SigmaCall

시스템 구성 및 API 연동 기술문서

System Architecture · PC Agent · REST/WebSocket · EMR Integration · Firmware Update

항목	내용
문서 버전	1.2
기준일	2026-10-21
문서 성격	현행 구현 사양 및 외부 연동 기술참조
정식 펌웨어 기준	Server v1.0.12 / Client v1.0.11
대상	EMR/차트 개발자, 시스템 연동 개발자, 운영·유지보수 담당자

1. 문서 목적과 범위

SigmaCall 은 복수 치료 베드의 치료용 타이머 단말, 중앙 SigmaCall 서버, PC Agent, Dashboard 및 EMR/외부 프로그램 연동 계층으로 구성되는 치료실 운영 시스템이다. 본 문서는 각 구성요소의 역할과 데이터 흐름, 서버 기준 타이머 상태, 치료 정량정보, 호출·메시지 처리, PC Agent 공개 REST API, WebSocket 이벤트 스트림, EMR 동기화, 환자 이름 표시, 펌웨어 업데이트 및 USB Serial 명령을 하나의 기술참조로 정리한다.

외부 프로그램은 원칙적으로 PC Agent 의 HTTP/JSON API 와 WebSocket 을 사용한다. USB Serial 명령은 장치 직접 제어 또는 진단·Agent 구현 확인을 위한 하위 인터페이스로 요약한다.

2. 전체 시스템 구성

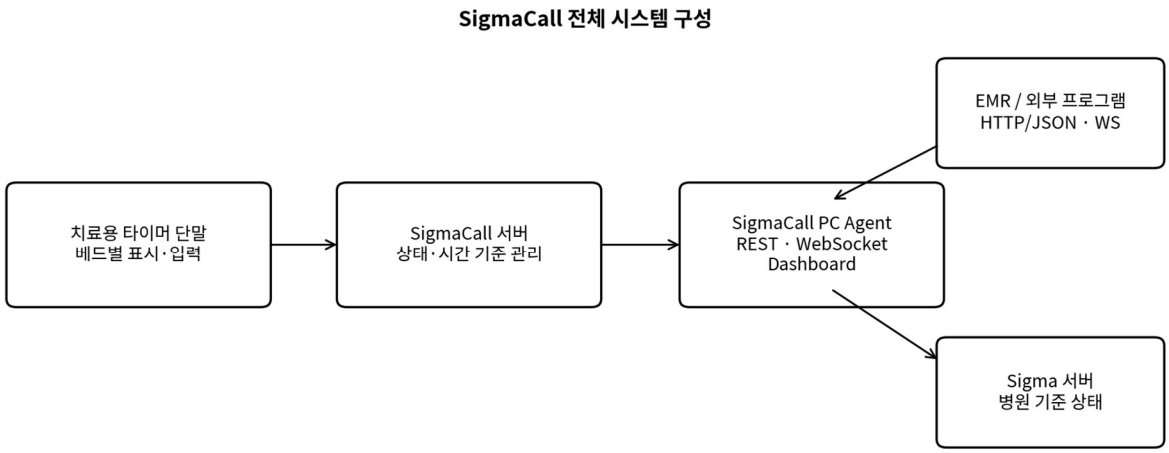


그림 1. SigmaCall 전체 시스템 구성

구성요소	주요 역할	연동
치료용 타이머 단말	베드별 치료 선택, 타이머 표시, 치료 정량정보 입력/표시, 환자호출·진료호출, 메시지 조회	SigmaCall 서버와 장치 통신
SigmaCall 서버	베드 상태 집계, 치료 타이머 기준 상태·시작/종료 시각 관리, 호출·메시지 중계	PC Agent 와 USB Serial
PC Agent	서버 상태 캐시, REST/OpenAPI, WebSocket, Dashboard, EMR 동기화, 펌웨어 업데이트	HTTP/JSON, WebSocket, USB Serial
EMR/외부 프로그램	베드 상태 조회·제어, 환자 이름, 치료·호출·메시지 연동	PC Agent 공개 API
Sigma 서버	병원 기준 치료/배정 상태 제공 및 장치 이벤트 수신	PC Agent 내부 연동

3. 서버/클라이언트 구조

현재 장치 구조는 SigmaCall 서버 1 대와 다수의 베드 단말로 구성된다. 서버는 PC Agent 에서 전달된 명령을 해석하여 대상 단말 상태에 반영하고, 단말에서 발생한 치료·호출·메시지 요청을 수집하여 PC Agent 에 전달한다.

항목	현행 기준
단말 heartbeat	기본 5 초
오프라인 판단	30 초
시간 기준	SigmaCall 서버의 기준 시간
Client ID	1~65535 범위의 외부 식별자
치료 목록	최대 8 개 항목
상태 조회	PC Agent 가 LIST 스냅샷을 수집하여 HTTP/WS 상태로 제공

4. 서버 권위 타이머 및 시간 동기화

치료 타이머 상태 흐름



그림 2. 치료 타이머 상태 흐름

타이머의 기준 상태는 SigmaCall 서버가 관리한다. 치료 시작 시 서버가 시작·종료 기준을 생성하고 단말과 PC Agent 는 해당 상태를 기준으로 표시한다. PC Agent 는 LIST 스냅샷의 서버 상태를 기준으로 RUN 남은 시간과 END 경과시간을 계산한다.

상태	표시 원칙	정량정보
IDLE	대기 상태	현재 선택 치료와 선택 용량 상태
RUN	남은 시간 표시	현재 치료용량 표시 및 변경
END	종료 후 경과시간 표시	종료 시점의 치료용량 유지
STOP 이후	대기 상태 복귀	선택 용량은 초기 상태로 복귀

5. 치료 항목과 치료 정량정보

치료 항목은 치료명, 치료시간, 현재 치료용량, 기본 치료용량 및 단위를 포함할 수 있다. PC Agent 의 Treatment 모델은 다음 필드를 사용한다.

필드	형식 / 의미

필드	형식 / 의미
name	치료명, 1~64 글자
durationSec	치료시간, 1~14400 초
treatmentDose	현재 치료용량, 0~9999
defaultTreatmentDose	기본 치료용량, 0~9999
treatmentUnit	치료단위, 최대 32 글자

```
{"name":"ICT","durationSec":900,"treatmentDose":12,"defaultTreatmentDose":10,"treatmentUnit":"개"}
```

6. 치료용 타이머 단말 표시 및 입력

입력/상태	동작
시작/확인	대기 중 선택 치료 시작, 설정 중 확정
종료/취소	치료 종료 또는 설정 취소
환자 호출	환자호출 상태 생성
진료 호출	진료호출 상태 생성
설정/로터리	치료 선택, 시간 조정, 치료용량 편집
메시지 조회	현재 베드의 메시지 조회 요청

단말 화면은 치료명, 남은 시간 또는 종료 상태, 선택적으로 치료용량과 단위를 표시한다. 표시 상태는 서버 기준 타이머와 동기화된다.

7. 환자호출 및 진료호출

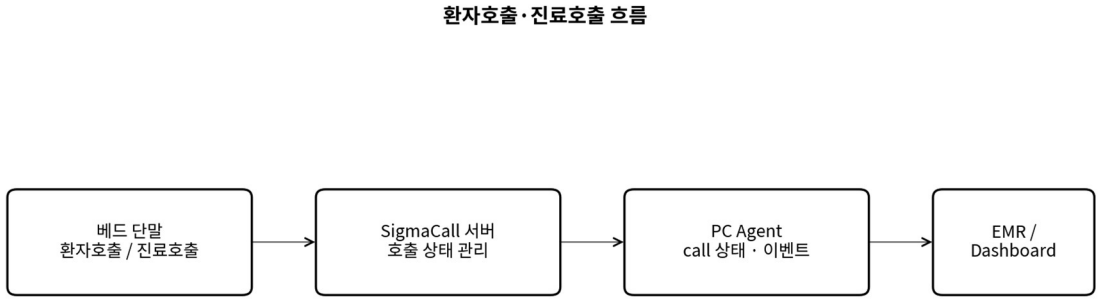


그림 3. 환자호출·진료호출 상태 및 API 흐름

각 Client 상태의 call 객체는 patientCallActive, doctorCallActive 및 active 를 제공한다. 외부 프로그램은 다음 REST API 로 호출 상태를 조회·제어한다.

Method	Path	기능
POST	/v1/calls/patient/clear	지정 베드의 환자호출 해제
POST	/v1/calls/doctor	지정 베드의 진료호출 활성화
POST	/v1/calls/doctor/clear	지정 베드의 진료호출 해제
GET	/v1/overview	통합 상태에서 call 객체 조회
GET	/v1/clients	베드별 call 객체 조회

```
POST /v1/calls/doctor
Content-Type: application/json

{"client_id":2}
```

8. 메시지 조회 및 표시

PC Agent 는 베드별 자동 메시지와 수동 메시지를 구분하여 관리하며, 단말의 메시지 조회 요청을 MessageRequest 로 제공한다. 외부 프로그램은 WebSocket 이벤트 또는 메시지 요청 API 로 요청을 확인하고 응답할 수 있다.

Method	Path	요청 / 기능
PUT	/v1/clients/{client_id}/message/auto	{"message":"...","request_id":17} / 자동 메시지 저장
DELETE	/v1/clients/{client_id}/message/auto	자동 메시지 초기화
PUT	/v1/clients/{client_id}/message/manual	{"message":"...","request_id":17} / 수동 메시지 저장
DELETE	/v1/clients/{client_id}/message/manual	수동 메시지 초기화
POST	/v1/message-requests	{"status":"pending"} 또는 {"status":"all"} / 요청 목록
POST	/v1/message-requests/{request_id}/response	{"action":"respond","message":"..."} 또는 {"action":"none"}

```
POST /v1/message-requests/17/response
Content-Type: application/json

{"action":"respond","message":"검사실 이동 전 금식 유지"}
```

9. PC Agent 상태 모델과 외부 API

PC Agent 는 SigmaCall 서버의 상태를 수집하여 외부 프로그램에 HTTP/JSON 과 WebSocket 으로 제공한다. 기본 API 주소는 http://<Agent PC IP>:48765 이며 동일 PC 에서는 http://127.0.0.1:48765 를 사용할 수 있다.

리소스	주소
OpenAPI	http://<Agent PC IP>:48765/openapi.json
Swagger UI	http://<Agent PC IP>:48765/docs
ReDoc	http://<Agent PC IP>:48765/redoc
Dashboard	http://<Agent PC IP>:48765/dashboard
WebSocket	ws://<Agent PC IP>:48765/v1/stream

9.1 상태 조회 API

Method	Path	성공 응답
GET	/v1/overview	{agent, clients, pendingMessageRequests, summary}
GET	/v1/clients	{items:[Client]}

```
GET /v1/overview

{
  "agent": {
    "connected": true,
    "apiBaseUrl": "http://192.168.0.50:48765",
    "websocketUrl": "ws://192.168.0.50:48765/v1/stream",
    "emrSync": {"required": false, "syncGeneration": 2}
  },
  "clients": [],
  "pendingMessageRequests": [],
  "summary": {}
}
```

9.2 타이머 및 치료 API

Method	Path	Request JSON
POST	/v1/timers/start	{"client_id":2,"duration_sec":900,"treatment_name":"ICT"}
POST	/v1/timers/stop	{"client_id":2}
POST	/v1/timers/default	{"client_id":2,"duration_sec":900,"treatment_name":"침","default_dose":10,"treatment_unit":"개"}
POST	/v1/timers/selection	{"client_id":2,"duration_sec":900,"treatment_name":"ICT","default_dose":0,"treatment_unit":""}
POST	/v1/timers/dose	{"client_id":2,"dose":12,"default_dose":10,"treatment_unit":"개"}
POST	/v1/timers/update	{"client_id":2,"delta_minutes":5}
PUT	/v1/clients/{client_id}/treatments	{"items":[Treatment,...]}

duration_sec 는 1~14400 초이며 delta_minutes 는 0 을 제외한 -1440~1440 정수다. 치료 목록 items 는 1~8 개다.

9.3 환자 이름 API

Method	Path	Request / Response
GET	/v1/clients/{client_id}/patient-name	{clientId,patientName,displayName,maskingApplied}
PUT	/v1/clients/{client_id}/patient-name	{"patient_name":"홍길동"} / 동일 상태 반환
DELETE	/v1/clients/{client_id}/patient-name	이름 초기화 상태 반환

9.4 EMR 동기화 API

Method	Path	기능
POST	/v1/emr/sync	EMR 기준 상태 일괄 반영
POST	/v1/emr/sync/complete	현재 동기화 세대 완료 처리

```
POST /v1/emr/sync
Content-Type: application/json

{
  "sync_generation": 2,
  "treatments": [{"name":"ICT","durationSec":900}],
  "clients": [
    {"client_id":2,"default_treatment_name":"ICT","default_duration_sec":900}
  ]
}
```

9.5 설정·진단·관리 API

Method	Path	기능
GET	/v1/low-level/health	Agent 및 장치 연결 상태
GET	/v1/low-level/ports	사용 가능한 Serial 포트 목록
GET / PUT	/v1/low-level/config	Agent 설정 조회/저장
POST	/v1/low-level/connection/open	Serial 연결 열기
POST	/v1/low-level/connection/close	Serial 연결 닫기
POST	/v1/low-level/clients/id	Client ID 설정
GET	/v1/low-level/server/events	서버 이벤트 조회
POST	/v1/low-level/server/commands/treatments	전체 치료 목록 서버 반영
POST	/v1/low-level/server/commands/raw	Raw Serial 명령 전송
POST	/v1/low-level/server/button	가상 버튼 입력
POST	/v1/low-level/server/rotary	가상 로터리 입력
GET / PUT	/v1/sigma-server	Sigma 서버 연결 설정
POST	/v1/sigma-server/discover	Sigma 서버 검색
POST	/v1/sigma-server/connect	Sigma 서버 연결
GET / PUT	/v1/ui-config	대시보드 UI 설정
GET / POST	/v1/sounds/files	사용자 소리 파일 조회/등록
POST	/v1/system/restart	SigmaCall 장치 시스템 재시작

10. 환자 이름 표시

환자 이름은 PC Agent 의 베드별 상태로 관리하며 patient-name API 를 통해 조회·저장·초기화한다. patientName 은 저장된 이름이고 displayName 은 대시보드 표시용 이름이다.

```
PUT /v1/clients/2/patient-name
Content-Type: application/json

{"patient_name":"홍길동"}
```

```
{
  "clientId": 2,
  "patientName": "홍길동",
```

```
  "displayName": "홍*동",
  "maskingApplied": true
}
```

11. WebSocket 이벤트 스트림

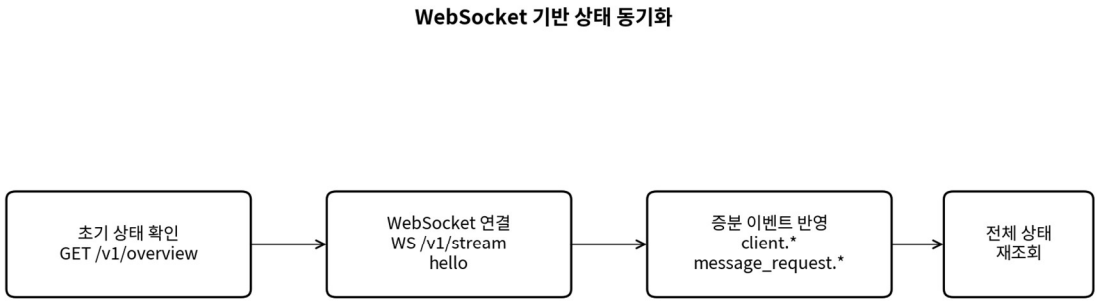


그림 4. WebSocket 이벤트 스트림 운용 흐름

외부 프로그램은 WS /v1/stream 을 통해 주요 상태 변경을 실시간으로 수신할 수 있다. 연결 직후 hello 프레임이 전송되고 이후 event 프레임이 전달된다.

```
{"type":"hello","sequence":42,"apiVersion":"1.0.0","status":{"connected":true},"resync":{"required":false,"
syncGeneration":2}}
```

```
{"type":"event","event":{"sequence":43,"timestamp":"2026-10-
21T01:23:45.678Z","event_type":"client.timer.updated","payload":{"clientId":2}}}
```

event_type	의미
clients.snapshot	전체 베드 상태 스냅샷
client.timer.updated	타이머 상태 변경
client.selection.updated	현재 선택 치료 변경
client.default.updated	기본 치료 변경
client.identity.updated	Client ID 관련 변경
client.call.updated	환자/진료 호출 상태 변경
client.message.auto.updated / cleared	자동 메시지 변경
client.message.manual.updated / cleared	수동 메시지 변경
message_request.created / responded	메시지 요청 생성/응답
agent.resync.required / completed	EMR 동기화 상태 변경
connection.opened / closed	장치 연결 상태 변경
firmware.update	펌웨어 작업 상태 변경

12. EMR 및 외부 프로그램 연동 실시예

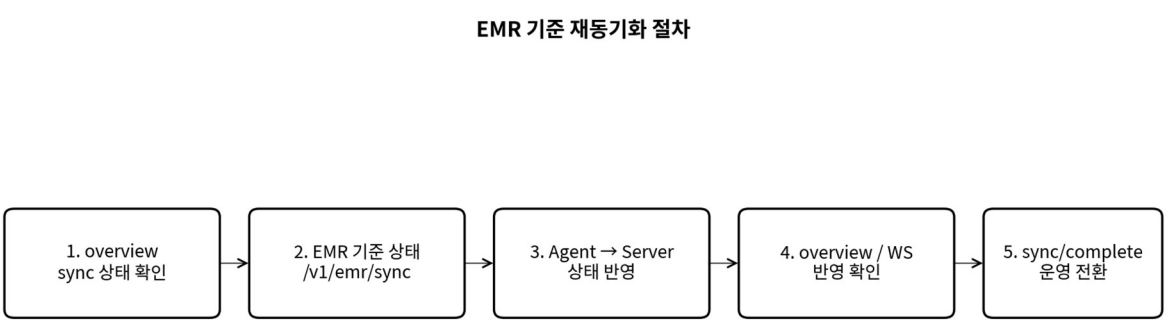


그림 5. EMR 기준 재동기화 절차

12.1 WebSocket 방식

- GET /v1/overview 로 Agent 연결 상태와 emrSync 상태를 확인한다.
- 동기화가 필요한 경우 POST /v1/emr/sync 로 EMR 기준 데이터를 반영한다.
- WS /v1/stream 에 연결하고 hello 를 수신한다.
- clients.snapshot 또는 overview 를 초기 상태로 사용한다.
- client.timer.updated, client.call.updated 등 이벤트를 sequence 순서로 반영한다.
- 필요 시 overview 를 다시 조회하여 전체 상태를 맞춘다.

12.2 Polling 방식

- 초기 GET /v1/overview 로 전체 상태를 조회한다.
- 운영 중 GET /v1/clients 또는 GET /v1/overview 를 주기적으로 조회한다.
- timer, selected/default, message, patientCallActive, doctorCallActive 를 이전 상태와 비교한다.
- 사용자 조작은 polling 주기와 별개로 해당 POST/PUT/DELETE API 를 즉시 호출한다.

13. 펌웨어 업데이트

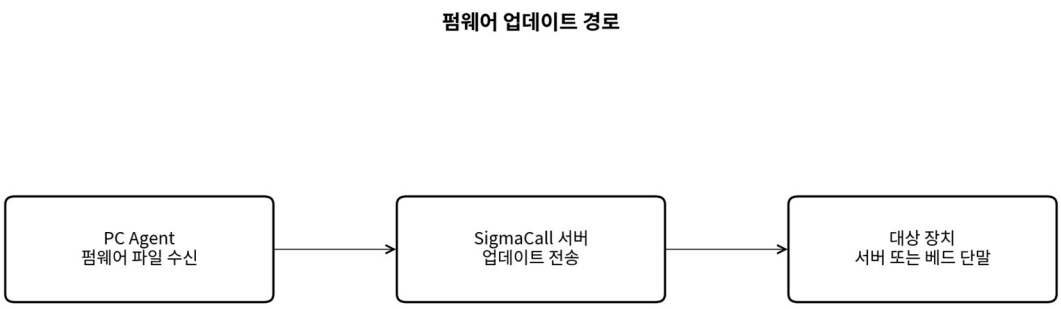


그림 6. 서버 및 베드 단말 펌웨어 업데이트 경로

Method	Path	설명
GET	/v1/firmware	서버/단말 버전 및 최근 작업 상태 조회
POST	/v1/firmware/server	SigmaCall 서버 펌웨어 업데이트 시작
POST	/v1/firmware/client	지정 client_id 단말 업데이트 시작
POST	/v1/firmware/jobs/{job_id}/cancel	진행 중 작업 취소 요청

업데이트 본문은 application/octet-stream 바이너리이며 version 쿼리를 사용한다. 클라이언트 대상 업데이트에는 client_id 를 지정한다. 작업 상태는 queued, transferring, cancelling, cancelled, completed, failed 와 progress 로 조회할 수 있다.

14. USB Serial 명령 요약

PC Agent 하위의 장치 직접 제어 인터페이스는 UTF-8 줄 단위 텍스트 프로토콜을 사용한다. 기본 통신은 460800 baud, 8N1, flow control 없음이며 명령은 LF 로 종료한다.

명령	형식 / 용도
HELP	HELP — 지원 명령 확인
LIST	LIST — 현재 Client/Timer 스냅샷
START	START <client_id> <duration_sec> <treatment_name>
STOP	STOP <client_id>
ADJUST	ADJUST <client_id> <signed_delta_sec>
DEFAULT / SETDEFAULT	기본 치료명·시간·기본용량·단위 설정
DOSE	현재/기본 치료용량 및 단위 설정
TREATMENTS / SETTREATMENTS	전체 치료 항목 목록 설정
CLEARCALL	환자호출 해제

명령	형식 / 용도
DOCTORCALL / CLEARDOCTORCALL	진료호출 활성/해제
MSGRESP / MSGNONE	메시지 요청 응답
MSGSET / MSGCLEAR	베드 메시지 설정/해제
ID / SETID / SAVECLIENTS	Client ID 레지스트리 관리
BUTTON / ROTARY	가상 입력
RESTART	장치 서버 재시작
PREPARE_UPLOAD	USB 펌웨어 업로드 준비

```
START 2 900 ICT
ADJUST 2 300
DOCTORCALL 2
MSGRESP 2 17 검사실 이동 전 금식 유지
```

LIST 응답은 Client ID, 온라인 상태, 펌웨어 버전, 타이머 상태, 선택/기본 치료, 치료용량, 호출 상태 등을 포함하는 스냅샷으로 사용된다. 외부 업무 프로그램은 이 Serial 포맷을 직접 처리하기보다 PC Agent 공개 API 를 사용하는 구성을 권장한다.

15. 구성 요약

SigmaCall 의 중심은 SigmaCall 서버가 복수 베드의 치료 타이머와 장치 상태를 관리하고, PC Agent 가 이를 외부 프로그램에서 사용할 수 있는 REST API 와 WebSocket 상태 모델로 변환하는 구조이다. 치료 선택, 치료시간, 치료용량, 환자호출, 진료호출, 메시지, 환자 표시 이름 및 펌웨어 상태를 베드 단위로 통합하여 운용할 수 있다.

외부 개발자는 GET /v1/overview 와 GET /v1/clients 를 기준 상태 조회에 사용하고, /v1/timers/*, /v1/calls/*, 메시지 API, patient-name API 및 EMR sync API 로 상태를 제어한다. 실시간 화면이나 EMR 연동은 WS /v1/stream 을 함께 사용하면 상태 변경을 즉시 반영할 수 있다.