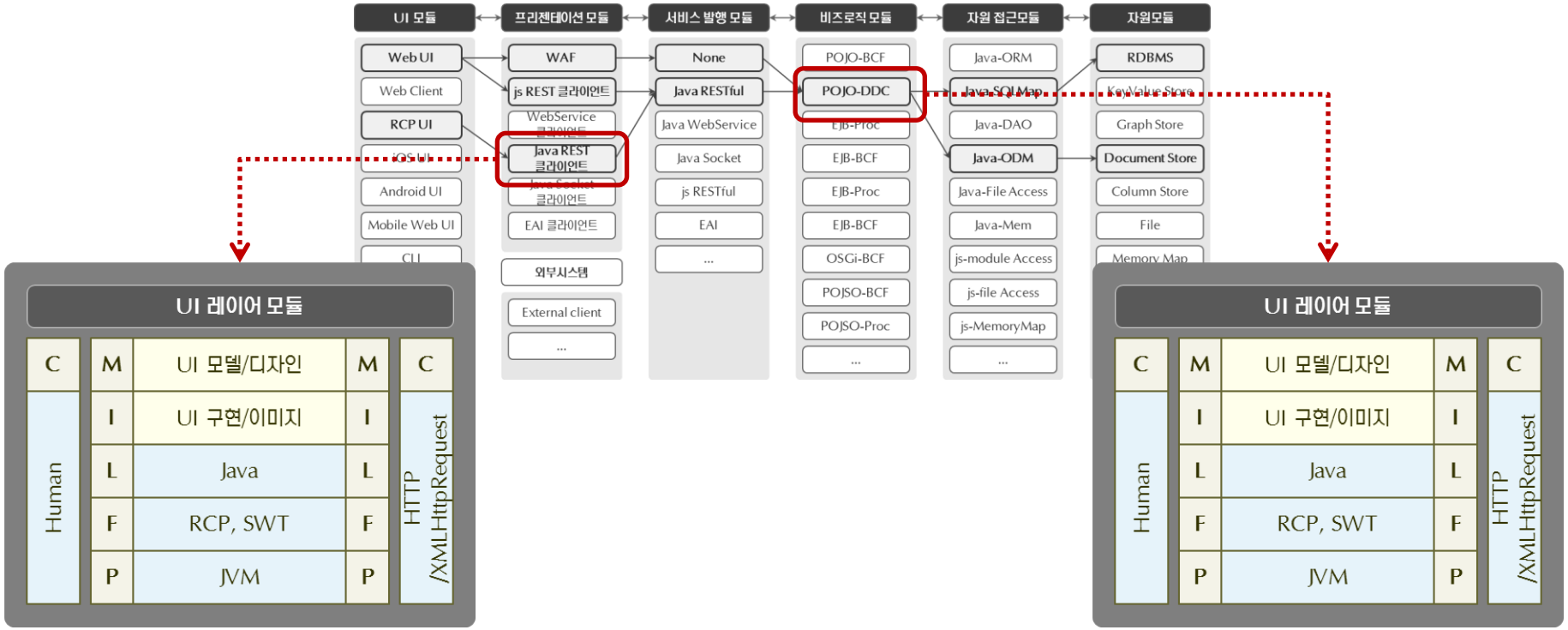
5. 요약 (2/2)

✓ 정보를 처리하는 시스템의 아키텍처 설계는,

- 레이어 개수 및 구성에 대한 선택, 레이어 모듈 유형 선택, 레이어 모듈 내부 구성 요소 선택

✓ 등과 같은 일련의 선택 활동이었습니다.

✓ 이러한 선택을 통해 설계가 가능한 것은 선택할 수 있는 아키텍처 자산이 풍부하기 때문입니다.



토의

- ✓ 질의 응답
- ✓ 토론

감사합니다....

❖넥스트리컨설팅(주)

❖CEO 송태국 / 대표 컨설턴트

❖tsong@nextree.co.kr