20230801

TINYIOT

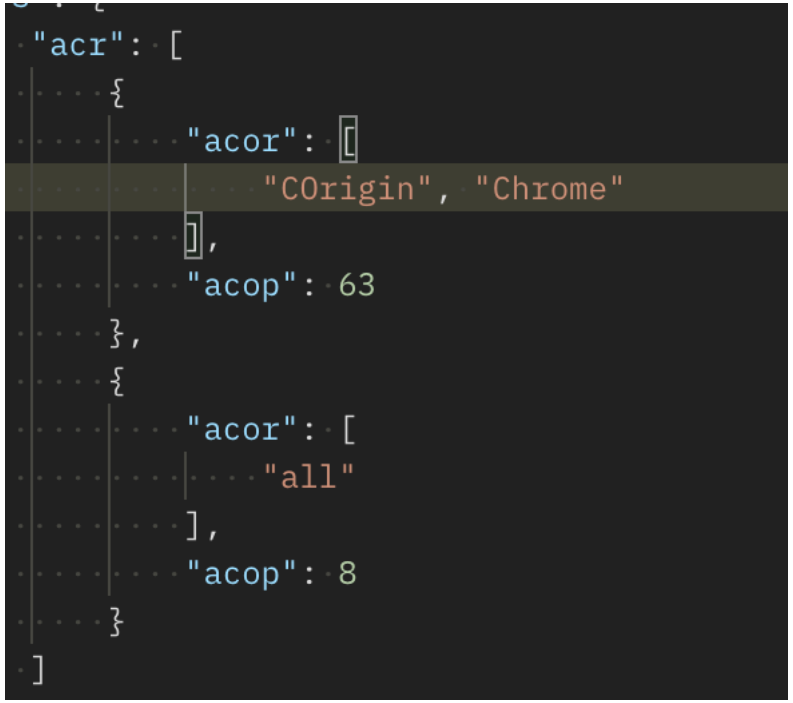
이번 주 진행 상황

1. TinyIoT RemoteCSE Update구현 완료
2. TestCase 정리
3. ACP Update버그 수정
4. AE Origin 검증 추가
5. CNT cr attr추가
6. cJSON으로 이전(고려)

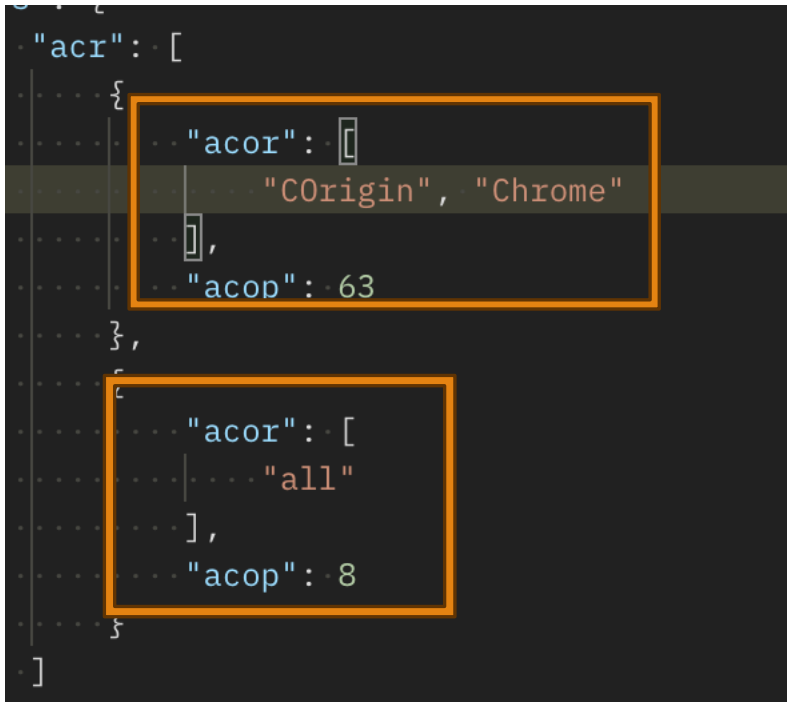
ACP Update버그 수정

맞는 형태
Pv_acor = “Corigin, Chrome, all”
Pv_acop = “63, 63, 8”

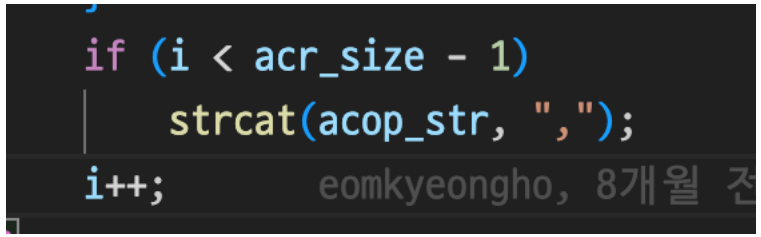
기존 출력
Pv_acor=“Corigin, Chromeall
Pv_acop = “63,638”



ACP Update버그 수정



각 칸을 처리하고 다음 칸으로 갈 때 , 를 삽입하는 코드 누락



AE Origin 검증

```
// AE Settings
// #define ALLOW_AE_ORIGIN "C*,S*" , no blankspace allowed
#define ALLOW_AE_ORIGIN "C*,S*"
```

CNT – cr attribute

ACME cr관련 test 통과

```
Delete <CNT> under <CNT> ... ok
test_createCNTWithCreatorWrong (__main__.TestCNT)
Create <CNT> with creator attribute (wrong) -> Fail ... ok
test_createCNTWithCreator (__main__.TestCNT)
Create <CNT> with creator attribute set to Null ... ok
test_deleteCNTByUnknownOriginator (__main__.TestCNT)
Delete <CNT> with wrong originator -> Fail ... ok
test_deleteCNTByAssignedOriginator (__main__.TestCNT)
Delete <CNT> with correct originator ... ok
test_createCNTUnderCSE (__main__.TestCNT)
Create <CNT> under <CB> with admin Originator ... ok
```

cr의 특성 :

- Cr속성을 추가하고 싶으면 요청에 ”cr”:NULL 을 보내야 함. 해당 attr이 있으면 request의 fr이 CR로 삽입됨.

- Mobius는 cr에 무조건 cr을 삽입함.

CNT – cr attribute

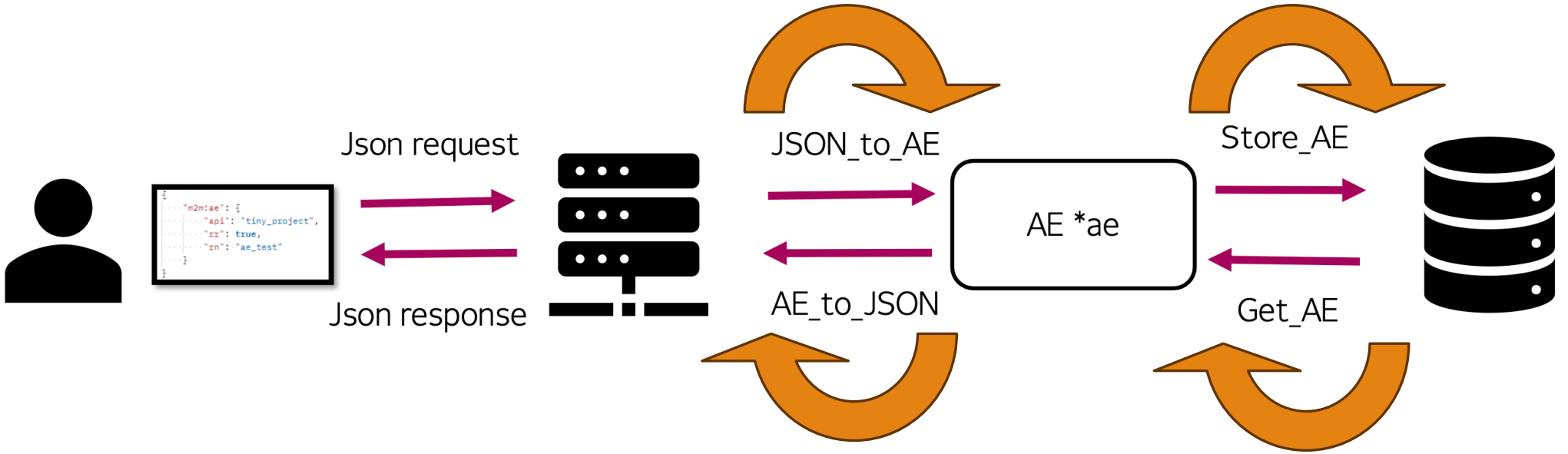
TS0001 10.1.2

The Receiver shall check whether a *creator* attribute is included in the ***Content*** parameter of the request. If included, the *creator* attribute shall not have a value in the ***Content*** parameter of the request. If the *creator* attribute is included in the request and the *creator* attribute is supported for the type of resource being created, then the Receiver shall include the *creator* attribute in the resource to be created. The Receiver shall assign a value equal to the value carried in the ***From*** request parameter. In the event that the originator provides a value for the *creator* attribute within the request, this request shall be deemed invalid.

On the other hand if the *creator* attribute is not included in the ***Content*** parameter of the request, then the Receiver shall not include the *creator* attribute in the resource to be created.

http

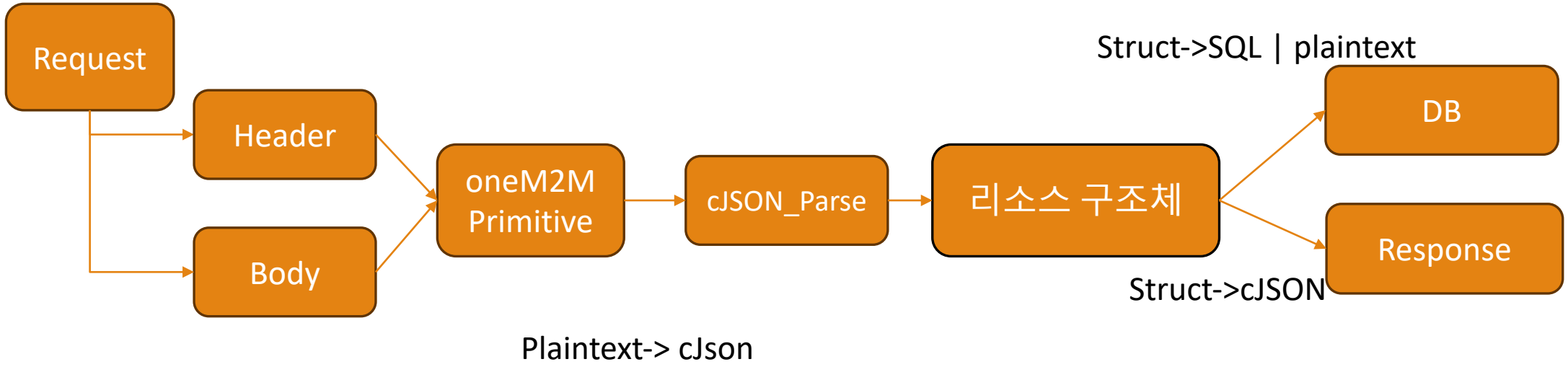
현재 CRUD



Body를 Parse 후 구조체에 할당, 구조체를 다시 cJSON으로 변환 후 Response

http

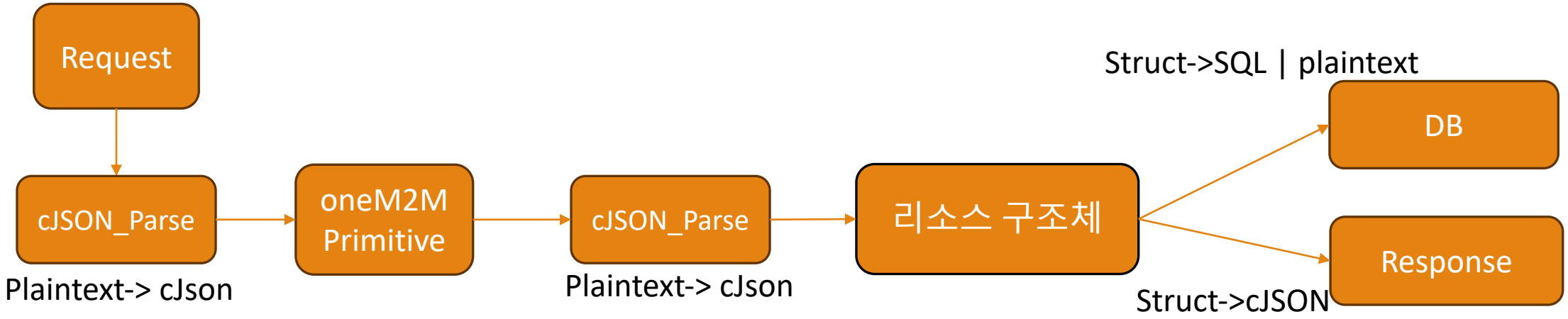
현재 CRUD



Request의 Header을 파싱하고, Body를 PC에 삽입
Parse 후 구조체에 할당, 구조체를 다시 cJSON으로 변환 후 Response

mqtt

현재 CRUD



Json으로 온 Request를 Parse 후, oneM2MPrimitive구조체로 각각 매핑,
PC를 다시 parse한 뒤, 리소스 구조체로 매핑.
리소스 구조체를 다시 cJSON으로 Parse한 뒤 Print.

http

현재 cRUD

매 요청마다 구조체를 다시 cJSON으로 변환 후 Response

cJSON으로 전환

- 장점
  - 매번 struct->cjson과정을 거칠 필요 없음.
  - 구조체 내에서 배열을 표기하기 위해 사용하는 기믹이 필요 없어짐
  - ACP의 pv:{acr:{acor:””, acop:””}}, acr:{acor:””,acop:””} }같은 복잡한 구조를 쉽게 해결함.
    (기존 방식은 pv_acor, pv_acop해서 ,로 구분함.)

- 단점
  - 기존엔 메모리 1회 참조로 끝날 일을 cJSON함수를 통해서 접근해야 함
  - cJSON 라이브러리에 버그가 있을 시 수정어려움.

Berkeley DB

DB를 이원화 할 필요가 굳이 있나?

-> 메모리 소모량도 유의미하게 차이나지 않음

-> SQLite의 속도가 유의미하게 느리지 않음

-> 두 DB간의 지원하는 기능도 다르지 않음

-> 제공 API의 수준 차이가 많이 남.

- 직접 작성한 코드는 유지보수가 어려움