

IIP-모니터링 UPGRADE

트래킹 적재 및 조회 표준 SPEC 및 SW 설계

목표

기존 트래킹 메시지 처리 방식의 단점을 개선하고 런타임 **APP** 과 모니터링 **APP** 을 최대한 디커플링될 수 있도록 설계하여 모니터링 성능 개선 및 기능 확장이 용이하도록 한다.

설계 POINT

- ❑ AS-IS 스펙 유지
- ❑ 대량 DATA 처리
- ❑ IIP 버전 UP (3.0 → 4.0)
- ❑ 프레임워크
- ❑ 경량화
- ❑ 쉽게

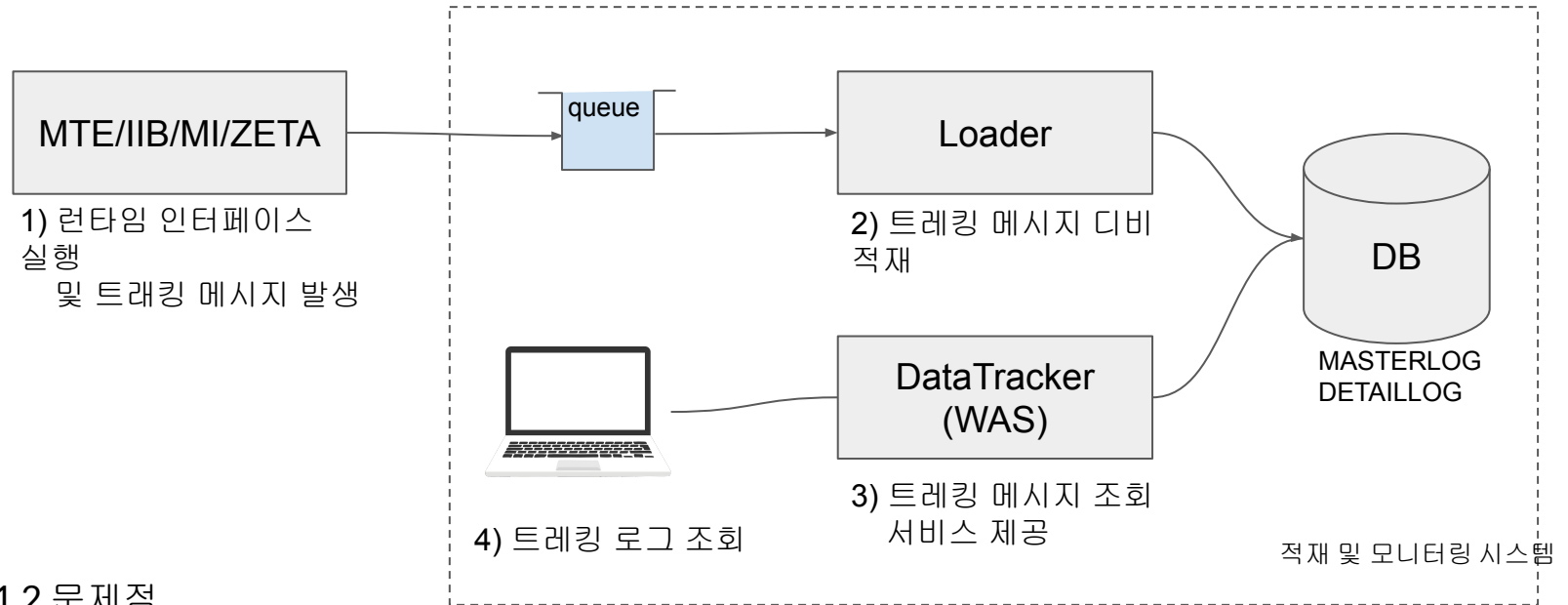


할 일

1. 트레킹 메시지 적재 프로세스 설계
 - 1.1.AS-IS 프로세스
 - 1.2.문제점
 - 1.3.TO-BE 프로세스
 - 1.4.개선점
 - 1.5.상세 프로세스 설계
 - 1.6.전체 프로세스
 - 1.7.다채널 INPUT
2. 조회 프로세스 설계
 - 2.1.caching 프로세스
3. 트레킹 메시지 구조 설계
 - 3.1.메시지 구조
 - 3.2.TrackingMessage 정의
4. 테이블 설계
 - 4.1.AS-IS 테이블 검토
 - 4.2.TO-BE 테이블 설계
 - 4.3.데이터 처리 프로세스
5. SW 설계
 - 5.1.개발 기능 리스트
 - 5.2.아키텍처, F/W
 - 5.3.대량 데이터 처리
 - 5.4.기능 상세 설계
6. 레퍼런스

1.트레킹 메시지 적재 프로세스 설계

1.1.AS-IS 프로세스

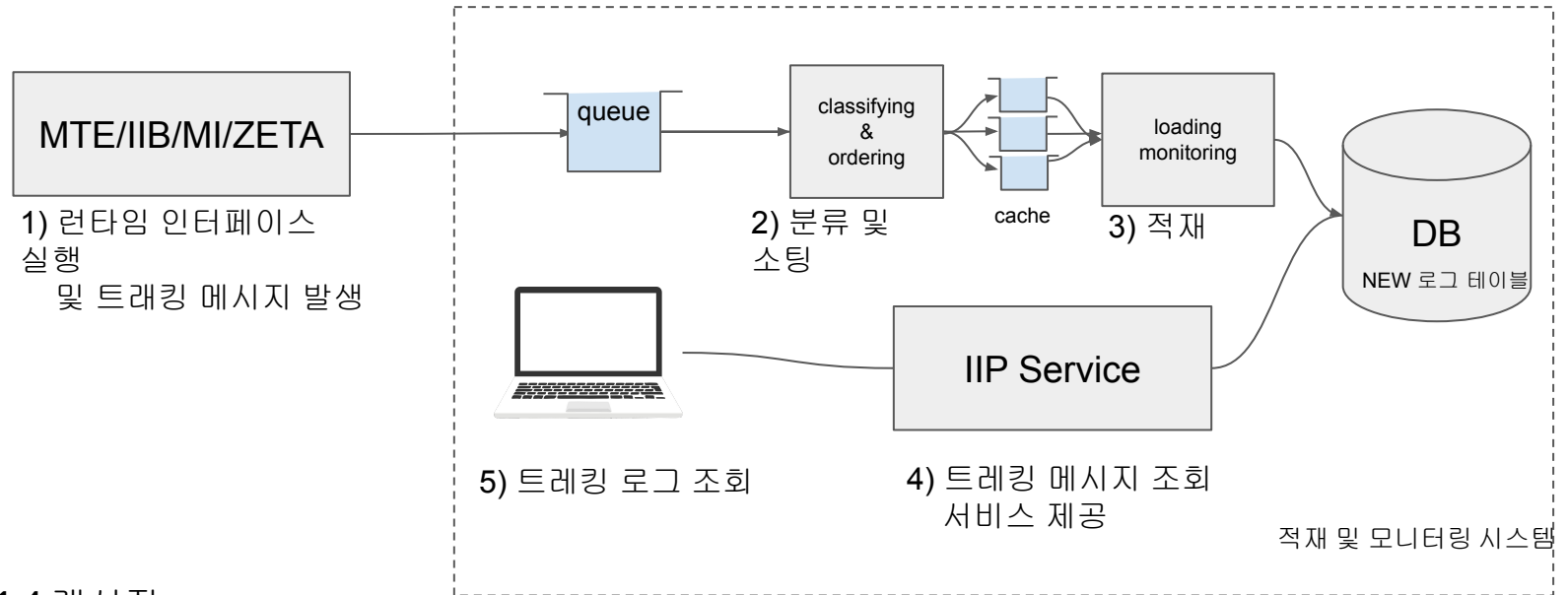


1.2.문제점

- ❑ Loder의 트레킹 메시지 병렬 처리시 병목 발생으로 인한 성능 지연 현상
- ❑ Loder의 사이트별 기능 및 성능 확장을 위한 무분별한 커스텀 코드 추가로 소스 관리 부담
- ❑ 결과 업데이트를 수반하는 마스터 디테일 구조의 테이블 구조는 성능 개선의 한계점 존재
- ❑ 대량의 트레킹 정보를 인터페이스 정보와 함께 조회시 조회 성능 저하

1.트레킹 메시지 적재 프로세스 설계

1.3.TO-BE 프로세스

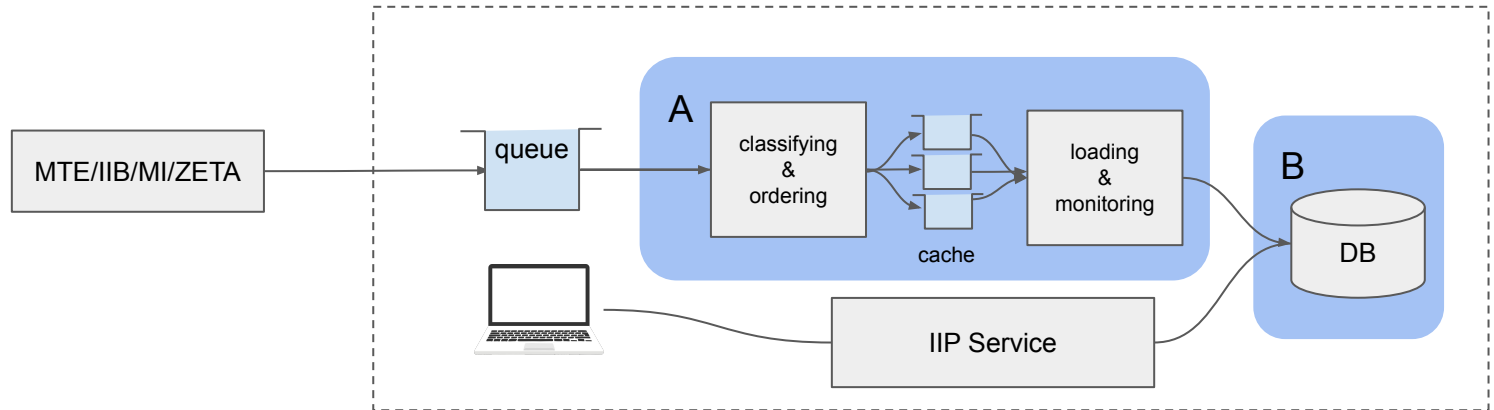


1.4.개선점

- ❑ 2) 분류 & 소팅 3) 적재 처리 분리를 통한 병렬처리로 병목 개선
- ❑ 프로세스 모듈화를 통한 기능 확장
- ❑ 캐시처리를 통한 성능 증대
- ❑ 새로운 로그 테이블 설계를 통한 적재 및 조회 성능 개선

1.트레킹 메시지 적재 프로세스 설계

1.5.상세 프로세스 설계

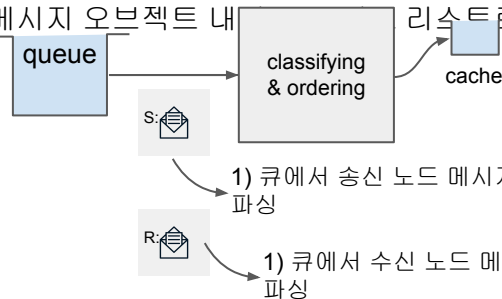


A. classifying & ordering, cache, loading, monitoring 프로세스 처리

1) classifying & ordering, caching 프로세스 처리

queue 로 부터 읽어 들인 메시지를 파싱 후 트레킹 메시지 오브젝트 생성 후 트레킹 ID 를 키 값으로 캐시에 보관한다.

인터페이스가 처리되는 과정에서 발생하는 송신, 허브, 수신 등 복수 개의 노드 처리 메시지들은 하나의 트레킹 메시지 오브젝트 내 리스트로 관리한다.



2) cache에 저장된 오브젝트가 있는지 체크 후 없으면 TrackingMessage 오브젝트

신규 생성 후 node 추가 및 상태 변경 후 cache에 보관하기

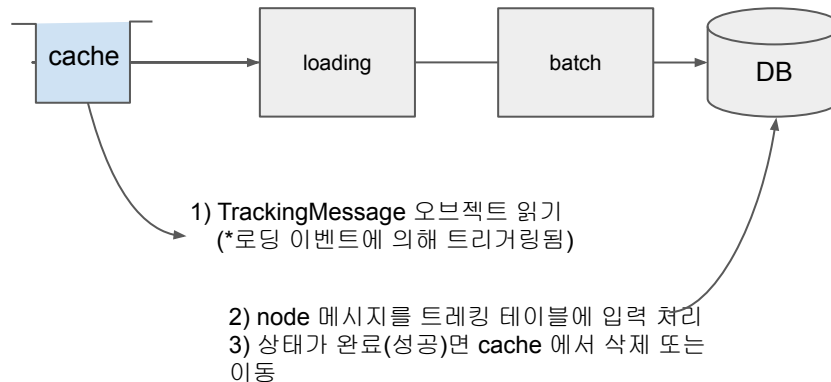
3) 로딩 이벤트를 발생시킨다.(트레킹 메시지 DB 테이블 적재를 위한 이벤트 발생)

1.트레킹 메시지 적재 프로세스 설계

1.5.상세 프로세스 설계

2) loading 프로세스 처리

로딩 이벤트에 의해 트리거링되는 **loading** 프로세스 는 **cache** 에서 **TrackingMessage** 오브젝트를 읽어 들어 디비에 적재한다.



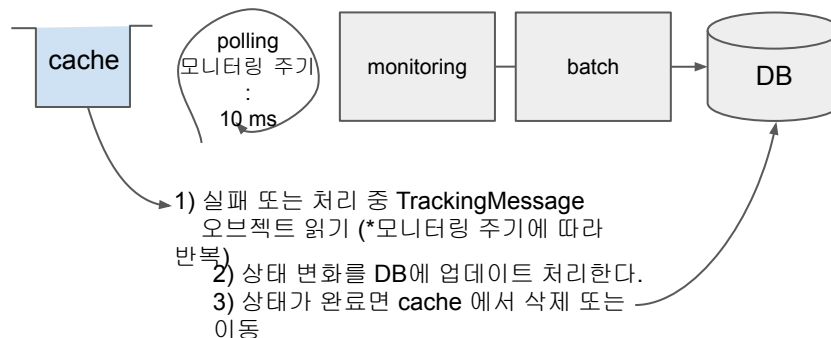
🧐 think about

- ❑ persistent cache
VM 이 셧다운되어도 작업중인 메시지는 유지될 수 있도록 한다.

3) monitoring 프로세스 처리

“실패” 상태 처리를 위한 프로세스

“실패” 상태의 **TrackingMessage** 를 읽어 DB에 업데이트 한다.



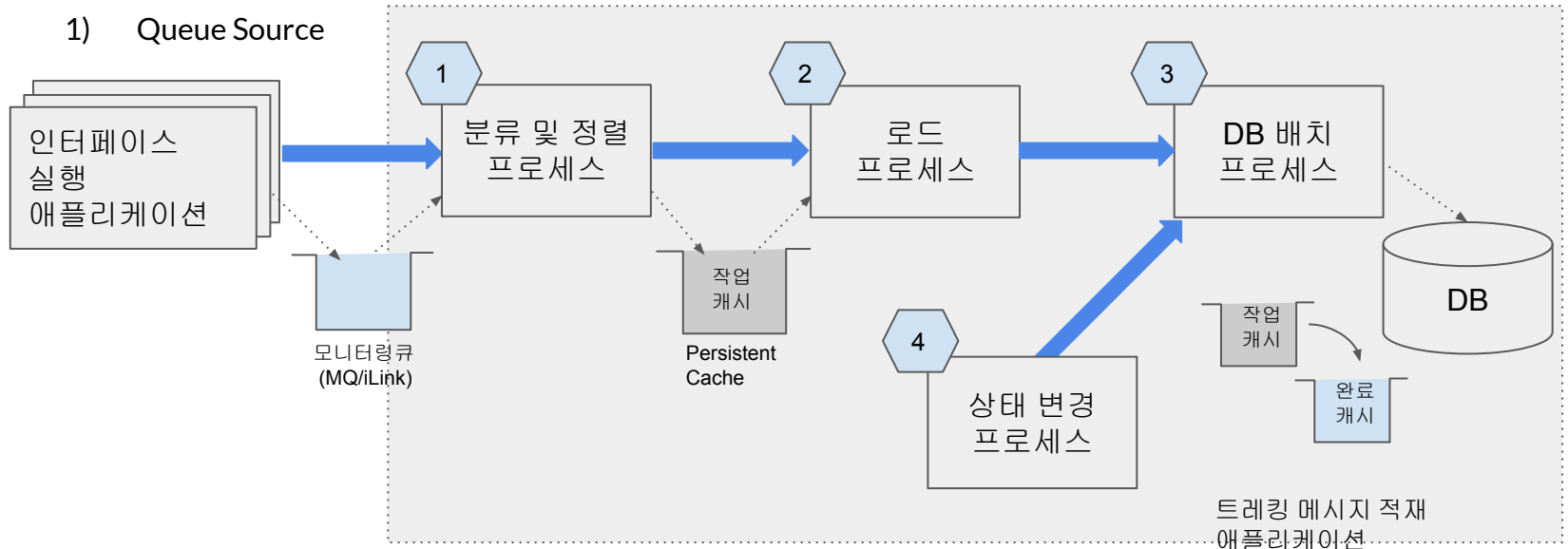
🧐 think about

- ❑ cache 동시접근 제어
- ❑ DB 입력처리에 대한 스레드 제한(성능 고려)
- ❑ cache 메시지 **polling** 시 메시지가 존재하지 않을 경우에만 모니터링 주기 만큼 대기

1.트레킹 메시지 적재 프로세스 설계

1.6.전체 프로세스

1) Queue Source



1

- **TrackingMessage** 오브젝트를 만들어 캐시에 저장한다.
- 캐시는 비휘발성 특성을 가진다.(VM 재 기동 시에도 오브젝트는 삭제되지 않는다.)

2

- 캐시에 저장된 **TrackingMessage**를 읽어들이어 인터페이스 정보를 추가하고 노드상태 정보를 수정한다.(인터페이스 정보는 전처리된 메모리 캐시에서 참조한다.)
- 최초 발생 건이면 **DB** 배치프로세스로 **TrackingMessage** 오브젝트를 전달하여 배치 처리 요청

3

- **TrackingMessage** 오브젝트 를 **DB** 배치 처리한다.
- 최초 건 **INSERT**
- 상태 변경 건 **UPDATE**
- 지정된 커밋 카운트에 따른 배치 처리
- 완료된 **TrackingMessage** 오브젝트는 작업 캐시에서 완료 캐시로 이동

4

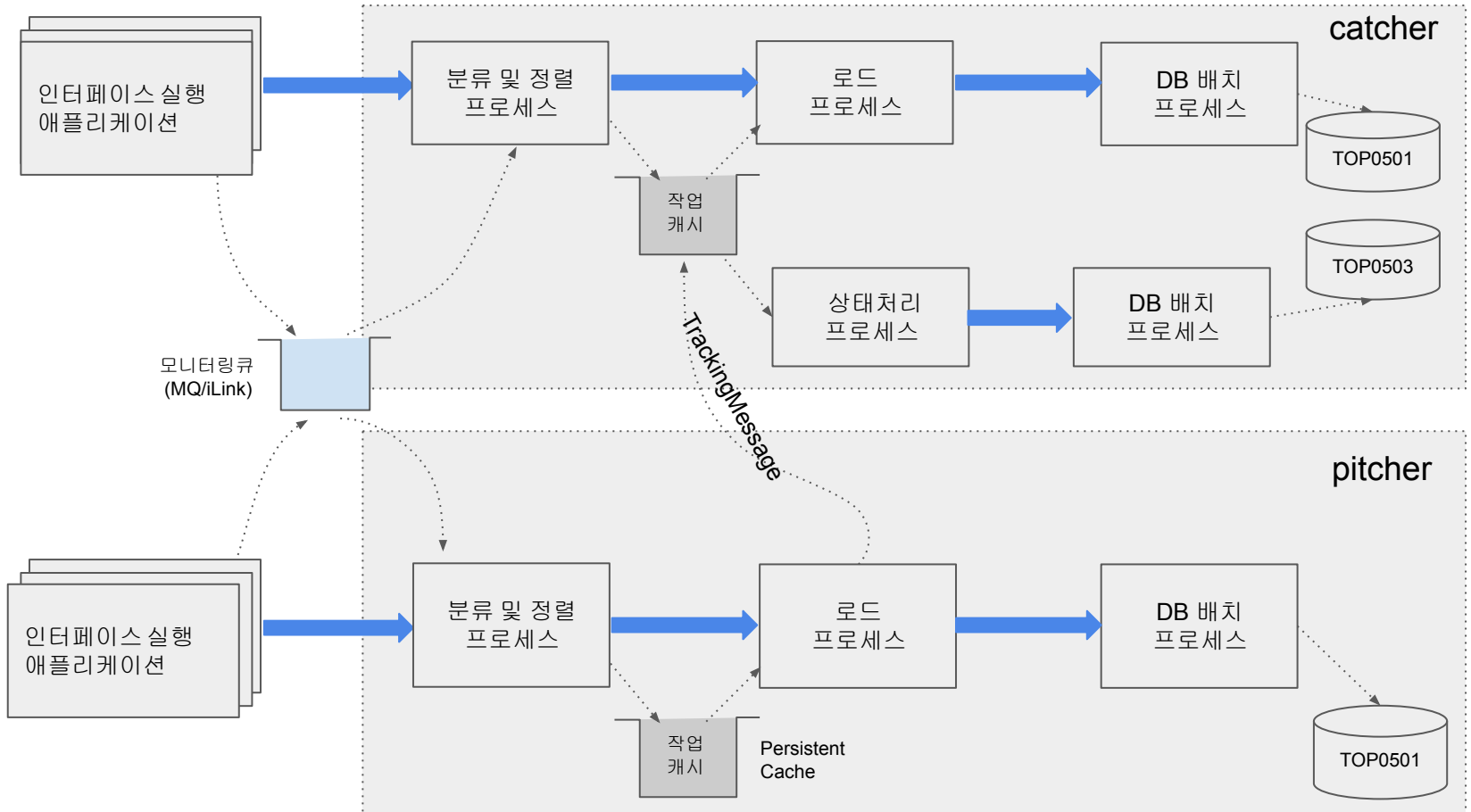
- 작업 캐시에서 상태(완료 또는 실패) 오브젝트를 읽어 **UPDATE** 배치 처리 요청
- 오브젝트의 상태 변경 체크는 **TrackingMessage** 오브젝트에 상태 필드를 통해서 체크한다.

1.트레킹 메시지 적재 프로세스 설계

1.6.전체 프로세스

2) pitcher-catcher 구성(1/2)

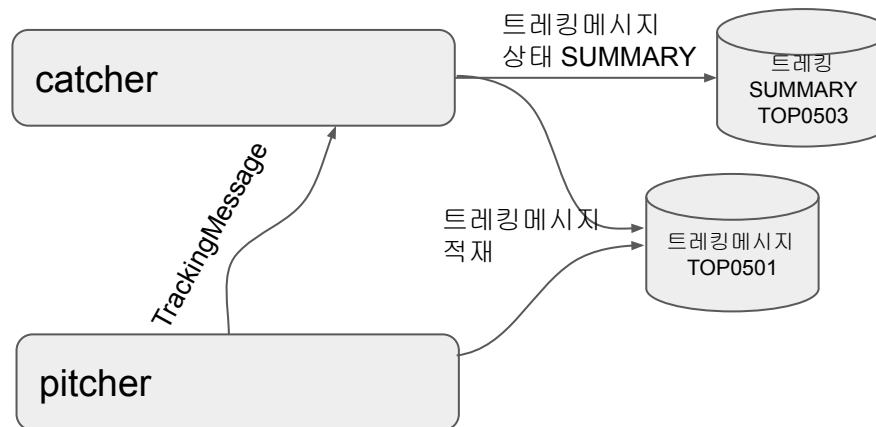
트레킹 시스템을 2곳 이상에서 운영할 경우 pitcher-catcher 로 구성가능하다. pitcher는 catcher 에게 TrackingMessage 를 던지고 catcher는 받은 메시지의 상태를 반영한다.



1.트레킹 메시지 적재 프로세스 설계

1.6.전체 프로세스

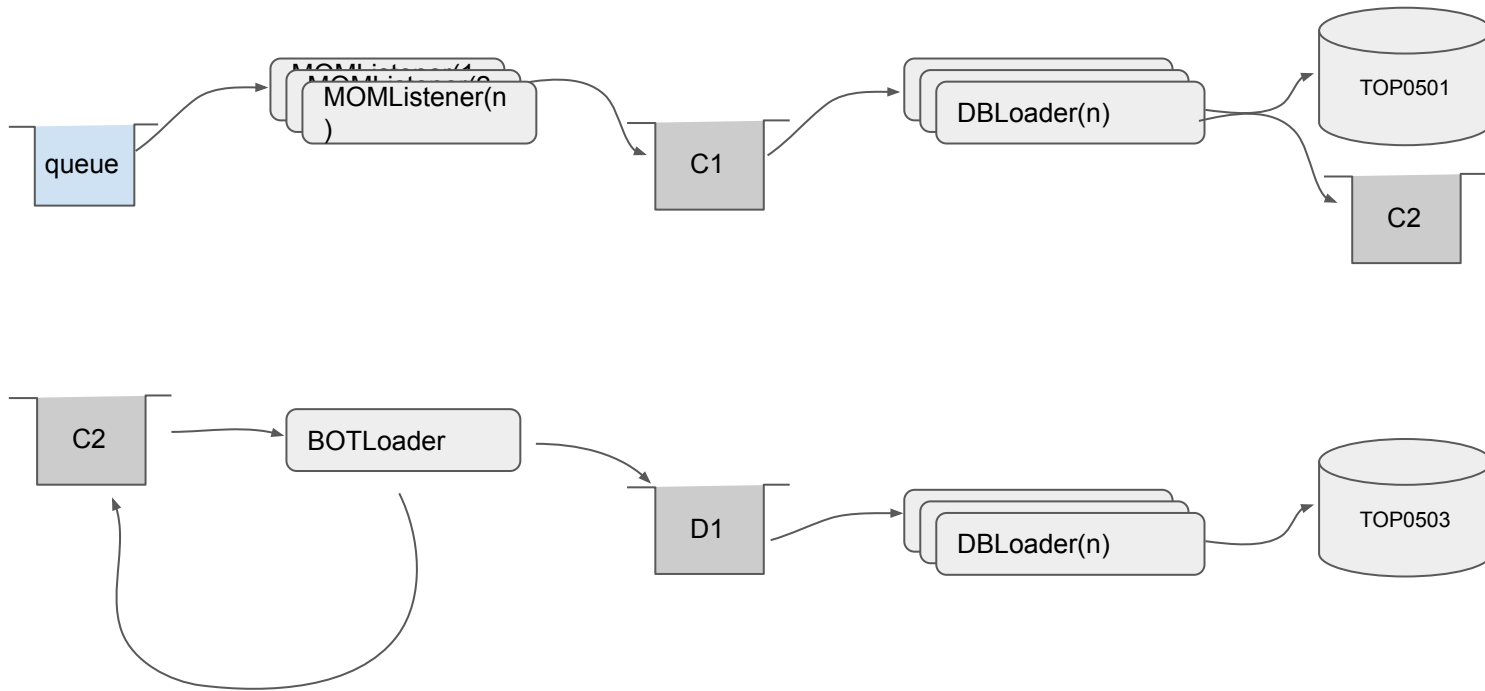
2) pitcher-catcher 구성(2/2)



트레킹 메시지 상태 **summary**는 **catcher** 시스템이 처리하도록 메시지를 전달한다.

1.트레킹 메시지 적재 프로세스 설계

1.6.전체 프로세스

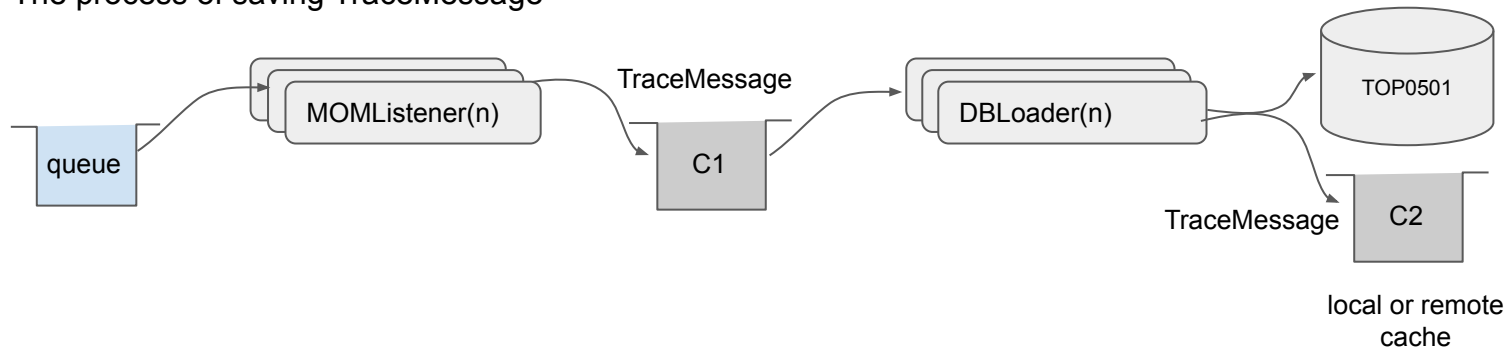


1.트레킹 메시지 적재 프로세스 설계

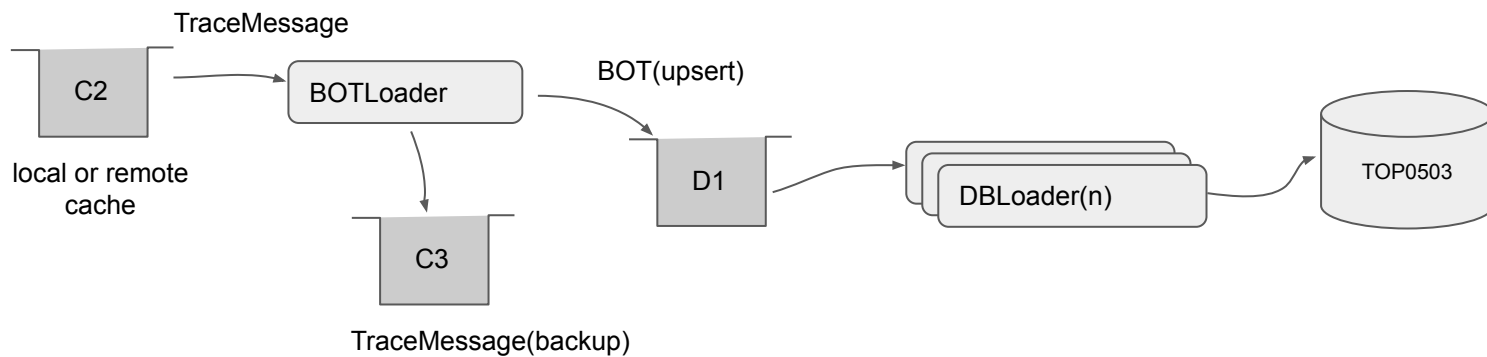
1.6.전체 프로세스

이걸로 진행?

The process of saving TraceMessage



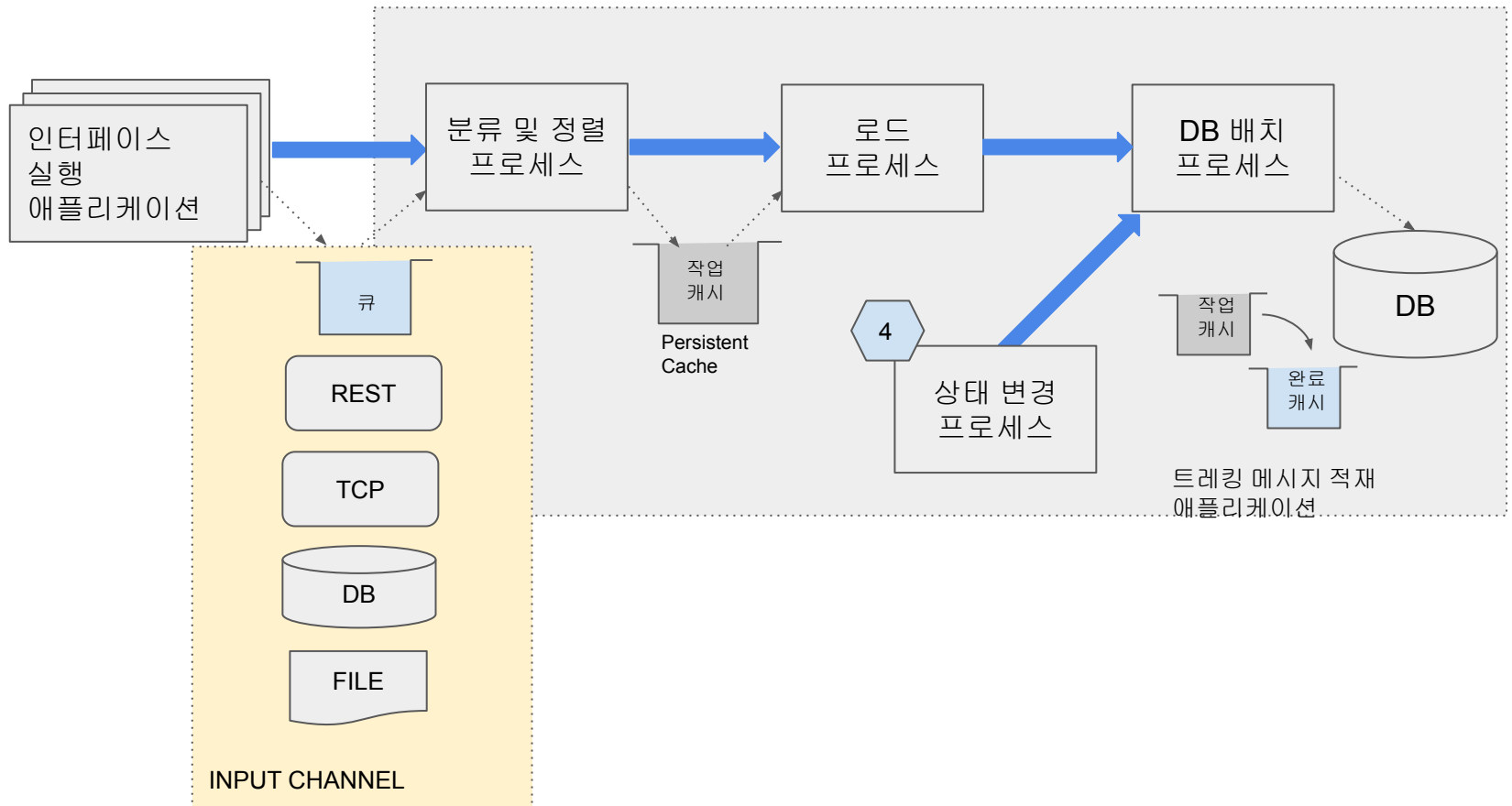
The process of saving BOT(Bill of Trace)



1.트레킹 메시지 적재 프로세스 설계

1.7.다채널 INPUT

트레킹 메시지 입력 채널을 큐 외의 REST, FILE, TCP, DB 등 다양한 채널을 지원할 수 있도록 한다.



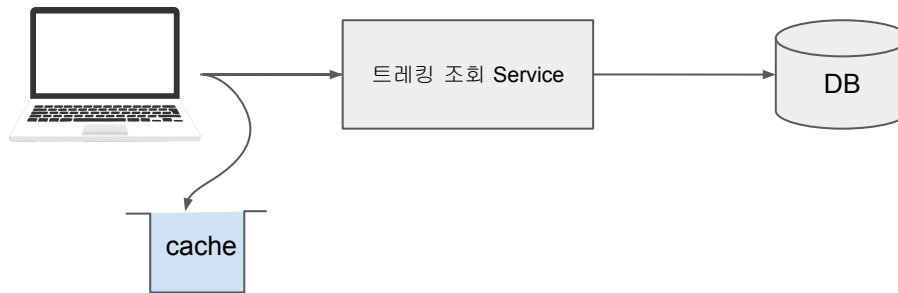
2.조회 프로세스 설계

2.1.caching 프로세스

1) 클라이언트 caching 방식

최초 서비스를 통해 데이터 조회 후 브라우저 **cache** 에 저장해 둔다.

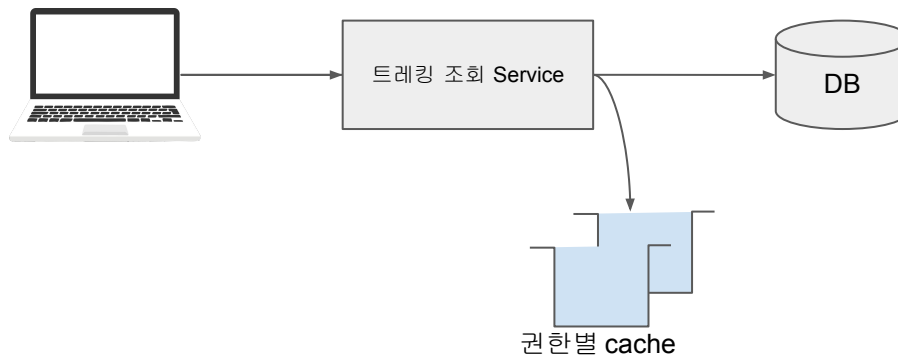
두번 째 조회 시 데이터 변경이 없는 과거 데이터 영역은 **cache** 에서 읽어들이고 최근 영역(예: 최근 1시간)만 서비스를 통해 조회 후 **cache** 영역 조회 결과와 더한다.



🧐 think about

- ❑ caching 프로세스 처리를 위한 F/W이 있는 지 조사해 본다.
- ❑ 트레킹 조회 **Service** 는 성능 개선의 관점에서 접근한다. (최소 JOIN)

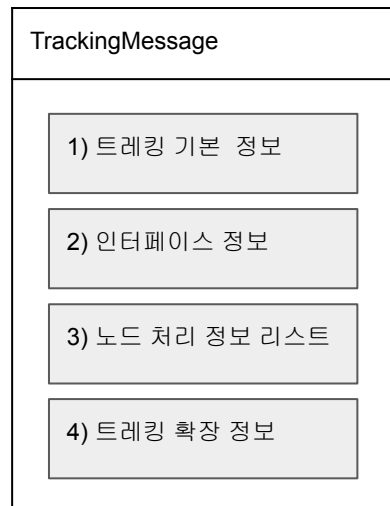
2) 서버 caching 방식은 권한별 조회 시 처리 복잡도가 높아지므로 고려하지 않기로 함.



3.트레킹 메시지 구조 설계

3.1.메시지 구조

TrackingMessage는 4개의 부분으로 구성된다.



1) 트레킹 기본 정보

- ☐ 트레킹 ID
- ☐ 트레킹 시간
- ☒ 상태
- ☐ 레코드 처리 건수
- ☐ 데이터 처리량
- ☐ 데이터 압축 구분
- ☐ 처리소요시간
- ☐ 처리할 노드 수
- ☐ 처리한 노드 수
- ☐ 에러 노드 수
- ☐ 에러코드(대표)
- ☐ 에러메시지(대표)
- ☐ 수신 노드 수


2) 인터페이스 정보

- ☐ 업무
- ☐ 인터페이스명
- ☐ 인터페이스ID
- ☐ 연계방식
- ☐ 데이터처리방
- ☐ 향
- ☐ 데이터처리방
- ☐ 식
- ☐ 처리방식
- ☐ 송신시스템
- ☐ 송신리소스
- ☐ 수신시스템
- ☐ 수신리소스
- ☐ 연계시스템

3) 노드 처리 정보

- ☐ 호스트 ID
- ☐ 프로세스 ID
- ☐ 프로세스 시작 시간
- ☐ 프로세스 종료 시간
- ☐ 노드 유형(송수신)
- ☐ 노드 IP
- ☐ 처리 상태
- ☐ 오류 코드
- ☐ 오류 메시지
- ☐ 레코드 처리 건수
- ☐ 데이터 처리량
- ☐ 데이터 압축 구분

4) 트레킹 확장 정보

-  향후 확장을 위한 정보



 think about

트레킹 메시지 상태 판단

- TDC : 처리할 노드 수
- FNC : 처리한 노드 수
- ERC : 에러 노드 수

- ❑ TDC = FNC and ERC = 0 이면 → “완료(성공)” : FS
- ❑ TDC = FNC and ERC > 0 이면 → “완료(실패)” : FF
- ❑ TDC > FNC and ERC > 0 이면 → “처리중(실패)” : IF
- ❑ TDC > FNC and ERC = 0 이면 → “처리중” : IN



 think about

- ❑ “처리중(실패)”를 AS-IS 처럼 “처리중”으로 할지 고민.
 - ❑ “처리중(실패)”는 좀 더 상세한 표현이 가능하다는 장점이 있음.
- 1:N 인터페이스의 경우 더 처리할 것이 있음을 의미

3.트레킹 메시지 구조 설계

3.2. TrackingMessage 정의

트레킹 기본정보						
SEQ	필드ID	필드명	유형	도메인/포맷	기본값	설명
1	trackingId	트레킹 ID	string(50)			유일값
2	trackingTime	트레킹 시간	string(20)	YYYYMMDDhh24mi ssSSS		인터페이스 트레킹 시작 시간
3	status	상태	string(4)	FS :완료(성공) FF : 완료(실패) IF : 처리중 (실패) IN : 처리중	1	인터페이스 처리 상태
4	recordCnt	레코드처리건수	number(12)		0	인터페이스 전체 처리 레코드 건수
5	dataSize	데이터처리량	number(12)		0	인터페이스 전체 처리 데이터 사이즈
6	compress	데이터압축구분	string(1)	0 : 비압축, 1 : 압축	0	처리 데이터 압축 여부
7	processingTime	처리소요시간	string(20)		0	인터페이스 처리 소요 시간(milli second)
8	todoNodeCnt	처리할노드수	number(4)		1	완료해야할 노드 개수
9	finNodeCnt	처리한노드수	number(4)		0	처리완료 노드 개수
10	errorNodeCnt	에러노드수	number(4)		0	에러 발생 노드 개수
11	errorCode	에러코드 (대표)	string(20)			에러코드 정의서 작성
12	errorMessage	에러메시지 (대표)	string			에러메시지 정의서 작성

3.트레킹 메시지 구조 설계

3.2. TrackingMessage 정의

인터페이스 정보						
SEQ	필드 ID	필드명	유형	도메인/포맷	기본값	설명
1	businessName	업무명	string			
2	intergrationId	인터페이스 ID	string			
3	interfaceName	인터페이스명	string			
4	channelName	연계방식	string			
5	dataPrDir	데이터처리방향	string			
6	dataPrMethod	데이터처리방식	string			
7	appPrMethod	처리방식	string			
8	sendSystemName	송신시스템명 [CD]	string			
9	sendResource	송신시스템리소스	string			
10	recvSystemName	수신시스템명 [CD]	string			
11	recvResource	수신시스템리소스	string			

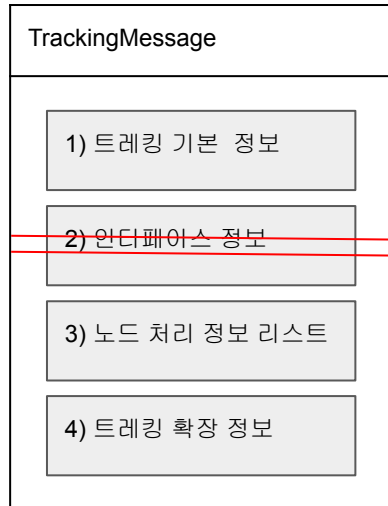
3.트레킹 메시지 구조 설계

3.2. TrackingMessage 정의

노드 처리 정보						
SEQ	필드ID	필드명	유형	도메인/포맷	기본값	설명
1	hostId	호스트 ID	string			프로세스가 실행된 노드의 호스트ID
2	processId	프로세스 ID	string			실행된 프로세스ID 또는 프로세스 실행시 임의로 부여한 값
3	startDate	프로세스시작시간	string(20)	YYYYMMDDhh24missSSS		프로세스 시작 시간
4	endDate	프로세스종료시간	string(20)	YYYYMMDDhh24missSSS		프로세스 종료 시간
5	type	노드유형	string(2)	0:송신, 1:허브, 2:수신	0	프로세스가 실행된 노드의 유형 (송신, 허브, 수신)
6	ip	노드 IP	string			서버 IP
7	status	처리상태	string(2)	0 : finished, 1 : processing, 9 : error	1	프로세스 처리 상태
8	errorCode	오류코드	string(20)			에러코드 정의서 작성
9	errorMessage	오류메시지	string			에러메시지 정의서 작성
10	recordCnt	레코드처리건수	number(4)		0	
11	dataSize	데이터처리량	number(4)		0	
12	compress	데이터압축구분	string(1)	0 : 비압축, 1 : 압축	0	

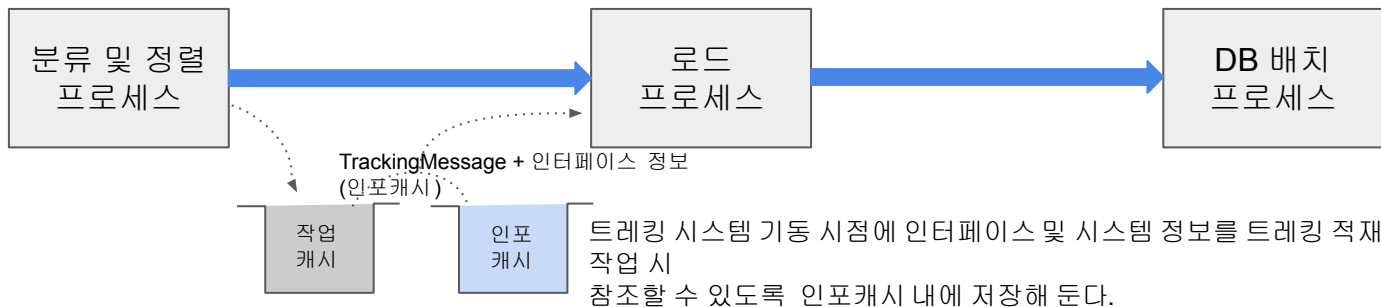
3.트레킹 메시지 구조 설계

3.3. TrackingMessage 설계 변경



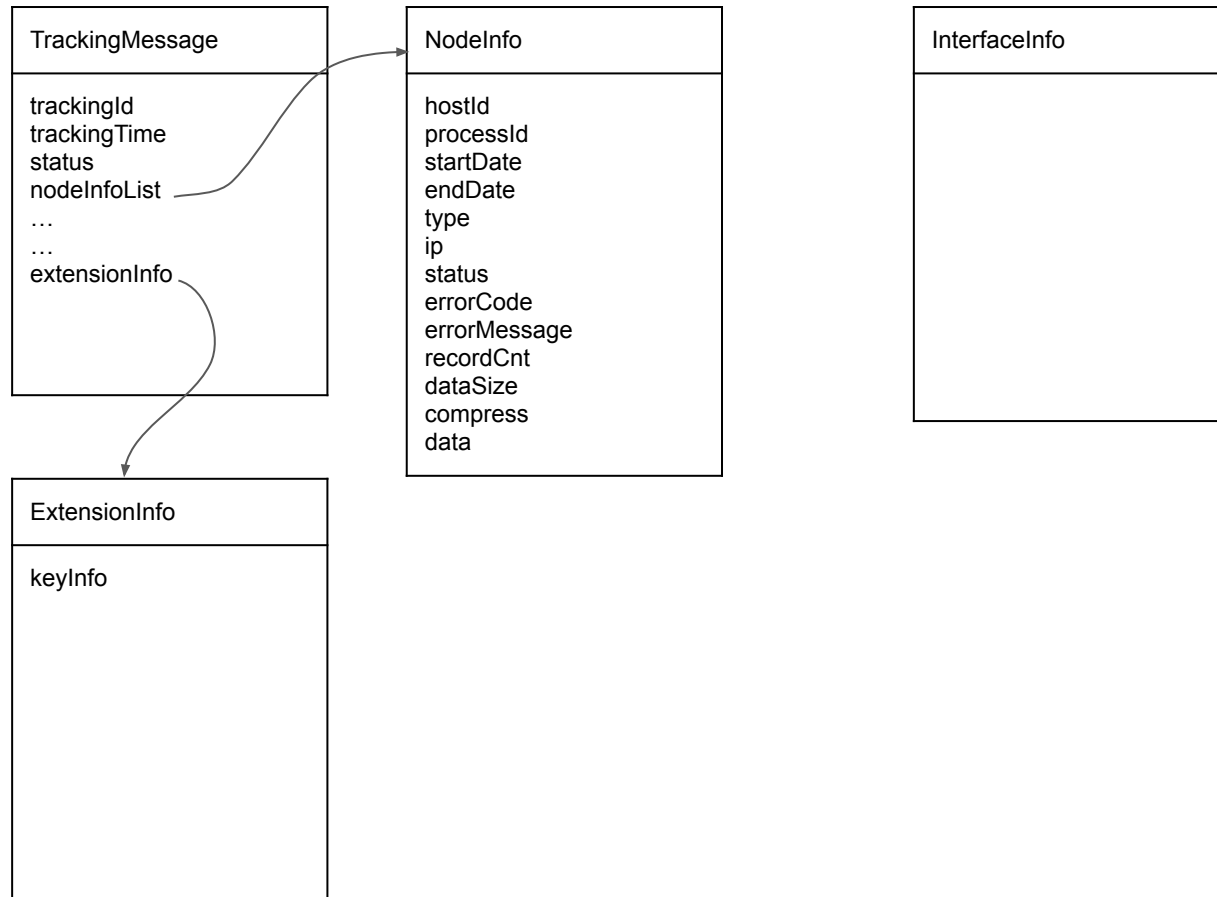
🧐 think about

- ❑ 인터페이스 정보는 제외한다.
(캐시 저장 용량을 고려함)
- ❑ 인터페이스 정보는 DB insert
시점에 인터페이스 캐싱정보에서
참고한다.



3.트레킹 메시지 구조 설계

3.4. TrackingMessage 클래스



4.테이블 설계

4.1.AS-IS 테이블 검토

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
디테일로그아이디	NUMBER	N N	DETAILLOG_ID
인터페이스시작시간	VARCHAR2(20)	N N	MSG_DATETIME
예리유형	VARCHAR2(16)	NULL	ERROR_TYPE
예리코드	VARCHAR2(10)	NULL	ERROR_CODE
응답유형	VARCHAR2(16)	NULL	RESP_TYPE
응답코드	VARCHAR2(10)	NULL	RESP_CODE
MQ예리코드	NUMBER(4)	NULL	REASON_CODE
예리메시지	VARCHAR2(2048)	NULL	ERROR_MSG
예리큐메시지아이디	VARCHAR2(48)	NULL	ERRORQ_MSG_ID
예리큐명	VARCHAR2(48)	NULL	ERRORQ_NM
타겟큐명	VARCHAR2(48)	NULL	TARGETQ_NM

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
디테일로그아이디	NUMBER	N N	DETAILLOG_ID
인터페이스시작시간	VARCHAR2(20)	N N	MSG_DATETIME
디렉토리명	VARCHAR2(256)	NULL	DIRECTORY_NM
파일명	VARCHAR2(256)	NULL	FILE_NM
파일사이즈	NUMBER(16)	NULL	FILE_SIZE
외부프로그램	VARCHAR2(20)	NULL	EXT_PROGRAM
리갈	VARCHAR2(32)	NULL	LEGAL

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
디테일로그아이디	NUMBER	N N	DETAILLOG_ID
인터페이스시작시간	VARCHAR2(20)	N N	MSG_DATETIME
DETAIL01	VARCHAR2(1024)	NULL	DETAIL01
DETAIL02	VARCHAR2(1024)	NULL	DETAIL02
DETAIL03	VARCHAR2(1024)	NULL	DETAIL03
DETAIL04	VARCHAR2(1024)	NULL	DETAIL04
DETAIL05	VARCHAR2(1024)	NULL	DETAIL05
DETAIL06	VARCHAR2(1024)	NULL	DETAIL06
DETAIL07	VARCHAR2(1024)	NULL	DETAIL07
DETAIL08	VARCHAR2(1024)	NULL	DETAIL08
DETAIL09	VARCHAR2(1024)	NULL	DETAIL09
DETAIL10	VARCHAR2(1024)	NULL	DETAIL10

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
디테일로그아이디	NUMBER	N N	DETAILLOG_ID
프로세스아이디	VARCHAR2(50)	NULL	PR_PROCESS_ID
호스트아이디	VARCHAR2(36)	NULL	PR_HOST_ID

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
디테일로그아이디	NUMBER	N N	DETAILLOG_ID
인터페이스시작시간	VARCHAR2(20)	N N	MSG_DATETIME
PR프로세스모드	VARCHAR2(16)	NULL	PR_PROCESS_MODE
PR프로세스타입	VARCHAR2(16)	NULL	PR_PROCESS_TYPE
PR프로세스일시	DATE	NULL	PR_DT
처리상태	CHAR(2)	NULL	MSG_STATUS
프로세스실행횟수	NUMBER(8)	NULL	PR_HOP_CNT
큐메지저명	VARCHAR2(48)	NULL	MQMD_QMGR
응답큐메지저명	VARCHAR2(48)	NULL	REPLY_QMGR
응답큐	VARCHAR2(48)	NULL	REPLY_QUEUE
레코드건수	NUMBER(8)	NULL	RECORD_CNT
레코드사이즈	NUMBER(16)	NULL	RECORD_SIZE
데이터사이즈	NUMBER(16)	NULL	DATA_SIZE
압축여부	CHAR(1)	NULL	COMPRESS_YN
압축방법	VARCHAR2(20)	NULL	COMPRESS_METHOD
압축모드	VARCHAR2(8)	NULL	COMPRESS_MODE
압축사이즈	NUMBER(16)	NULL	COMPRESS_SIZE
변환여부	CHAR(1)	NULL	CONV_YN
변환모드	VARCHAR2(8)	NULL	CONV_MODE
변환사이즈	NUMBER(16)	NULL	CONV_SIZE
링크코드	VARCHAR2(5)	NULL	LINK_CODE
어댑터코드	VARCHAR2(10)	NULL	ADAPTER_CODE
레코드S건수	NUMBER	NULL	RECORD_S_CNT
레코드E건수	NUMBER	NULL	RECORD_E_CNT
처리시간	VARCHAR2(20)	NULL	PROCESS_TIME

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
디테일로그아이디	NUMBER	N N	DETAILLOG_ID
인터페이스시작시간	VARCHAR2(20)	N N	MSG_DATETIME
데이터	BLOB	NULL	DATA

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
호스트아이디	VARCHAR2(36)	NULL	HOST_ID
그룹아이디	VARCHAR2(48)	NULL	GROUP_ID
인터페이스아이디	VARCHAR2(48)	NULL	INTF_ID
어댑터아이디	VARCHAR2(10)	NULL	ADAPTER_CODE

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
인터페이스시작시간	VARCHAR2(20)	N N	MSG_DATETIME
상태	CHAR(2)	NULL	MSG_STATUS
허브실행횟수	NUMBER(8)	NULL	HUB_CNT
스포크실행횟수	NUMBER(8)	NULL	SPOKE_CNT
수신허브실행횟수	NUMBER(8)	NULL	RECV_HUB_CNT
수신스포크실행횟수	NUMBER(8)	NULL	RECV_SPOKE_CNT
예리유형	VARCHAR2(16)	NULL	ERROR_TYPE
예리발생개수	NUMBER(16)	NULL	ERROR_CNT
예리코드	NUMBER(4)	NULL	REASON_CODE
메시지TTL	DATE	NULL	MSG_TTL
메시지ALERT	CHAR(1)	NULL	MSG_ALERT
레코드처리건수	NUMBER(8)	NULL	RECORD_CNT
어댑터CODE	VARCHAR2(10)	NULL	ADAPTER_CODE
레코드사이즈	NUMBER(16)	NULL	RECORD_SIZE
데이터사이즈	NUMBER(16)	NULL	DATA_SIZE
압축여부	CHAR(1)	NULL	COMPRESS_YN
압축사이즈	NUMBER(16)	NULL	COMPRESS_SIZE
소스디렉토리명	VARCHAR2(128)	NULL	SOURCE_DIR_NM
소스파일명	VARCHAR2(64)	NULL	SOURCE_FILE_NM
타겟디렉토리명	VARCHAR2(128)	NULL	TARGET_DIR_NM
타겟파일명	VARCHAR2(64)	NULL	TARGET_FILE_NM
레코드S건수	NUMBER	NULL	RECORD_S_CNT
레코드E건수	NUMBER	NULL	RECORD_E_CNT
처리시간	VARCHAR2(48)	NULL	P_TIME
수신호스트ID	VARCHAR2(4000)	NULL	RECV_HOST_ID

논리 이름*	데이터 타입	널 허용	물리 이름*
마스터로그아이디	NUMBER	N N	MASTERLOG_ID
인터페이스시작시간	VARCHAR2(20)	N N	MSG_DATETIME
MASTER01	VARCHAR2(1024)	NULL	MASTER01
MASTER02	VARCHAR2(1024)	NULL	MASTER02

4.테이블 설계

4.2.TO-BE 테이블 설계(1/2)

TOP0501 트래킹메시지				
물리 이름*	논리 이름*	도메인	데이터 타입	널 허용
KEY INTERFACE_ID	인터페이스ID	엔티티아이디	VARCHAR(50)	N N
KEY TRACKING_DATE	트래킹일시	일시20	VARCHAR(20)	N N
KEY NODE_ID	노드ID	사용자일력50	VARCHAR(50)	N N
STATUS	상태	코드값2	VARCHAR(2)	N N
NODE_TYPE	노드유형	사용자일력50	VARCHAR(50)	N N
START_DATE	노드시작시간	일시20	VARCHAR(20)	N N
END_DATE	노드종료시간	일시20	VARCHAR(20)	N N
SEQ	순번	순서	INTEGER	N N
HOST_ID	호스트ID	사용자일력50	VARCHAR(50)	N N
PROCESS_ID	프로세스ID	사용자일력50	VARCHAR(50)	N N
IP	노드IP	IP	VARCHAR(15)	NULL
OS	OS	사용자일력50	VARCHAR(50)	NULL
APP_NM	애플리케이션	사용자일력50	VARCHAR(50)	NULL
ERROR_CD	에러코드	코드값	VARCHAR(5)	NULL
ERROR_MSG	에러메시지	사용자일력1024	VARCHAR(1024)	NULL
RECORD_CNT	레코드처리건수	숫자	INTEGER	NULL
DATA_AMT	데이터처리량	숫자	INTEGER	NULL
COMPRESS	데이터압축구분	구분	VARCHAR(1)	NULL
DIRECTORY	디렉토리	사용자일력코드	VARCHAR(255)	NULL
FILE	파일명	사용자일력코드	VARCHAR(255)	NULL
FILE_SIZE	파일사이즈	숫자	INTEGER	NULL

TOP0503 트래킹요약정보				
물리 이름*	논리 이름*	도메인	데이터 타입	널 허용
KEY INTERFACE_ID	인터페이스ID	엔티티아이디	VARCHAR(50)	N N
KEY TRACKING_DATE	트래킹일시	일시20	VARCHAR(20)	N N
STATUS	상태	코드값2	VARCHAR(2)	N N
RECORD_CNT	레코드처리건수	숫자	INTEGER	NULL
DATA_AMT	데이터처리량	숫자	INTEGER	NULL
COMPRESS	데이터압축구분	구분	VARCHAR(1)	NULL
COST	처리소요시간	숫자	INTEGER	NULL
TDC	처리할노드수	숫자	INTEGER	NULL
FNC	처리한노드수	숫자	INTEGER	NULL
ERC	에러노드수	숫자	INTEGER	NULL
ERROR_CD	에러코드	코드값	VARCHAR(5)	NULL
ERROR_MSG	에러메시지	사용자일력1024	VARCHAR(1024)	NULL
BUSINESS_NM	업무명	사용자일력50	VARCHAR(50)	NULL
INTERFACE_NM	인터페이스명	사용자일력50	VARCHAR(50)	NULL
INTEGRATION_ID	인티그레이션ID	엔티티아이디	VARCHAR(50)	NULL
CHANNEL_NM	연계방식명	사용자일력50	VARCHAR(50)	NULL
DATA_PR_DIR_NM	데이터처리방향명	사용자일력50	VARCHAR(50)	NULL
DATA_PR_METHOD_NM	데이터처리방식명	사용자일력50	VARCHAR(50)	NULL
APP_PR_METHOD_NM	처리방식명	사용자일력50	VARCHAR(50)	NULL
SND_SYSTEM_NM	송신시스템명	사용자일력50	VARCHAR(50)	NULL
SND_RES_NM	송신리소스명	사용자일력50	VARCHAR(50)	NULL
RCV_SYSTEM_NM	수신시스템명	사용자일력50	VARCHAR(50)	NULL
RCV_RES_NM	수신리소스명	사용자일력50	VARCHAR(50)	NULL

TOP0504 트래킹추가정보				
물리 이름*	논리 이름*	도메인	데이터 타입	널 허용
KEY INTERFACE_ID	인터페이스ID	엔티티아이디	VARCHAR(50)	N N
KEY TRACKING_DATE	트래킹일시	일시20	VARCHAR(20)	N N

🤔 think about

- ❑ 저장 속도 개선을 위한 단일 테이블 고려
- ❑ 조회 속도 개선을 위한 최소 JOIN
- ❑ 로그성 테이블이므로 중복 허용 이름 값 길이 제한 VARCHAR(50)

TOP0502 트래킹데이터				
물리 이름*	논리 이름*	도메인	데이터 타입	널 허용
KEY INTERFACE_ID	인터페이스ID	엔티티아이디	VARCHAR(50)	N N
KEY TRACKING_DATE	트래킹일시	일시20	VARCHAR(20)	N N
KEY NODE_ID	노드ID	사용자일력50	VARCHAR(50)	N N
DATA	데이터	파일객체	BLOB	NULL

4.테이블 설계

4.2.TO-BE 테이블 설계(2/2)

트레킹 시스템의 관리를 위한 환경 설정값 저장

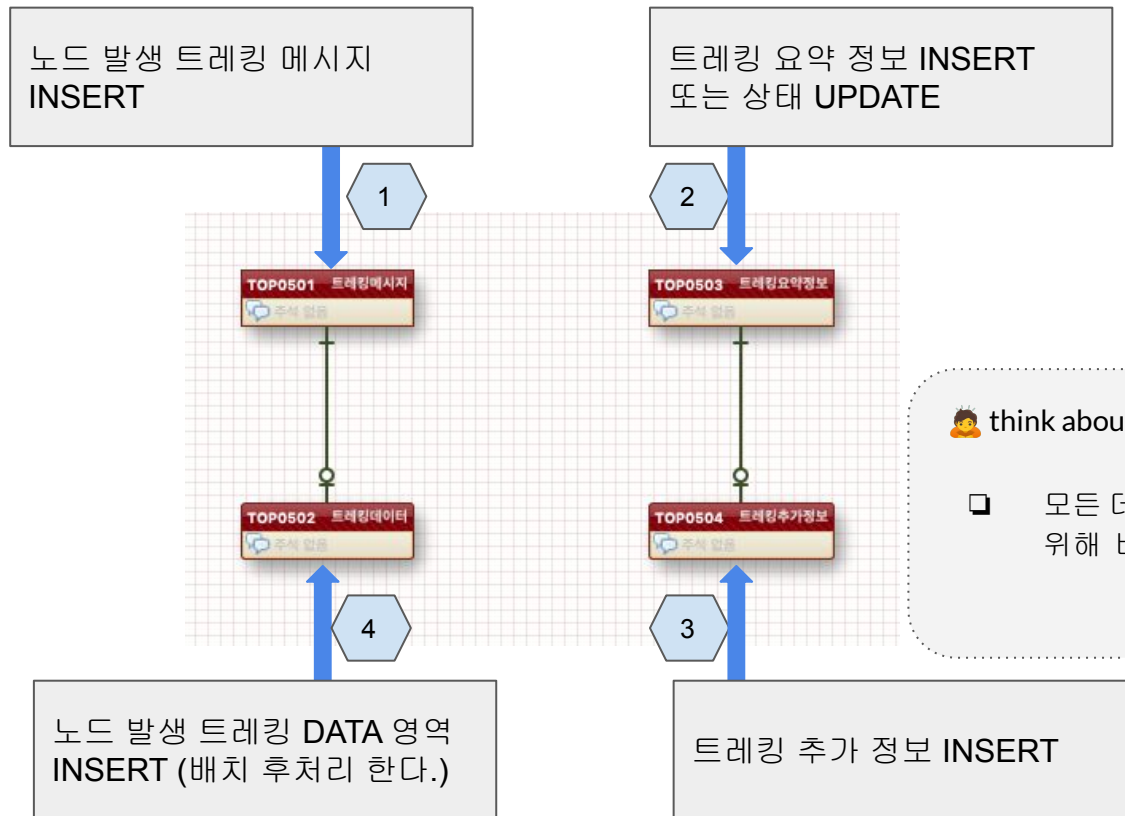
TOP0505		트레킹환경설정		
물리 이름*	논리 이름*	도메인	데이터 타입	널 허용
🔑 PACKAGE	패키지	N/A	VARCHAR(255)	N·N
🔑 ATTRIBUTE_ID	속성ID	엔터티아이디	VARCHAR(50)	N·N
🔑 IDX	인덱스	순서	INTEGER	N·N
● ATTRIBUTE_NM	속성명	엔터티명	VARCHAR(255)	NULL
● ATTRIBUTE_VALUE	속성값	속성값	VARCHAR(100)	NULL
● COMMENTS	설명	설명	VARCHAR(1000)	NULL
● DEL_YN	삭제구분	삭제구분	VARCHAR(1)	N·N
● REG_DATE	등록일	등록일	VARCHAR(17)	N·N
● REG_USER	등록자	등록자	VARCHAR(100)	N·N
● MOD_DATE	최종수정일	수정일	VARCHAR(17)	NULL
● MOD_USER	최종수정자	수정자	VARCHAR(100)	NULL

트레킹 시스템 처리 상태 데이터 기록

TOP0506		트레킹처리모니터		
물리 이름*	논리 이름*	도메인	데이터 타입	널 허용
🔑 MON_DATE	모니터링일시	일시	VARCHAR(17)	N·N
● STATUS	상태	코드값2	VARCHAR(2)	N·N
● ERROR_MSG	예러메시지	사용자입력1024	VARCHAR(1024)	NULL
● TPS	TPS	숫자	INTEGER	NULL
● CPU	CPU사용량	숫자	INTEGER	NULL
● MEM	메모리사용량	숫자	INTEGER	NULL
● DISK	디스크사용량	숫자	INTEGER	NULL
● CACHE	캐시건수	숫자	INTEGER	NULL

4.테이블 설계

4.3.데이터 처리 프로세스



5.SW 설계

5.1.개발 기능 리스트

5.1.1.트레킹 적재	5.1.2.트레킹 조회	5.1.3.트레킹 관리
<ul style="list-style-type: none"> ❑ 트레킹 INPUT 핸들링(큐, REST, API, 파일 등) ❑ 트레킹 메시지 파싱 & 오브젝트 변환 ❑ 오브젝트 캐싱 ❑ 기본정보 프리로딩 및 캐싱 (설정 / 인터페이스 등) ❑ DB 배치 처리 ❑ 스레드 관리 ❑ 트랜잭션 관리(메시지 보장) ❑ 예외처리(비정상 종료 /OOM/디스크풀 등) ❑ 데이터 암호화 ❑ 인터페이스별 상태처리 옵션 (예 : 송신 트레킹만 발생해도 완료 처리) ❑ 트레킹 노드 순서 부여 옵션 제공 	<ul style="list-style-type: none"> ❑ 트레킹 조회 및 상세 조회 ❑ 집계 프로그램 ❑ 대시보드(로더 상태) ❑ 트레킹 관리 ❑ 인터페이스 상세 조회→최근 발생량 	<ul style="list-style-type: none"> ❑ 관리 화면 (설정/명령/캐시뷰/리소스/통계) ❑ 명령 처리 (시작/종료/재시작/변경) ❑ 리소스 사용량 체크 (디스크/메모리/CPU) ❑ 통계 기록(처리속도/처리량) ❑ 트레킹 데이터 정리(백업/삭제) ❑ 프리로딩 데이터 및 캐싱 관리 ❑ 설정 관리 ❑ 오브젝트 캐싱 관리 ❑ 로그 ❑ 상태 전송(→ IIP)

5.SW 설계

5.1.개발 기능 리스트

5.1.1.트레킹 적재(1/2)

기능	내용
❑ 트레킹 INPUT 핸들링	<ul style="list-style-type: none"> 처리할 트레킹 메시지 입력 채널을 큐로만 한정하지 않고 다양한 채널을 제공하도록 한다. 1) 큐 2) REST 3) TCP 4) 파일 5) 디비 등 리스너 패턴을 적용하여 다양한 INPUT에 대해 하나의 표준화된 출력이 가능하도록 한다.
❑ 트레킹 메시지 파싱 & 오브젝트 변환	<ul style="list-style-type: none"> 입력 채널로 들어온 트레킹 메시지를 설정된 포맷으로 파싱하고 표준 오브젝트로 변환한다. 트레킹 메시지 포맷이 변경되어도 시스템 변경이 발생하지 않도록 맵핑기능이 있어야 한다.
❑ 오브젝트 캐싱	<ul style="list-style-type: none"> 표준 트레킹 오브젝트를 캐시에 저장하고 대량 배치처리 가능하도록 한다. 캐시는 저장된 데이터를 지속할 수 있어야 한다.(VM 셧다운 시)
❑ 기본정보 프리로딩 및 캐싱	<ul style="list-style-type: none"> 트레킹 데이터의 신속한 처리를 위해 기준정보를 프로그램 로딩시 캐시에 저장하여 사용한다. 캐시 대상 정보 : 인터페이스, 옵션처리 정보, 환경 설정 정보 정보 변경에 따른 리로딩 기능을 제공한다.
❑ DB 배치 처리	<ul style="list-style-type: none"> 트레킹 데이터의 신속한 DB 처리를 위해 INSERT, UPDATE 문의 배치 처리를 수행한다. 모든 DB 처리를 한 곳에서 처리하도록 한다. 배치처리는 병렬 스레드로 실행 가능하도록 한다.
❑ 스레드 관리	<ul style="list-style-type: none"> INPUT 채널 리스닝, 트레킹메시지 로딩, DB 배치 등 스레드 기반 태스크를 지정 숫자 및 라이프 사이클을 설정하고 모니터링 가능하도록 개발한다.

5.SW 설계

5.1.개발 기능 리스트

5.1.1.트레킹 적재(2/2)

기능	내용
❑ 트랜잭션 관리	<ul style="list-style-type: none">입력 채널 → 캐시, 캐시 → DB 배치 처리에 대한 실행 및 취소에 대한 트랜잭션을 관리한다.
❑ 예외처리	<ul style="list-style-type: none">가상머신의 OOM(Out Of Memory) 및 외부적 요인에 의한 셧다운에 대한 복구 기능을 보장한다.
❑ 데이터 암호화	<ul style="list-style-type: none">트레킹 메시지 기록 시 데이터부를 저장 옵션에 대해 암호화 기능을 구현한다.기본 암호화 로직 반영 및 사용자 정의 암호화 적용이 가능하도록 한다.
❑ 인터페이스별 상태처리 옵션	<ul style="list-style-type: none">송신 트레킹만 존재할 수 있는 경우 완료 상태로 처리될 수 있도록 옵션을 제공한다.
❑ 트레킹 노드 순서 부여 옵션 제공	<ul style="list-style-type: none">미리 등록된 기준 값에 의해 노드 처리 순서가 정렬되어 DB에 입력 처리 될수 있도록 맵핑 값을 설정할 수 있도록 한다.

5.SW 설계

5.1.개발 기능 리스트

5.1.2.트레킹 조회

기능	내용
❑ 트레킹 조회 및 상세 조회	<ul style="list-style-type: none">• TO-BE 트레킹 타겟으로 하는 서비스 및 조회, 상세조회 화면을 제공한다.(서비스 & 프론트)
❑ 집계	<ul style="list-style-type: none">• TO-BE 트레킹 테이블을 소스로 하는 집계를 수행한다.(배치 프로그램 개발)
❑ 대시보드	<ul style="list-style-type: none">• TO-BE 트레킹 타겟으로 하는 대시보드 서비스를 제공한다.• 트레킹 알람 기능도 제공한다.
❑ 트레킹 관리	<ul style="list-style-type: none">• 트레킹 애플리케이션을 관리하는 프론트 화면을 제공한다.(트레킹 시작/중지/변경 등)
❑ 인터페이스 상세 조회	<ul style="list-style-type: none">• 기존 인터페이스 상세 조회 화면에 최근 발생량 정보를 제공한다.

5.SW 설계

5.1.개발 기능 리스트

5.1.3.트레킹 관리(1/2)

기능	내용
❑ 관리 화면	<ul style="list-style-type: none">• 트레킹 Config 설정 화면을 제공한다.• 캐시상태 뷰 화면을 제공한다.• 리소스 뷰 화면을 제공한다.• 통계화면을 제공한다.
❑ 명령 처리	<ul style="list-style-type: none">• 시작 / 종료 / 설정 변경 명령을 수신하고 처리한다.• 명령의 수신은 로컬화면, 콘솔, 원격 IIP 서버에서 요청할 수 있도록 한다.
❑ 리소스 사용량 체크	<ul style="list-style-type: none">• 디스크 / 메모리 / CPU 등 사용량을 체크 기록 전송한다.
❑ 통계 기록	<ul style="list-style-type: none">• 트레킹 처리 속도 및 처리량을 기록 전송한다.
❑ 트레킹 데이터 정리	<ul style="list-style-type: none">• 데이터 백업 정책을 설정하고 정책에 따라 정리하는 기능을 제공한다.
❑ 프리로딩 데이터 및 캐시 관리	<ul style="list-style-type: none">• 인터페이스 정보 등 프리로딩 대상 데이터를 재반영할 수 있도록 한다.
❑ 설정 관리	<ul style="list-style-type: none">• 각종 설정값을 관리하고 재반영 할 수 있도록 한다.
❑ 오브젝트 캐시 관리	<ul style="list-style-type: none">• 오브젝트 캐시의 상태 모니터링 서비스를 제공한다.

5.SW 설계

5.1.개발 기능 리스트

5.1.3.트래킹 관리(2/2)

기능	내용
❑ 로그	<ul style="list-style-type: none">● 애플리케이션 로그를 관리한다.
❑ 상태 전송	<ul style="list-style-type: none">● IIP로 애플리케이션 상태를 전송한다.

5.SW 설계

5.2.아키텍처, F/W

5.2.1.아키텍처

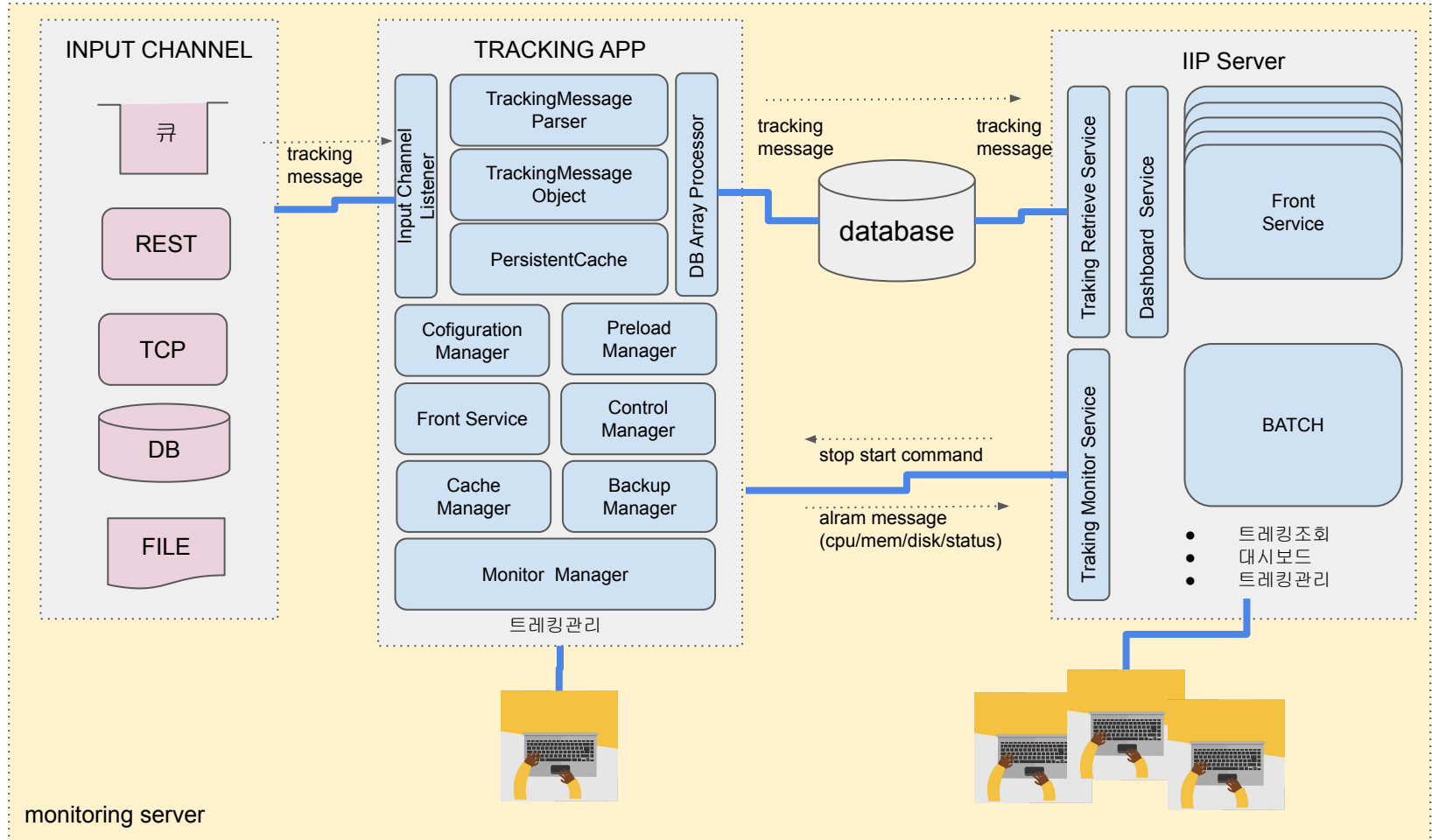
🤖 think about

❑ TRACKING APP 은 1개만 실행 가능

→ 트래킹의 최종 상태 값을 로컬 캐시에서 적재하므로 노드 데이터가 분산되면

상태 처리 불가능한 아키텍처 임.

→ 그래도 문제 없을까? (스레드 기반 아키텍처와 처리 성능 간의 관계)



5.SW 설계

5.2.아키텍처, F/W

5.2.2.F/W

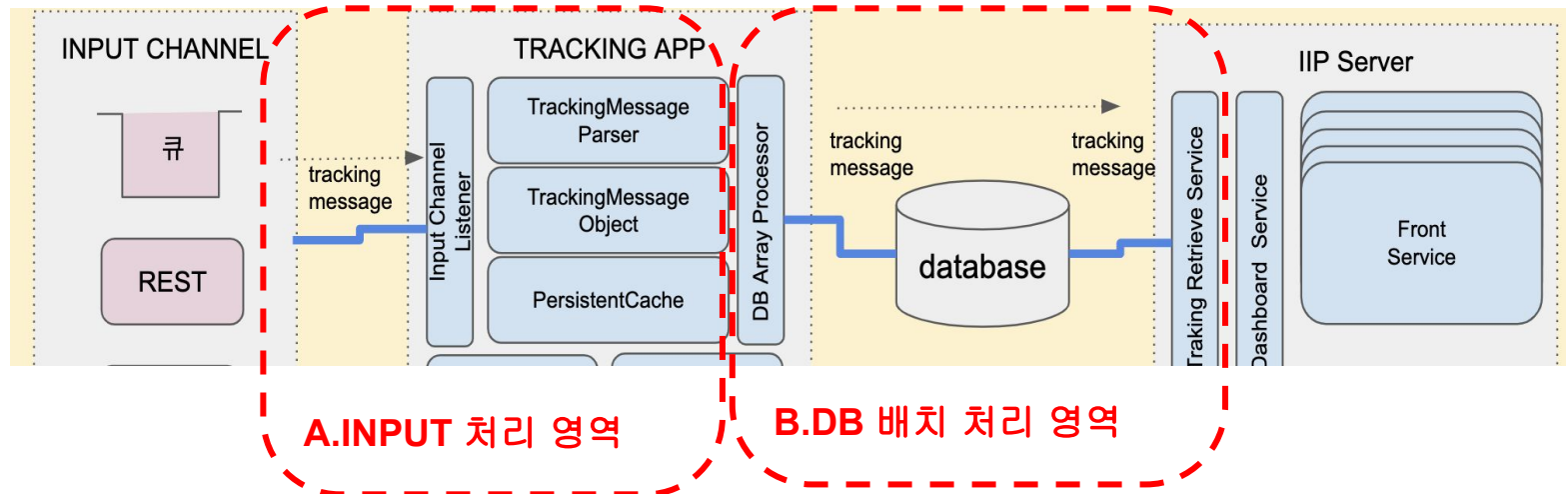
개발에 적용할 F/W 리스트

- ☐ springframework
- ☐ spring-boot
- ☐ Ehcache
- ☐ H2Database
- ☐ ibatis
- ☐ java1.8
- ☐ gradle
- ☐ github

5.SW 설계

5.3.대량 데이터 처리

5.3.1.데이터 처리 병목 구간



5.SW 설계

5.3.대량 데이터 처리

5.3.2.구간별 성능(초당 처리 건 수:TPS) 측정 테스트

테스트 환경 :

OS : Linux 3.10.0-514 x86_64

Memory : 24Gbytes

CPU : 8 core, Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30GHz

no	input msg	A 스레드 개수	B 스레드 개수	commit count	tps	cpu 사용량	메모리 사용량
1	1,000,000	1	1	100			
2	1,000,000	10	1	1000			
3	1,000,000	20	1	1000			
4	1,000,000	40	2	500			
5	1,000,000	100	10	100			
6							
7							
8							
9							
10							

5.SW 설계

5.3.대량 데이터 처리

5.3.3.처리 방식 및 튜닝

테스트 결과를 검토하여 설정 값 임계치를 찾고 평균 처리 속도를 구해 본다.

결과가 목표치에 미치지 못할 경우 아키텍처 및 테이블 구조를 재검토 하도록 한다.

트래킹 목표 처리량

- ❑ 일 트랜잭션 발생량 : 10,000,000 건 (10일에 1억 건 발생)
- ❑ 처리 시간 : 8 시간
- ❑ TPS 계산 :
 $10,000,000 \text{ 건} / (8 \text{ 시간} * 60 * 60) = \text{약 } 350 \text{ tps}$
- ❑ 목표 처리량 : 500 tps

5.SW 설계

5.4.기능 상세 설계

5.4.1.개발 항목(1/2)

분류	항목	설명
A.트래킹	트래킹 메시지 로딩 APP	트래킹 적재 애플리케이션 기능 개발
B.트래킹 매니저	설정 관리 매니저	트래킹 애플리케이션 설정과 관련한 정보를 관리 기능 개발
	프리로딩 매니저	인터페이스 정보 등 프리로딩 데이터를 관리 기능 개발
	컨트롤 매니저	애플리케이션 시작, 종료, 설정 변경 등을 관리 기능 개발
	채널 매니저	트래킹 INPUT 채널 관리 기능 개발
	캐시 매니저	Persistent Cache 를 관리 기능 개발
	백업 매니저	트래킹 로그 백업 정리 관리 기능 개발
	모니터 매니저	리소스 모니터링 및 알람 관리 기능 개발
C.트래킹 매니저 프론트	매니저 관리 서비스 & 컨트롤러	트래킹 매니저들 관리를 위한 서비스 및 컨트롤러 기능 개발
	매니저 관리 화면	트래킹 애플리케이션 관리를 위한 화면 개발
	매니저 관리 콘솔 모드	트래킹 애플리케이션 관리를 위한 명령 콘솔 개발
	트래킹 조회	1일치 로컬 DB 저장 영역에서 조회기능 제공.

5.SW 설계

5.4.기능 상세 설계

5.4.1.개발 항목(2/2)

분류	항목	설명
D.IIP 모니터링	트레킹 조회 및 상세조회 화면	모니터링 조회 화면 개발
	트레킹 조회 및 상세조회 서비스	모니터링 조회 서비스 기능 개발
	기타 트레킹 관련 AS-IS 화면 리뉴얼	AS-IS 화면 리뉴얼
E.IIP 대시보드	AS-IS 대시보드 화면 리뉴얼	AS-IS 대시보드 리뉴얼
F.IIP 통계	트레킹 통계 집계	트레킹 집계 배치 기능 개발
G.IIP 트레킹 관리	트레킹 매니저 원격 관리 화면 및 서비스	트레킹 애플리케이션 관리를 위한 화면 개발
	트레킹 매니저 상태 모니터링 화면 및 서비스	트레킹 애플리케이션의 상태를 디스플레이 하기 위한 화면 개발

5.SW 설계

5.4.기능 상세 설계

5.4.2.항목별 설계

C.트레킹매니저 프론트 - 매니저 관리 - 메인 화면



5.SW 설계

5.4.기능 상세 설계

5.4.2.항목별 설계

C.트레킹매니저 프론트 - 매니저 관리 - 로그 화면

트레킹 시스템 운영 로그 뷰를 제공한다. (옵션 : 별도 파일로 저장하거나 메일 발송 기능 제공)



5.SW 설계

5.4.기능 상세 설계

5.4.2.항목별 설계

C.트레킹매니저 프론트 - 매니저 관리 - 모니터 화면

트레킹 시스템의 **CPU MEMORY DISK** 및 처리 상태를 모니터링할 수 있도록 기능을 제공한다.



5.SW 설계

5.4.기능 상세 설계

5.4.2.항목별 설계

C.트레킹매니저 프론트 - 매니저 관리 - 설정 화면

트레킹 관리자 > 설정하기

트레킹 환경 :

저장

배치

버킷
카운트

1,000

메시
스레드

1

배치 주기

1

캐시

캐시 개수

3

최대값

500

지연처리

1

채널

INPUT

MQ

백업 :

저장

로그 | 모니터 | 설정

🧐 think about

- ❑ 설정 대상 카테고리
 - 1) 트레킹 환경
 - 2) 백업
 - 3) 채널
 - 4) 로그
 - 5) 캐시
 - 6) 보안

5.SW 설계

5.4.기능 상세 설계

5.4.2.항목별 설계

D.IIP 모니터링 - 트래킹 조회

<>

→

Q

≡

모니터링 > 트래킹 조회

상태	인터페이스 명	인터페이스 ID	송신 시스템	*송신 리소스	수신 시스템	*수신 리소스	시작 시간	*소요 시간	데이터 사이즈

think about

☐ 조회 조건 및 결과 구성 필드는
1) 트래킹 기본정보 및
2) 인터페이스 정보 에서
선택

1) 트래킹 기본 정보

☐ 트래킹 ID
☐ 트래킹 시간
☒ 상태
☐ 레코드 처리 건수
☐ 데이터 처리량
☐ 데이터 압축 구분
☐ 처리소요시간
☐ 처리할 노드 수
☐ 처리한 노드 수
☐ 에러 노드 수
☐ 에러코드(대표)
☐ 에러메시지(대표)
☐ 수신 노드 수

2) 인터페이스 정보

☐ 업무
☐ 인터페이스명
☐ 인터페이스 ID
☐ 연계방식
☐ 데이터처리방향
☐ 데이터처리방식
☐ 처리방식
☐ 송신시스템
☐ 송신리소스
☐ 수신시스템
☐ 수신리소스
☐ 연계시스템

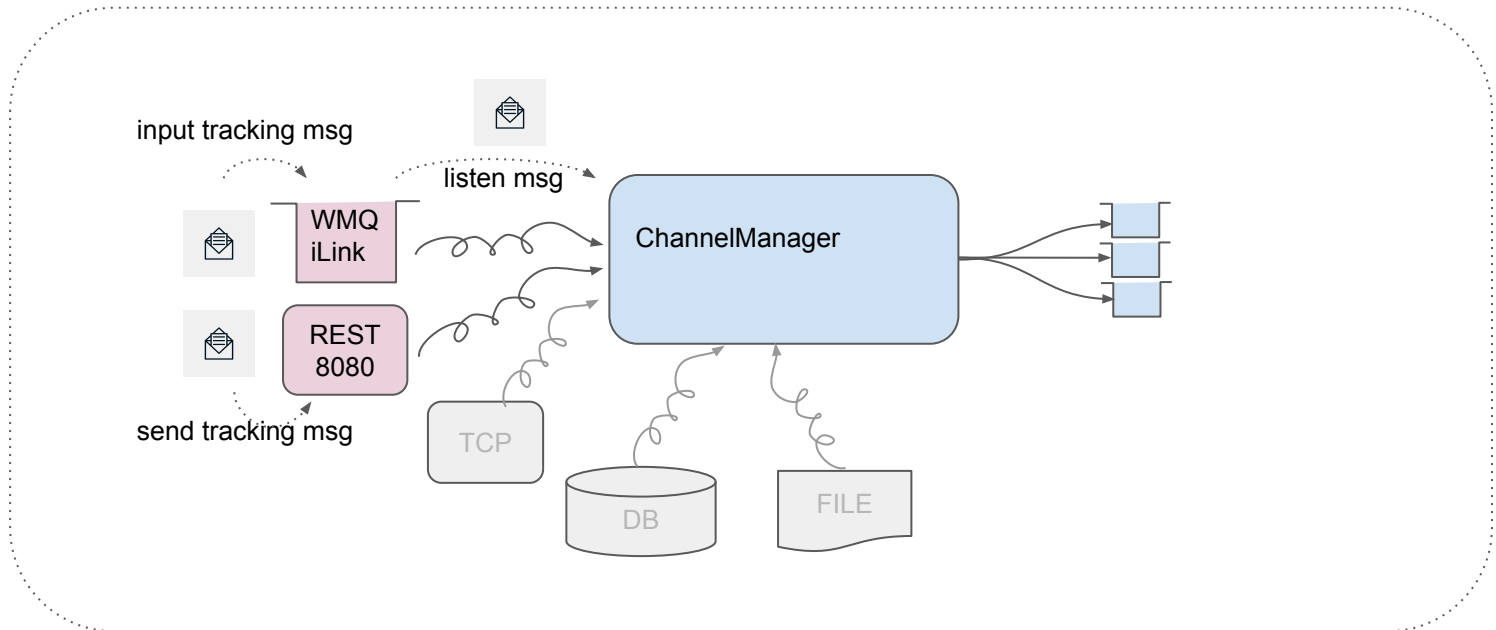
5.SW 설계

5.4.기능 상세 설계

5.4.3.채널 매니저

다채널 트래킹 메시지 소스 입력을 처리하는 매니저 기능을 설계한다.

- ❑ 동시에 여러 입력채널로 부터 트래킹메시지 유입이 가능하도록 한다.
- ❑ 리스너는 개별적으로 시작 종료 및 변경 적용이 가능하도록 한다.
- ❑ 리스너 상태를 모니터링 가능하도록 한다.
- ❑ 개발 범위는 WMQ / iLink / REST 까지로 한정한다.



5.SW 설계

5.4.기능 상세 설계

5.4.3.채널 매니저

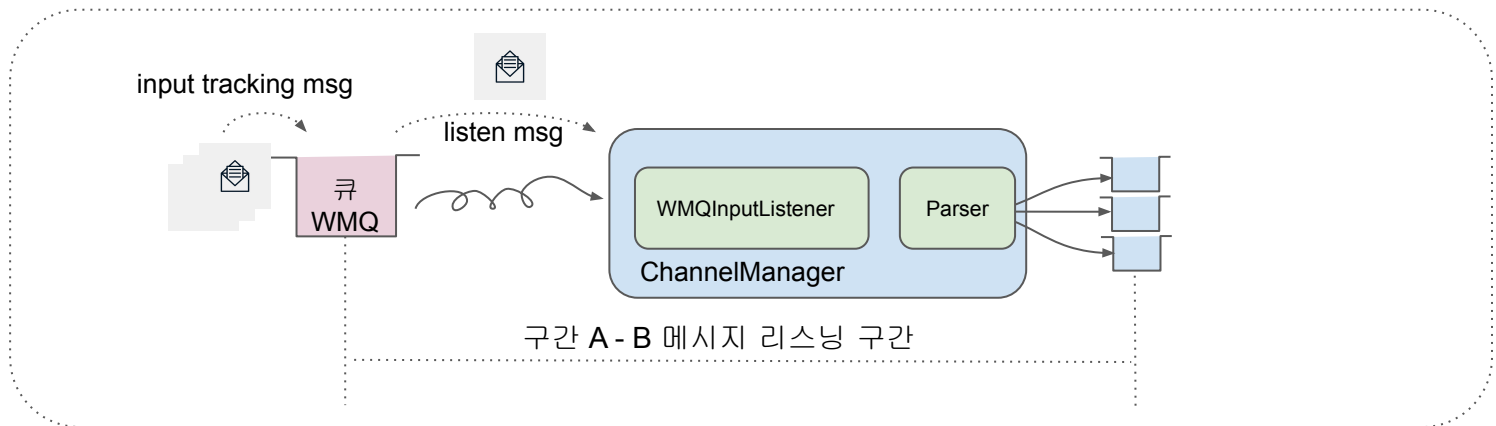
5.4.3.1.WMQInputListener

WMQ의 큐를 입력 채널로 관리하는 리스너

1) 개발 내용

- ❑ WMQ 큐에 **INPUT** 되는 트래킹 메시지를 리스닝한다.
- ❑ 트래킹 메시지를 **TrackingMessage** 로 파싱한다.
- ❑ 파싱 처리 성능 수치를 관리한다.
- ❑ 파싱한 **TrackingMessage**를 할당된 캐시에 저장한다.
- ❑ 구간 **A - B** 특정 시점 처리량(**TPS**)을 관리한다.
- ❑ 트랜잭션 처리는 **WMQ** 의 큐로 부터 메시지 **GET**을 시작으로 해서 캐시 **INPUT**을 완료하기 까지를 처리 **UNIT** 으로 본다.

인터페이스
TraceListener
구현 클래스
MQTraceListener
구현 함수
trace(TraceEvent te)



5.SW 설계

5.4.기능 상세 설계

5.4.3.채널 매니저

5.4.3.1.WMQInputListener

2) 메인 프로퍼티

프로퍼티	설명
qmgrName	큐매니저명
qmgrp	큐매니저 서버 IP
qmgrPort	큐매니저 서버 PORT
queueName	큐명
commitCount	A - B 구간 트랜잭션 처리 commit 단위
messageSetName	parser 가 참조할 메시지셋
readTimeout	큐에 대한 메시지 대기 시간
maxCacheWaitTime	캐시가 한계에 달했을 때 대기 시간
tps1	A - B 구간 처리 속도

5.SW 설계

5.4.기능 상세 설계

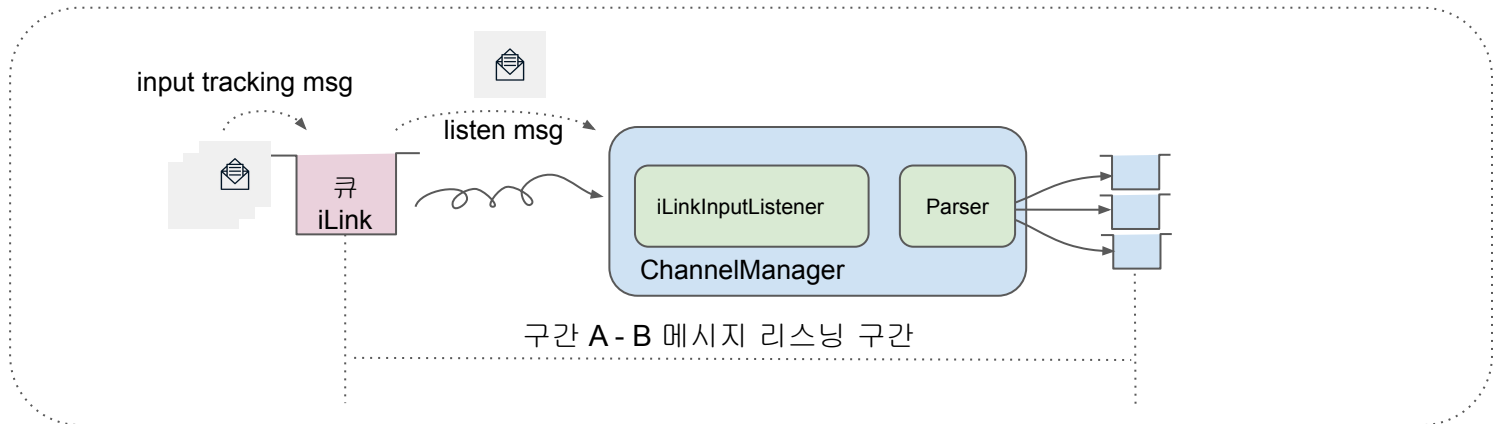
5.4.3.채널 매니저

5.4.3.2.iLinkInputListener

iLink의 큐를 입력 채널로 관리하는 리스너

1) 개발 내용

- ❑ iLink 큐에 **INPUT** 되는 트래킹 메시지를 리스닝한다.
- ❑ 트래킹메시지를 **TrackingMessage** 로 파싱한다.
- ❑ 파싱 처리 성능 수치를 관리한다.
- ❑ 파싱한 **TrackingMessage**를 할당된 캐시에 저장한다.
- ❑ 구간 **A - B** 특정 시점 처리량(**TPS**)을 관리한다.
- ❑ 트랜잭션 처리는 iLink 의 큐로 부터 메시지 **GET**을 시작으로 해서 캐시 **INPUT**을 완료하기 까지를 처리 **UNIT**으로 본다.



5.SW 설계

5.4.기능 상세 설계

5.4.3.채널 매니저

5.4.3.2.iLinkInputListener

2) 메인 프로퍼티

프로퍼티	설명
qmgrName	큐매니저명
qmgrp	큐매니저 서버 IP
qmgrPort	큐매니저 서버 PORT
queueName	큐명
commitCount	A - B 구간 트랜잭션 처리 commit 단위
messageSetName	parser 가 참조할 메시지셋
readTimeout	큐에 대한 메시지 대기 시간
maxCacheWaitTime	캐시가 한계에 달했을 때 대기 시간
tps1	A - B 구간 처리 속도

5.SW 설계

5.4.기능 상세 설계

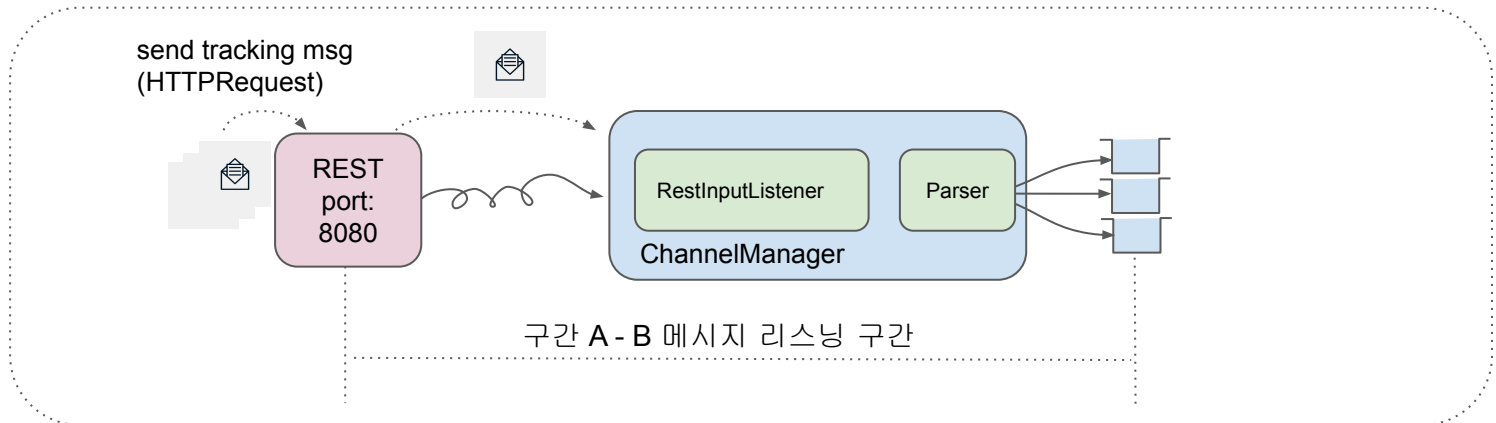
5.4.3.채널 매니저

5.4.3.3.RestInputListener

RESTful 서비스를 입력 채널로 관리하는 리스너

1) 개발 내용

- ❑ HTTP RESTful 서비스 로 요청되는 트래킹 메시지를 리스닝한다.
- ❑ 트래킹메시지를 **TrackingMessage** 로 파싱한다.
- ❑ 파싱 처리 성능 수치를 관리한다.
- ❑ 파싱한 **TrackingMessage**를 할당된 캐시에 저장한다.
- ❑ 구간 **A - B** 특정 시점 처리량(TPS)을 관리한다.
- ❑ 트랜잭션 처리는 HTTP 리스너로 수신된 요청 메시지를 캐시 **INPUT**을 완료하기 까지의 처리를 하나의 **UNIT**으로 본다.



5.SW 설계

5.4.기능 상세 설계

5.4.3.채널 매니저

5.4.3.3.RestInputListener

2) 메인 프로퍼티

프로퍼티	설명
listeningPort	서비스 포트
commitCount	A - B 구간 트랜잭션 처리 commit 단위
messageSetName	parser 가 참조할 메시지셋
maxCacheWaitTime	캐시가 한계에 달했을 때 대기 시간
tps1	A - B 구간 처리 속도

5.SW 설계

5.4.기능 상세 설계

5.4.4.Configuration 매니저

트래킹 시스템의 환경 설정 값들을 관리하는 기능을 설계한다.

패키지	속성	설명
channel.listener.qmgr	name	큐매니저명
	ip	큐매니저 서버 IP
	port	큐매니저 서버 PORT
	queue	큐명
	read.timeout	큐에 대한 메시지 대기 시간
channel.manager	commitcount	A - B 구간 트랜잭션 처리 commit 단위
	messageset	parser 가 참조할 메시지셋
	max.cache.waittime	캐시가 한계에 달했을 때 대기 시간

5.SW 설계

5.4.기능 상세 설계

5.4.5.Preload 매니저

트래킹 시스템의 성능 향상을 위해 필요한 데이터를 캐싱 및 관리 하는 기능을 설계한다.

1) 프리로딩 기능 옵션

- ❑ 프리로딩 데이터 캐시 업데이트 (주기 설정)
- ❑ 환경설정값의 실시간 반영

2) 프리로딩 데이터 리스트

- ❑ 인터페이스
- ❑ 시스템
- ❑ 서버(옵션)
- ❑ 메시지셋
- ❑ 환경설정

think about

- ❑ 트래킹 시스템은 데이터베이스를 직접 액세스 가능한 아키텍처으므로 환경설정 값들도 데이터베이스에 저장 관리한다.

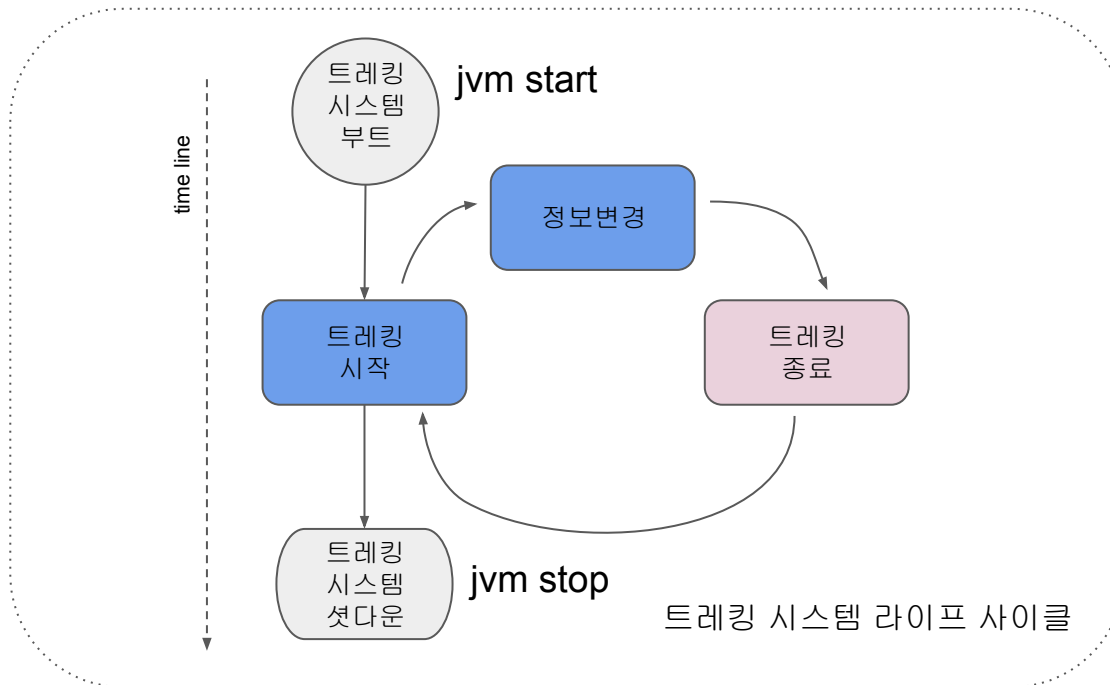
5.SW 설계

5.4.기능 상세 설계

5.4.6.Control 매니저

트레킹 시스템의 부트, 셧다운, 시작, 종료, 테스트, 정보 변경 등 컨트롤 기능을 설계한다.

- ❑ 부트(**boot**), 셧다운(**shutdown**)는 JVM 시작 종료를 의미한다.
- ❑ 시작(**start**), 종료(**end**)는 트레킹 작업 스레드의 시작 종료를 의미한다.
- ❑ 트레킹작업에 영향을 주는 정보변경은 실시간 반영되도록 한다.
- ❑ 테스트는 현재 트레킹시스템의 동작 상태를 체크하고 테스트 트레킹을 발생시킨다.





TraceMessage