Mermaid 엣지케이스 가이드 [복잡도]

보고된 증상

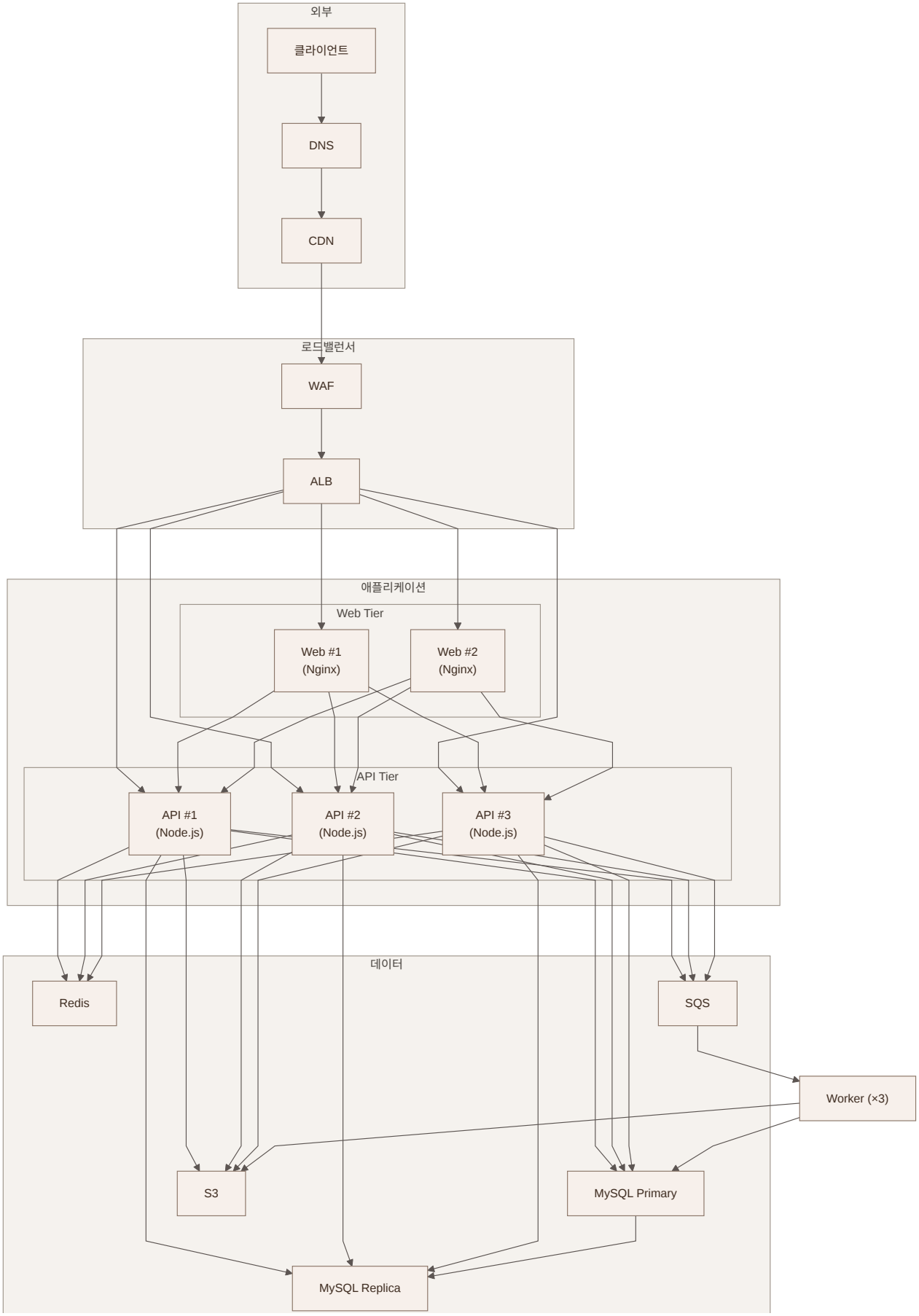
Mermaid 다이어그램에 노드를 많이 추가하고, 서브그래프를 중첩하고, `<br/>`로 줄바꿈을 넣거나 HTML 서식 태그를 사용하면 다이어그램이 의도대로 렌더링되지 않는다. 구체적으로는 노드가 예상 밖의 위치에 배치되거나, 엣지가 다른 노드를 관통하거나, 레이아웃이 작은 변경에도 급격하게 바뀌는 현상이 발생한다.

이 현상은 **Mermaid 자체의 설계 범위에서 비롯되는 한계**이다.

case 1: 복잡한 서버 구성 아키텍처를 Mermaid로 옮겼을 때

아래는 실제 서버 구성을 Mermaid flowchart로 표현한 예시이다. 외부 트래픽이 로드밸런서를 거쳐 웹 서버, API 서버, 워커로 분기되고, 각 컴포넌트가 데이터베이스와 캐시에 연결되는 일반적인 구성이다.

복잡한 다이어그램 예시:



이 다이어그램은 노드 18개, 서브그래프 5개, 엣지 20개 이상을 포함한다. 실제로 렌더링해보면 다음 문제를 확인할 수 있다.

API 서버 3대가 데이터베이스, 캐시, 스토리지에 동시에 연결되면서 엣지가 교차하고, 어떤 선이 어디에서 어디로 가는지 추적하기 어렵다. Dagre 레이아웃 엔진이 노드를 예측 불가능한 위치에 배치하며, 서브그래프 간 간격이 불균일해진다. 읽는 사람이 이 다이어그램에서 트래픽 흐름을 파악하려면 한참을 들여다봐야 한다.

동일한 구성을 분할한 예시:

동일한 정보를 2개의 간결한 다이어그램으로 나누면 각각이 훨씬 명확해진다.

Diagram1

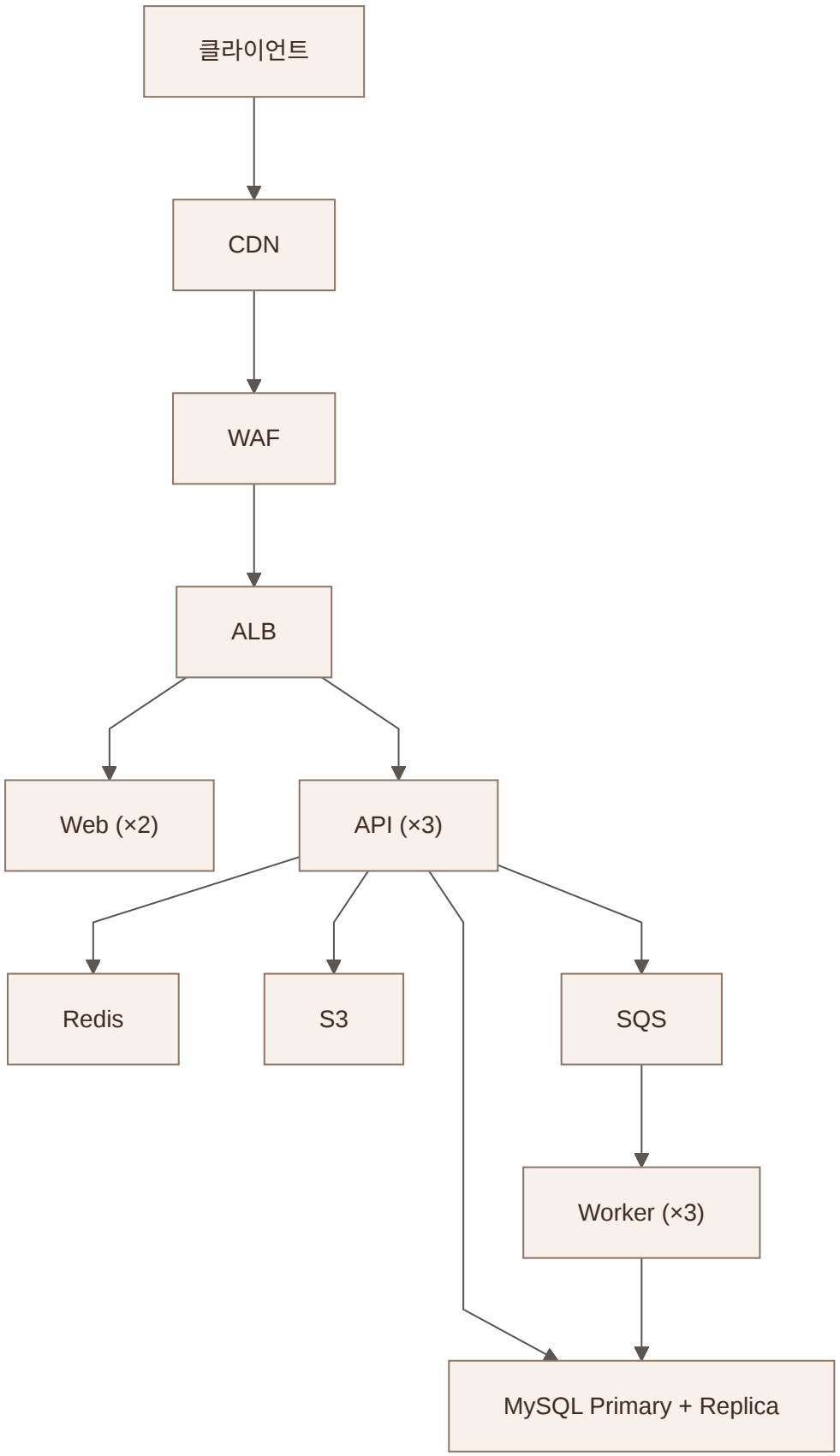
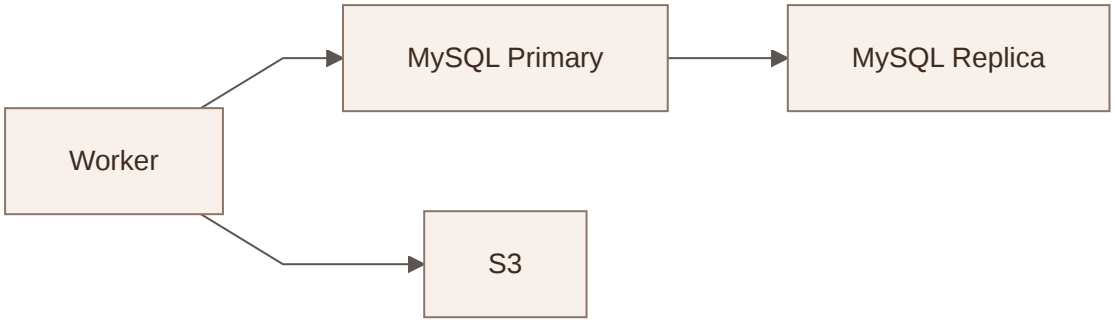


Diagram2



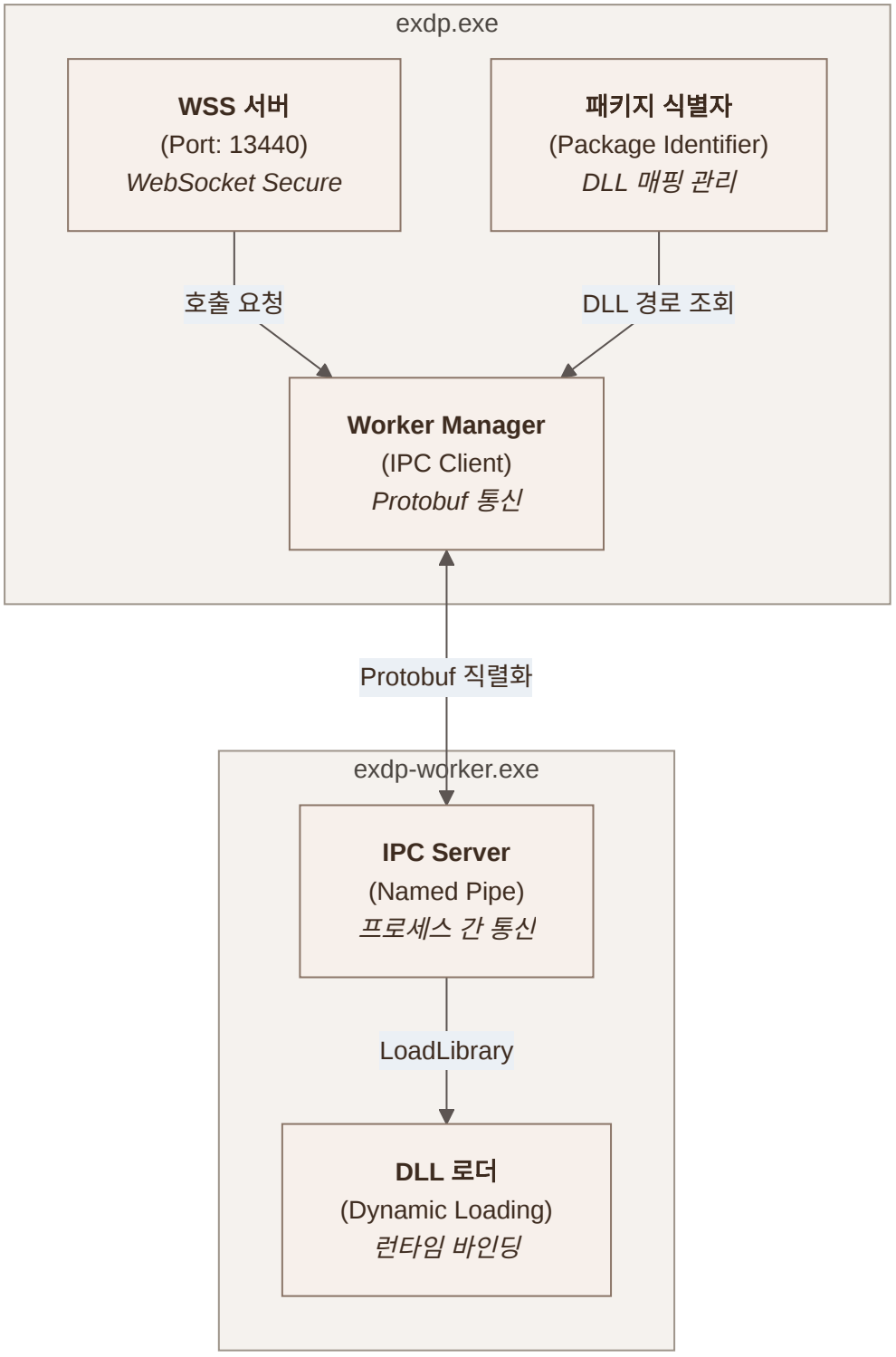
첫 번째 다이어그램은 트래픽 흐름을 5초 안에 파악할 수 있다. 두 번째는 데이터 쓰기 경로만 별도로 보여준다. 복잡한 하나보다 간결한 둘이 낫다.

case 2: HTML 라벨을 과도하게 사용했을 때

Mermaid 노드에는 `<br/>`로 줄바꿈을 넣고, `<b>`, `<i>` 등 HTML 서식 태그를 사용할 수 있다. 이 기능은 `htmlLabels: true`(기본값) 설정에서 SVG 내부의 `<foreignObject>` 태그를 통해 동작한다.

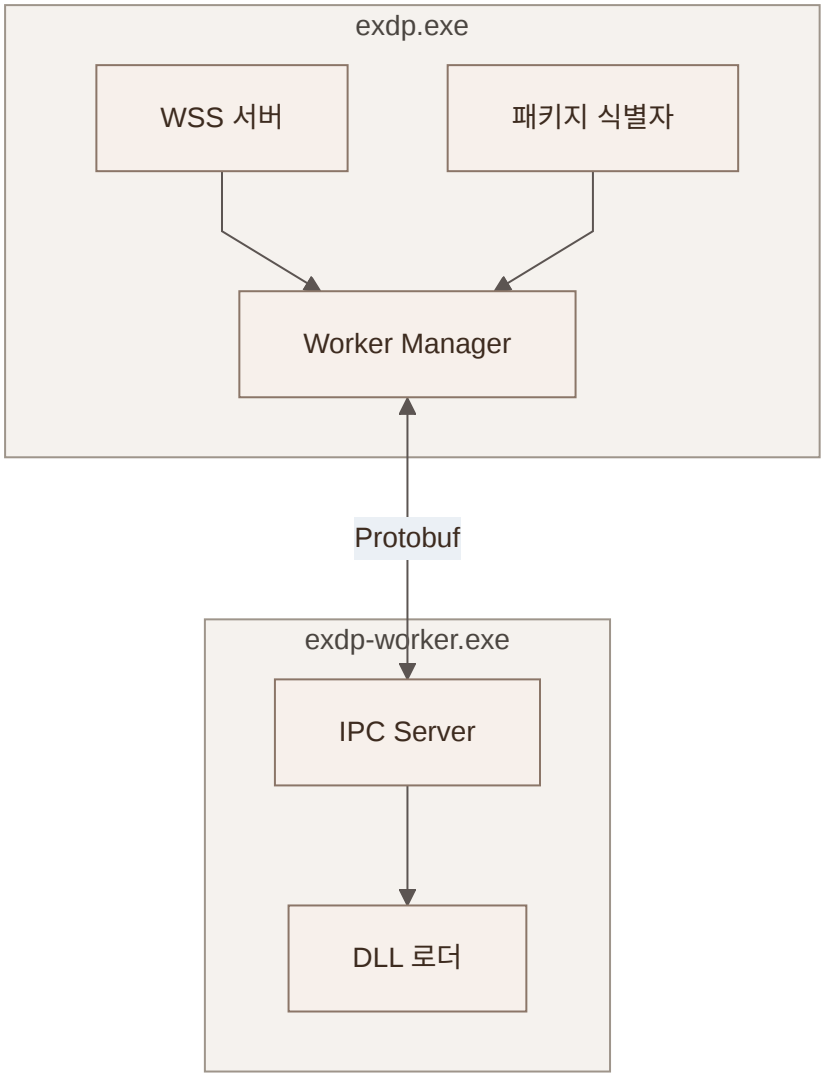
문법적으로 허용되고 동작도 하지만, HTML 라벨이 많아질수록 두 가지 문제가 심화된다.

가독성 저하 — 노드 크기 비대화:



각 노드에 `<b>`, `<i>`, `<br/>`가 3줄씩 들어가면서 노드 박스가 커지고, 실제로 중요한 **연결 관계**(화살표)가 상대적으로 눈에 들어오지 않는다. 다이어그램의 목적은 컴포넌트 간 관계를 보여주는 것인데, 각 노드의 상세 설명이 그 관계를 가린다.

동일한 구성을 간결하게 작성한 예시:



노드 라벨을 핵심 이름만 남기니 컴포넌트 간 관계가 한눈에 보인다. Port 번호, 프로토콜 상세, 기술 스택 설명은 다이어그램 아래 본문에 텍스트로 보충하면 된다.

PDF 내보내기의 추가 제약 — foreignObject:

HTML 라벨은 브라우저에서는 정상 동작하지만, PDF 내보내기에서는 별도의 처리가 필요하다. Mermaid가 `htmlLabels: true`로 생성한 SVG에는 `<foreignObject>` 태그 안에 HTML이 포함되는데, PDF 변환을 담당하는 WeasyPrint는 이 태그를 지원하지 않아 라벨 텍스트가 사라진다. BookStack에서는 이 문제를 우회하기 위해 PDF 내보내기 시 Mermaid를 PNG 이미지로 변환하여 삽입하는 방식을 채택했다. 이 과정에서 블록당 Chromium 기동 오버헤드가 발생하므로, HTML 라벨이 많은 복잡한 다이어그램일수록 내보내기가 느려진다.

Mermaid의 설계 목적

Mermaid 프로젝트의 공식 문서는 다음과 같이 명시한다:

> "The main purpose of Mermaid is to help documentation catch up with development. Doc-Rot is a Catch-22 that Mermaid helps to solve."
> — Mermaid 공식 문서 (mermaid.js.org/intro)

Mermaid는 마크다운처럼 텍스트로 빠르게 다이어그램을 작성하고, 버전 관리 가능한 형태로 문서에 포함시키기 위해 만들어졌다. 창시자 Knut Sveidqvist는 Packt 출판사 인터뷰에서 "문서화가 번거롭고 지루할 필요가 없다"고 밝혔으며, Mermaid Chart 블로그에서는 UML 다이어그램이 지나치게 포괄적이 되어 오히려 개발자의 시간을 낭비하고 있었다는 문제의식을 공유한 바 있다.

특히 Mermaid 공식 Flowchart 문서에는 스웨덴어 개념인 **"lagom"** (너무 많지도 적지도 않은 적당함)이 직접 언급되어 있다. 과도한 체이닝은 가독성을 떨어뜨리며, lagom 원칙이 적용된다는 것이다. 스웨덴인인 Knut Sveidqvist가 이 원칙을 문서에 직접 포함시킨 것은 Mermaid의 설계 철학을 단적으로 보여준다.

Dagre 레이아웃 엔진의 구조적 한계

Mermaid의 기본 레이아웃 엔진인 Dagre는 노드와 엣지가 많아질수록 배치 품질이 떨어진다. 이 문제는 Mermaid 프로젝트 자체가 인정하고 있으며, 그 결과 ELK(Eclipse Layout Kernel)라는 대체 레이아웃 엔진을 추가했다.

Mermaid 공식 문서에는 다음과 같이 설명되어 있다:

> "ELK: For those who need more sophisticated layout capabilities, especially when working with large or intricate diagrams, the ELK layout offers advanced options."
> — Mermaid Syntax Reference (mermaid.js.org/intro/syntax-reference.html)

Dagre에서 ELK로의 전환을 요청한 GitHub Issue #1969에서는, 큰 다이어그램에서 연결이 하나뿐인 블록이 다이어그램 전체를 가로질러 예상 밖의 위치에 배치되고, 엣지가 다른 블록을 관통하는 문제가 보고되었다. 이 이슈는 Knut Sveidqvist 본인이 담당자로 지정되었다.

경험이 풍부한 개발자들의 평가도 일관적이다. Korny Sietsma는 "Mermaid의 엔진은 훨씬 혼란스럽고, 작은 변경이 전체 레이아웃을 급격하게 바꾼다"고 지적했으며, 자신은 단순한 것에는 Mermaid를, 복잡하거나 레이아웃 제어가 필요한 것에는 Excalidraw를 사용한다고 밝혔다.

foreignObject와 PDF 렌더링 파이프라인

Mermaid가 `htmlLabels: true`(기본값)로 생성한 SVG에는 `<foreignObject>` 태그 안에 HTML 라벨이 포함된다. 브라우저(Chromium)는 이를 정상적으로 렌더링하지만, PDF 변환을 담당하는 WeasyPrint는 `<foreignObject>`를 지원하지 않는다.

이것은 WeasyPrint만의 문제가 아니다. draw.io 프로젝트의 Issue #3350에서도 foreignObject를 처리할 수 있는 SVG 렌더러는 브라우저뿐이라는 점이 확인되었으며, CairoSVG의 공식 SVG 1.1 지원 목록에도 foreignObject는 누락되어 있다. WeasyPrint Issue #1441에서 코어 메인테이너 Guillaume Ayoub(liZe)는 foreignObject 구현의 난이도를 언급했고, 이 이슈는 4년 이상 오픈 상태로 남아 있다.

결과적으로, Mermaid 노드에 HTML 라벨을 사용할수록 브라우저 렌더링과 PDF 출력 사이의 일관성 유지에 더 많은 처리가 필요해진다. BookStack에서는 이를 PNG 변환으로 우회했으나, 이 방식에는 블록당 Chromium 기동 오버헤드와 래스터 품질 저하라는 비용이 수반된다.

올바른 작성법 가이드

기본 원칙

Mermaid 다이어그램은 **시스템의 전체 구현을 담는 것이 아니라, 핵심 흐름을 요약하는 것**이다. 읽는 사람이 5초 안에 전체 구조를 파악할 수 있어야 한다.

적정 규모:

노드 3~15개, 서브그래프 1~2단계 중첩까지가 Mermaid의 최적 사용 범위이다. 노드가 20개를 넘어가면 다이어그램을 분할하거나, 복잡한 레이아웃 제어가 필요한 경우 draw.io, Excalidraw 등 다른 도구를 고려한다.

복잡한 다이어그램을 분할하는 방법

하나의 복잡한 다이어그램 대신, **관심사 별로 분리된 여러 개의 간결한 다이어그램**을 사용한다.

예를 들어 앞서 본 서버 아키텍처를 문서화할 때, 하나의 다이어그램에 모든 것을 넣는 대신 "트래픽 흐름", "데이터 쓰기 경로", "모니터링"으로 나누어 각각 별도의 다이어그램으로 작성하면, 각 다이어그램이 하나의 관심사에 집중하므로 읽는 사람이 필요한 정보를 빠르게 찾을 수 있다.

노드 라벨은 짧게 유지한다

`<br/>`로 한 노드에 여러 줄을 넣어야 할 정도로 텍스트가 길다면, 그 노드는 더 작은 단위로 분할하거나 약어를 사용하는 것이 낫다. Port 번호, 프로토콜 상세, 기술 스택 같은 부가 정보는 다이어그램 아래 본문에 텍스트로 보충한다.

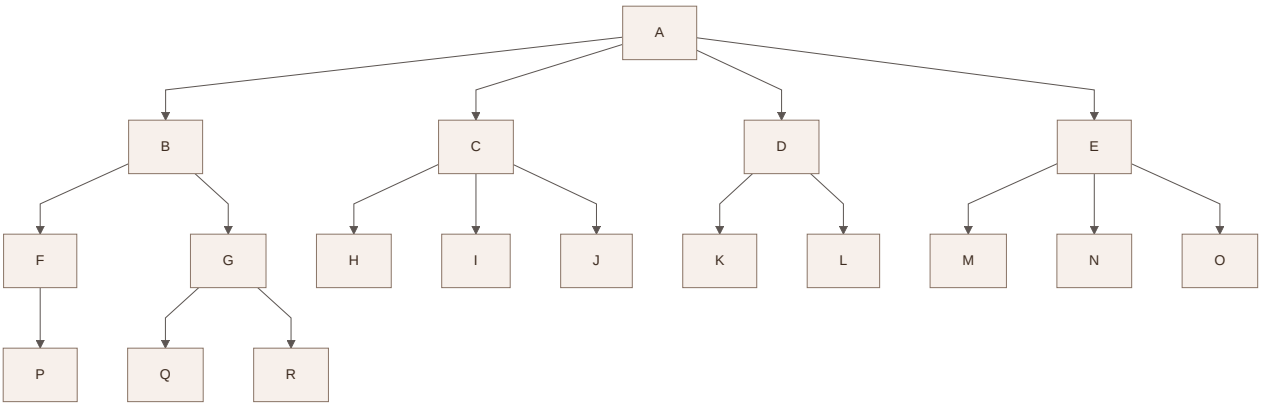
`<br/>`와 HTML 서식 태그(`<b>`, `<i>` 등)는 문법적으로 허용되며, BookStack에서 브라우저 렌더링과 PDF 내보내기 모두 정상 동작한다. 사용을 금지하는 것이 아니라, 다이어그램의 가치가 시각적 단순함에 있다는 점을 고려하여 최소한으로 사용하는 것을 권장한다.

한 페이지에 다이어그램 수를 제한한다

PDF 내보내기 시 Mermaid 블록마다 Chromium이 기동되므로, 블록이 많을수록 내보내기 시간이 선형적으로 증가한다. 한 페이지에 5개 이상의 다이어그램이 필요하다면 페이지를 분할하는 것을 권장한다.

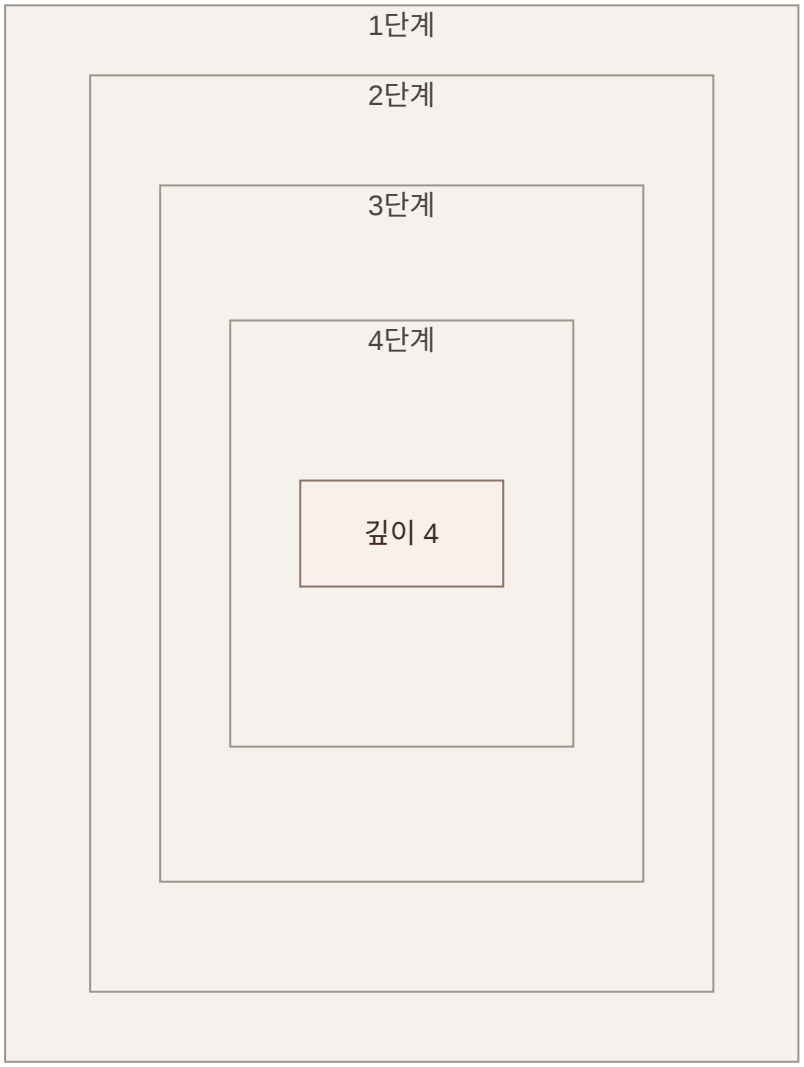
피해야 할 패턴

노드 수 과다:



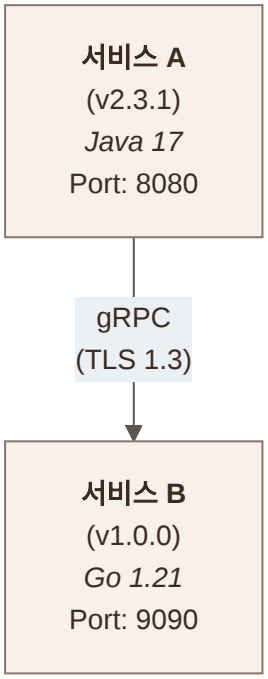
노드가 20개를 넘어가면 Dagre 레이아웃이 불안정해지고, 한 곳을 수정하면 전혀 다른 곳의 배치가 바뀌는 현상이 발생한다. 이 시점에서 다이어그램을 분할해야 한다.

서브그래프 3단계 이상 중첩:



서브그래프가 깊어질수록 레이아웃 엔진의 공간 할당이 비효율적이 되어 다이어그램의 대부분이 빈 여백으로 채워진다. 2단계까지가 가독성의 실질적 한계이다.

HTML 라벨 과다 사용:



노드 하나에 4줄씩 들어가면 노드 박스가 커져서, 정작 중요한 연결 관계(화살표)가 눈에 들어오지 않는다. 또한 이 HTML 라벨은 내부적으로 SVG의 `<foreignObject>`를 통해 렌더링되므로, 브라우저 이외의 SVG 소비자(PDF 변환기, Inkscape 등)에서 호환성 문제를 일으킬 수 있다.

다른 플랫폼에서의 동작

이 한계는 BookStack만의 문제가 아니라 Mermaid를 사용하는 모든 플랫폼에서 동일하게 발생한다.

플랫폼 / 도구	Mermaid 지원	동일 복잡도 한계
GitHub (GFM)	O	O
GitLab	O	O
Notion	O	O
Obsidian	O	O
VS Code (미리보기)	O	O
Mermaid Live Editor	O	O

Mermaid의 복잡도 한계는 Dagre/ELK 레이아웃 엔진의 동작 방식에서 비롯되므로, 어떤 플랫폼에서 렌더링하든 동일하게 적용된다.

간단한 확인법: Mermaid Live Editor(https://mermaid.live)에서 다이어그램을 먼저 작성해보고, 레이아웃이 의도대로 나오는지 확인한 후 BookStack에 붙여넣는다. Live Editor에서 깨지면 BookStack에서도 깨진다.

레퍼런스

문서	URL
Mermaid 공식 문서 — About	https://mermaid.js.org/intro/
Packt 인터뷰 — Knut Sveidqvist	https://partnerships.packt.com/31533-2/
Mermaid Chart 블로그 — UML 한계	https://mermaid.ai/blog/posts/uml-diagram-tool
Mermaid 공식 Flowchart 문서 (lagom 원칙)	https://mermaid.js.org/syntax/flowchart.html
Mermaid Syntax Reference — ELK 레이아웃	https://mermaid.js.org/intro/syntax-reference.html
GitHub Issue #1969 — Dagre→ELK 전환 요청	https://github.com/mermaid-js/mermaid/issues/1969
Korny's Blog — Revisiting Mermaid.js	https://blog.korny.info/2025/03/14/mermaid-js-revisited
WeasyPrint Issue #1441 — foreignObject 미지원	https://github.com/Kozea/WeasyPrint/issues/1441
CairoSVG SVG 1.1 지원 현황	https://cairosvg.org/svg_support/
draw.io Issue #3350 — foreignObject 생태계 한계	https://github.com/jgraph/drawio/issues/3350
Mermaid Issue #2688 — foreignObject→SVG 전환 요청	https://github.com/mermaid-js/mermaid/issues/2688