

BACKEND ENGINEER · APPLICATION DESIGN

업무의 요구와 제약을 설계와 구현으로 연결합니다

정동건 · DKun

백엔드 개발 · 애플리케이션 설계

접근 조건에 따른 서버 책임, 기존 계산 규칙을 지키는 데이터 처리, 다른 개발자가 적용할 공통 기능을 설계하고 구현했습니다.

이 포트폴리오에는 경찰청 자율주행관제 플랫폼, ITS 성능평가 시스템, FTMS 공통 지도 라이브러리에서 내린 판단과 구현 과정을 담았습니다.

세 사례에서 설명할 설계 판단

사례	해결해야 했던 문제	설명할 판단	페이지
경찰청 자율주행관제 플랫폼	DB·화면·외부 연계의 접근 조건과 통신 방식이 다름	서버 책임과 DB 접근 경로를 나누고 외부 데이터를 업무 데이터로 변환	03-06
도로공사 ITS 성능평가 시스템	기존 계산 규칙을 유지하면서 장비별 차이와 편집 상태를 처리해야 함	공통 처리와 장비별 분석을 분리하고, 변경된 입력과 저장 결과의 관계를 관리	07-10
FTMS 공통 지도 라이브러리	여러 프로젝트가 같은 지도 기능을 서로 다른 화면에서 사용	공통 API와 화면별 옵션을 구분하고, 실제 사용 요구를 배포 기능으로 반영	11-13

보완 사례: 청라하늘대교의 서버 기본 구조·샘플 제공과 SalesBridge의 연계 설명·인수인계 경험을 14페이지에 정리했습니다.

경찰청 자율주행관제 플랫폼의 서버 설계와 외부 연계

엑스콜로 · 2024.02-2026.05 / Java · Spring · Kafka · PostgreSQL · Docker



서비스 맥락

자율주행 차량 운행 정보와 도로 상황, 교차로 정보를 지도에서 확인하는 관제 화면입니다.

화면 요청과 외부 연계의 접근 조건을 구분하고, 업무 서버의 데이터 처리 책임을 정했습니다.

서비스 화면 · 지도 화면 구현은 다른 개발자 담당 · 촬영 시점은 개인 참여 기간과 구분

개인 담당: 기술스택 선정·초기 서버 구조 설계, 업무 서버·외부 연계 구현, Docker 배포·운영 대응

초기 책임 분리와 CCTV API의 실제 호출 경로

접근 구간별 진입점을 나누고, 업무 데이터 접근은 Business Server로 모으는 초기 구조를 설계했습니다.



코드로 확인되는 CCTV 요청 경로

`CctvRepository (@FeignClient)` → `Business Server /cctvs` → `Service · DAO / MyBatis` → `DB`

초기 논리 설계입니다. 이후 다른 담당자가 개발한 TCP 서버에는 직접 DB 접근 예외가 있습니다.

BER 메시지를 해석해 차량 DTO와 저장 로직에 연결

Kafka Consumer에서 받은 Hex 데이터를 BER로 디코딩하고, ASN.1의 위치 정보를 업무 DTO로 변환했습니다.



ExternalKafkaConsumer의 실제 처리 코드 발췌

```
ASN1Primitive root = berDecoderService.decodeFromHex(eventJsonString);
Pol3AdVehDto dto = proveVehicleDataParser.parse(root);
boolean success = pol3Service.insertProveVehicleData(dto);
```

파싱 경계

snapshots → thePosition → latitude / longitude 등 위치 항목 추출
계층형 JSON은 별도 경로에서 업무 DTO와 MyBatis 저장 로직으로 연결했습니다.

근거: ExternalKafkaConsumer · ProveVehicleDataParser. 코드 발췌는 로그·예외 처리 부분을 생략했습니다.

로그 누적 문제를 출력과 보관 정책으로 조치했습니다

자체 개발 서버와 고객사 운영 서버의 Docker 환경을 구성하고 수동으로 배포했습니다. 운영 과정에서 로그가 누적되는 문제를 확인해, 어떤 로그를 출력하고 얼마만큼 보관할지 설정을 조정했습니다.

조치 대상	적용한 변경
콘솔 로그	출력 수준을 WARN으로 조정
파일 로그	콘솔 출력과 구분해 파일 로그 관리
SQL 상세 로그	상세 출력 비활성화
Docker 로그	회전 크기와 보관 파일 개수 제한

로그 누적에 대응해 출력 범위와 보관량을 제한했습니다. 서버 분리 이후의 운영에서는 각 서버의 배포 설정뿐 아니라 로그 관리 정책도 함께 관리해야 했습니다.

ITS — 계산 규칙을 유지하면서 처리 구조를 설계했습니다

도로공사 ITS 성능평가 시스템 · 엑스콜로 · 2025.12-2026.05

Java · Spring Boot · JPA · Querydsl

기존 규칙 유지 · 처리 책임 분리

기존 시스템의 계산식과 수치 처리 규칙을 유지해야 했습니다. 동시에 장비마다 입력 데이터와 계산·결과 형식이 달라, 공통 업무 흐름 안에서 이 차이를 처리할 구조가 필요했습니다.

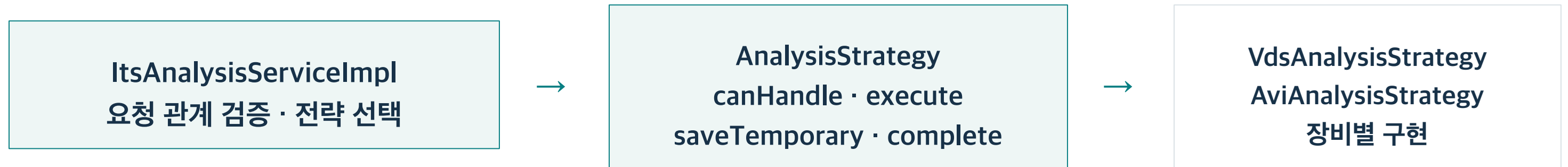
수집·분석 영역에서 공통 API와 장비별 분석 전략의 책임, 작업 데이터와 확정 결과의 처리 규칙을 설계하고 구현했습니다. 다른 담당자의 파일 처리 모듈은 이 흐름에 연동했습니다.

개인 담당: 수집·분석 영역의 설계·개발 리딩 및 구현.

협업 범위: 기획, 파일 처리 모듈, 프론트엔드와 운영 배포는 다른 담당자가 수행했습니다.

장비 선택은 공통 서비스에서, 분석 처리는 장비별 전략에서

기존 계산 규칙은 유지하면서, 공통 업무 검증과 장비별 입력·분석·저장 처리를 분리했습니다.



분석 실행 경로 · 실제 코드 발췌

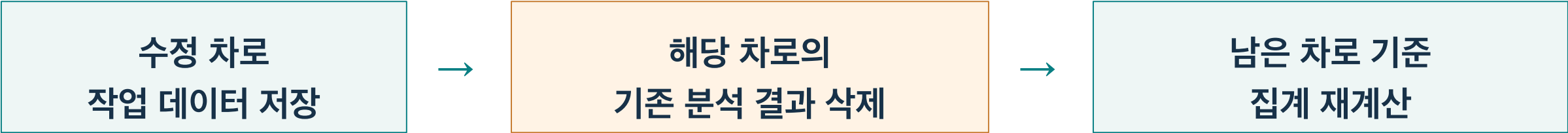
```
analysisStrategies.stream()
    .filter(strategy -> strategy.canHandle(request.getEquipmentType()))
    .findFirst()
    .map(strategy -> strategy.execute(request))
```

공통 서비스의 요청 검증 뒤 장비 전략을 선택합니다. 발췌 코드에는 미지원 장비 예외 처리를 생략했습니다.

입력 변경 시 기존 결과를 무효화하고 서버에서 재계산

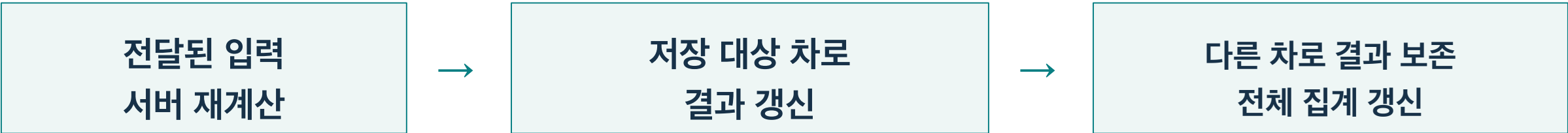
VDS 작업 데이터와 확정 결과를 구분했습니다. 수정 차로의 이전 결과가 새로운 입력과 함께 남지 않도록 처리했습니다.

임시저장



saveTemporary: modified=true인 차로를 선택해 결과를 삭제하고 집계를 다시 계산

결과 저장



complete: 저장 대상 차로의 결과만 교체하고, 다른 차로의 저장 결과를 포함해 집계

VdsAnalysisStrategy 구현을 재구성한 설명용 처리도. 입력·계산 수치의 실제 테스트 결과를 나타내는 그림은 아닙니다.

기존·신규 비교에서 발견한 대상교통량 차이를 수정

테스트 담당자의 비교 리포트를 받아 판단 기준을 확인하고 수정했습니다. 배포 반영 후 재비교에서 해당 수치를 맞췄습니다.

수정 전 신규 화면 · 대상교통량 33

사용 여부	주기	측정시간		표준편차 정확도		
		시작 시각	종료 시각	기준 교통량(Yi)	대상 교통량(Xi)	교통량 MAPE
☐		11:15:00	11:19:59	0	0	0.00
☒	1	11:20:00	11:24:59	41	33	0.20
☒	2	11:25:00	11:29:59	34	31	0.09
☒	3	11:30:00	11:34:59	30	27	0.10
☒	4	11:35:00	11:39:59	39	36	0.08
☒	5	11:40:00	11:44:59	31	25	0.19
☒	6	11:45:00	11:49:59	35	30	0.14
☐						

기존 기준 화면 · 대상교통량 34

주기분석정보 [8/6] 동영상1 보기 동영상2 보기 그래프 보기 DATA ☐ 수동 처리								
			교통량 정확도			속도 정확도		
주기	시작시각	종료시각	기준교통량(Yi)	대상교통량(Xi)	교통량 MAPE	기준속도(Yi)	대상속도(Xi)	속도 MAPE
	11:15:00		0	0	0.0000	0.00	0.00	0.0000
1	11:20:00		41	34	0.1707	76.13	93.26	0.2250
2	11:25:00		34	31	0.0882	73.89	166.00	1.0000
3	11:30:00		30	27	0.1000	75.67	76.54	0.0114
4	11:35:00		39	36	0.0769	75.21	77.02	0.0240
5	11:40:00		31	25	0.1935	73.70	95.21	0.2918
6	11:45:00		35	30	0.1428	76.21	77.07	0.0112
	11:50:00		0	0	0.0000	0.00	0.00	0.0000
			210	183	87.13	75.13	97.52	73.94

문제: 대상교통량 수치 차이

판단·조치: 기존 판단 기준에 맞춰 수정

확인: 배포 반영 후 재비교

수정 전 비교 리포트의 강조 표시를 정돈한 이미지입니다. 수정·재비교 후 해당 수치의 일치를 확인했습니다. 운영 배포는 다른 담당자가 수행했습니다.

지도 — 공통 기능과 화면별 사용 방식을 나눴습니다

FTMS 공통 지도 라이브러리 · 엑스콜로 · 2024.03-2026.06
초기 개발 및 후속 개선·지원 · React · TypeScript · OpenLayers · Vworld · Rollup

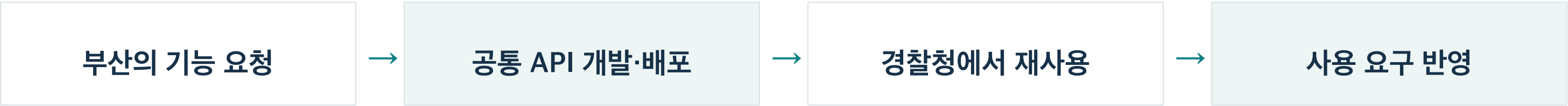
후속 FTMS 프로젝트에서도 지도 표현과 마커·링크, 이벤트·오버레이가 반복적으로 필요할 것으로 봤습니다. 이를 공통 라이브러리로 제공하되, 각 화면의 업무 데이터와 동작은 옵션·콜백으로 연결하도록 구성했습니다.

라이브러리에서 제공	사용하는 프로젝트에서 연결
배경지도·지도 표현과 벡터 표시	업무별 지도 데이터
공통 이벤트·오버레이·스타일	화면별 옵션·스타일과 이벤트 콜백
측정·영역 선택 기능과 제어 API	측정·선택 결과를 사용하는 화면 동작

설계·개발·배포와 예제·교육을 맡았습니다. 경찰청·화성 ITS의 지도 화면은 각 프로젝트의 프론트엔드 개발자가 구현했습니다.

프로젝트 요구를 공통 API로 반영하고 재사용

부산 프로젝트의 폴리라인·면적 측정 요청을 공통 기능으로 제공했고, 경찰청 고도화에서 재사용하며 후속 요구를 반영했습니다.



라이브러리에서 제공

지도 상태·조작 API
측정 스타일과 팝업 상태 콜백
GeoJSON·영역 선택 API

사용하는 개발자가 연결

화면의 선택·측정 모드 전환
업무별 결과 표시와 팝업 내용
선택 좌표와 업무 기능 연계

v0.2 계열에서 상태·측정 제어를 확장하고, v0.3 계열에서 GeoJSON·클릭·영역 선택을 보강했습니다.

지도 예제로 API 조합을 설명하고 다른 개발자의 적용 지원

사용 가이드의 신호등 모니터링 예제 · 로컬 실행 화면



화면으로 설명하는 API 조합

Marker: 장비 위치·상태 표현
FeatureSelect: 클릭·선택 처리
Overlay: 업무별 팝업 구성
Animation: 이상 상태 강조

적용 결과

경찰청·화성 ITS: 다른 개발자가
라이브러리로 지도 화면 구현
청라하늘대교: 운영 서비스 적용

마커 선택과 업무 팝업 연결을 설명하는 사용 가이드 예제입니다. 예제 개발·교육·기술 지원을 담당했습니다.

초기 개발 기반 제공과 앱 연계·인수인계

청라하늘대교 초기 개발 기반

아키텍처 설계 참여 → 서버 기본 구조·샘플 코드 제공

담당 범위는 초기 서버 설계 참여와 개발 기반 제공까지입니다.

SalesBridge 앱 연계와 인수인계

Android 하이브리드 앱 개발·검수 → 웹 개발자 연계 설명 → 고객사 전달·앱 인수인계

2025.09-2026.02, 웹페이지를 표시하는 앱의 기능 개발과 웹 개발자에게 연계 방식을 설명하는 일, 검수·전달·인수인계를 맡았습니다.

WORKING TOGETHER

설계의 이유를 설명하고, 구현과 적용까지 말합니다

설계할 때는 처리 책임과 데이터 흐름을 정하고, 선택의 이유와 관리 부담을 함께 살핍니다. 공통 기능은 코드와 예제로 설명하고, 다른 개발자가 적용하면서 겪는 문제도 함께 해결합니다.

대화는 편하게, 기술적 판단은 근거를 갖고 나눕니다.

정동건 · DKun
dkjung0824@gmail.com

[추가 자료: 전체 참여 프로젝트와 추가 기록 — Notion](#)