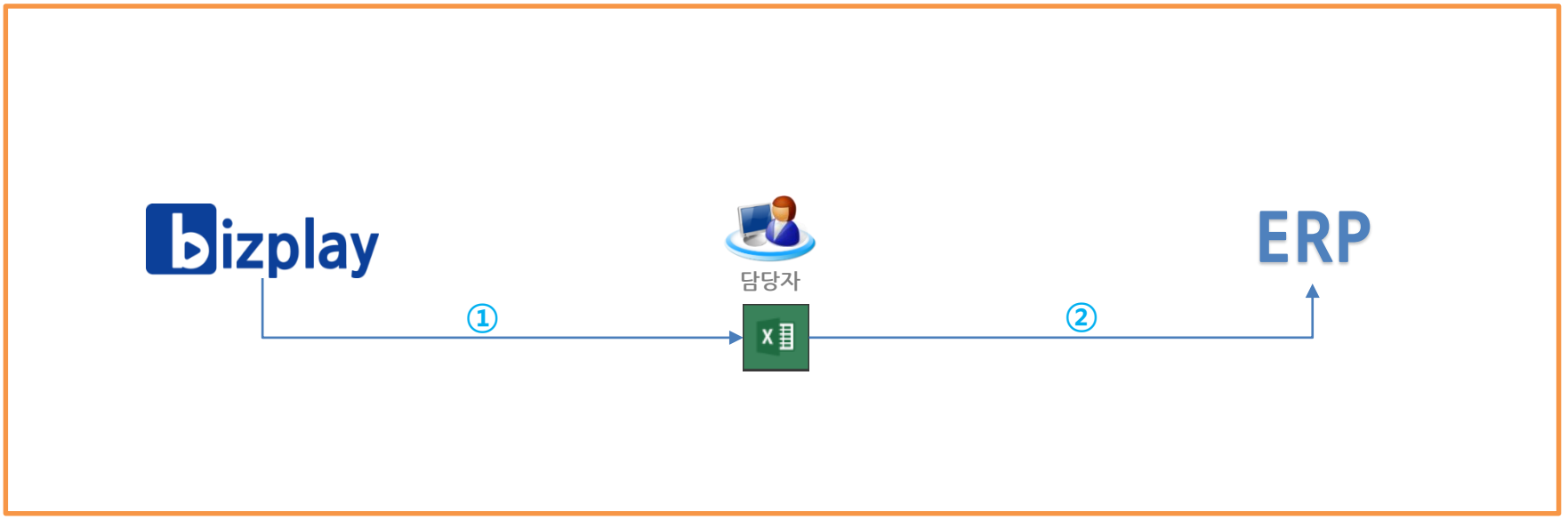
비즈플레이 자료 연동

Index

1. 연동방식

엑셀 upload

수정이 어려운 패키지 ERP를 이용하는 경우 비즈플레이에서 해당 ERP에 바로 upload가 가능한 고유 서식으로 자료를 다운 하도록 기능을 제공 하여 직접 연동 하는 방식 입니다.



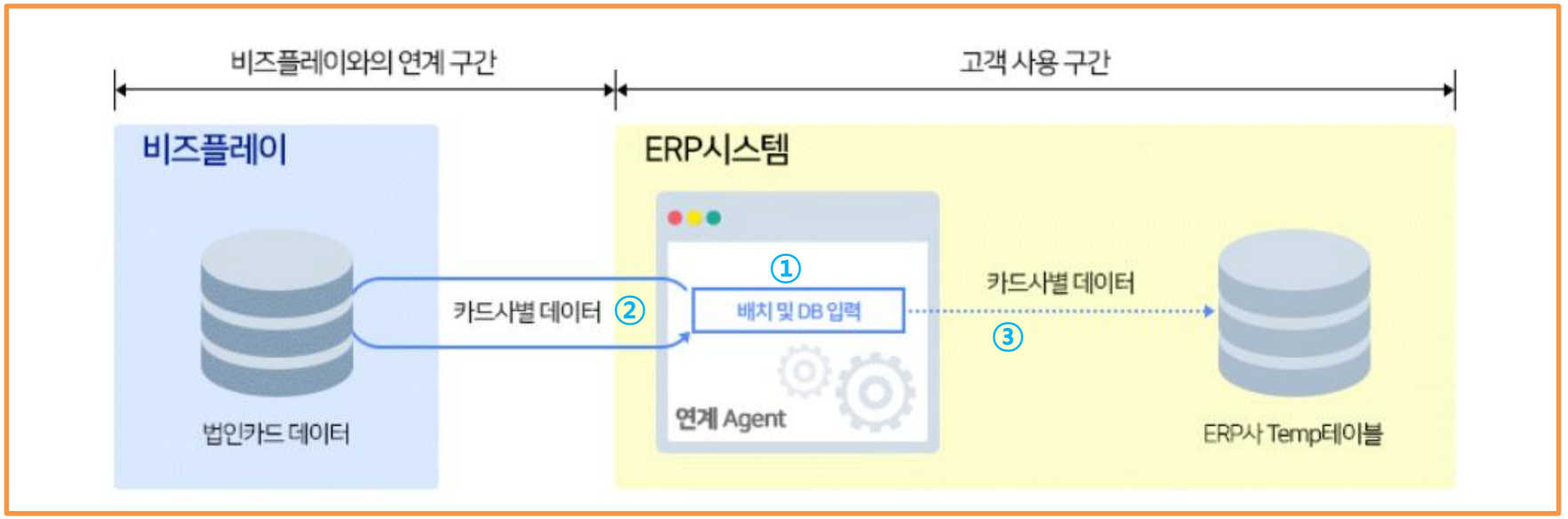
① 비즈플레이 엑셀서식 다운로드에서 원하는 서식을 선택하고 download 합니다.
② pc에 저장된 엑셀 서식을 ERP에서 제공하는 upload기능을 활용하여 저장 합니다,

- ERP에 upload제공 여부는 사용하시는 ERP 운영사에 확인하여 주세요.
- 만약 표준 서식 download를 미 제공 하는 경우 요청 하시면 추가해 드립니다.

1. 연동방식

에이전트 방식(쿠콘박스2 제공)

쿠콘박스2는 ERP 제품 독립적으로 설치되어 운영되는 Agent 프로그램으로, 비즈플레이의 경비관리 데이터를 배치 방식으로 고객사 내부의 임시 데이터베이스 테이블에 저장합니다.



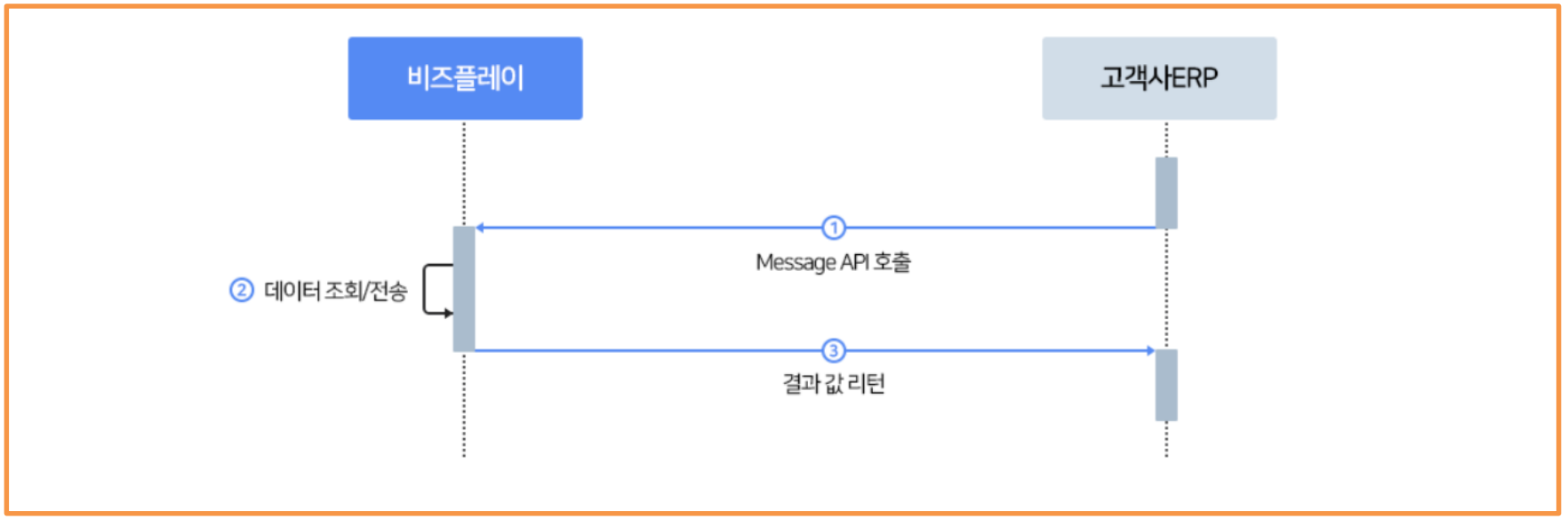
① 제공된 에이전트 데몬이 작동하여 작업을 시작 (일정시간 마다 배치 실행)
② 비즈플레이 OPEN API 호출 하여 응답 받음
③ 고객 요청 DB에 Temp Table로 저장

- 고객사 DBMS는 Oracle, MS-Sql, DB2, 파일DB 등 가능 함
- ③번 DB 저장 외 SAP RFC방식, FTP(파일전송) 등 가능 함
- Temp Table에서 내부에 활용하는 업무는 고객 또는 ERP업체가 작업 해야 합니다.

1. 연동방식

OPEN API 방식

API (Application Programming Interface)는 비즈플레이 경비관리 서비스 내에 저장된 데이터를 다양한 개발 언어를 통해서 데이터를 연동할 수 있도록 지원합니다. Restful 기반의 JSON 방식으로 지원하는 웹 기반 API 입니다



① Message API 호출을 비즈플레이로 전송함.
② 데이터 조회/전송을 수행함.
③ 요청된 결과 값을 다시 전송 함.

- 고객을 식별할 수 있는 고유 키를 발급해 드립니다.
- 운영 환경은 Https 인증방식 암호화 합니다.

1. 연동방식

특징 및 기능 비교

항목	엑셀 upload	에이전트 방식	OPEN API 방식
연동 실행 방식	사용자 작업	배치성 자동	ERP이벤트 발생 시점 or 배치성 자동
ERP 수정 개발 여부	불필요	Temp DB사용 부분 개발	API 호출 및 처리 개발
연동 소요 시간(예상)	즉시 사용	개발 시간 소요(1~2달)	개발 시간 소요(1~2달)
패키지 ERP 적용	○	X	X
자체개발 ERP 적용	△	○	○
SAP RFC방식 적용	X	○	△
비즈플레이 지원	서식 관리	에이전트 및 temp Table 자료	OPEN API 가이드 API 유지 관리
방화벽 추가 오픈	불필요	API 사용 포트 OPEN	API 사용 포트 OPEN
유지보수 원격접속 허용	X	○	X

2. 방화벽 설정

비즈플레이 IDC센터와 고객사 간

비즈플레이는 OPEN API 로 연결 되어 아래와 같은 Out-Bound 방화벽 설정이 필요 합니다(In-bound 없음)

용도	출발지 IP	도착지 URL	도착지 IP	포트	비고
개발 API 전문테스트	API 호출 서버 IP	webankdev.appplay.co.kr	211.217.152.244	80, 443	리얼오픈 이전 까지 필요
첨부 이미지 서버	API 호출 서버 IP	platform.bizplay.co.kr	183.111.151.14	80, 443	리얼공통
운영 API 서버	API 호출 서버 IP	webankapi.appplay.co.kr	183.111.151.122	80, 443	
이미지 링크	이미지 링크 필요서버 IP	platform.bizplay.co.kr	183.111.151.14	80, 443	이미지 링크 필요 시

① 비즈플레이는 목동 KT IDC센터에 금융 클라우드 센터로 운영 합니다.
② 도착지 호출은 URL 방식으로 내부 서버에서 DNS를 찾을 수 없으면 HOSTS파일에 등록해야 합니다.
③ 개발 API 전문테스트는 개발 작업 시 에만 필요 합니다.
④ 이미지 링크는 ERP나 GW에서 첨부 이미지 링크가 필요할 경우 사용 합니다.

3. API 실습

GET방식으로 외부사이트 이용하여 실습을 합니다

Protocol	HTTP (개발)/HTTPS(운영)	데이터형식	JSON
Network	인터넷망	데이터 제공방식	실시간
Base URL	(개발) http://webankdev.appplay.co.kr/gateway.do (전문테스트) http://webankdev.appplay.co.kr/api_test.jsp (POST 방식 소스보기 제공)		
	(운영) https://webankapi.appplay.co.kr/gateway.do (전문테스트) https://webankapi.appplay.co.kr/api_test.jsp		

ⓐ 개발 전문을 테스트 페이지에서 확인 합니다.

① http://webankdev.appplay.co.kr/api_test.jsp 온라인에서 열고
② 서비스 코드 목록을 '카드영수증 처리내역 조회(0411) 선택 쿼리전송 〉 팝업(확인)〉 잠시 대기 〉 결과 보임
③ 결과 확인 http://jsonviewer.stack.hu/ 에 접속 ②번 결과 전체 복사
④ 결과 확인 페이지에 붙여넣기〉 Format tab으로 이동

3. API 실습

ⓑ get 방식 API 테스트

① 메모장프로그램 실행
- http://webankdev.appplay.co.kr/gateway.do?JSONData= 복사 붙여 넣기
- 바로뒤에 **URL 인코딩 된 API 입력 값** 넣을 예정

② **URL 인코딩 된 API 입력 값 구하기**
- http://webankdev.appplay.co.kr/api_test.jsp 접속
  → 원하는 서비스 코드 목록 선택
  → JSON_IN의 오른쪽 출력값 전체 복사

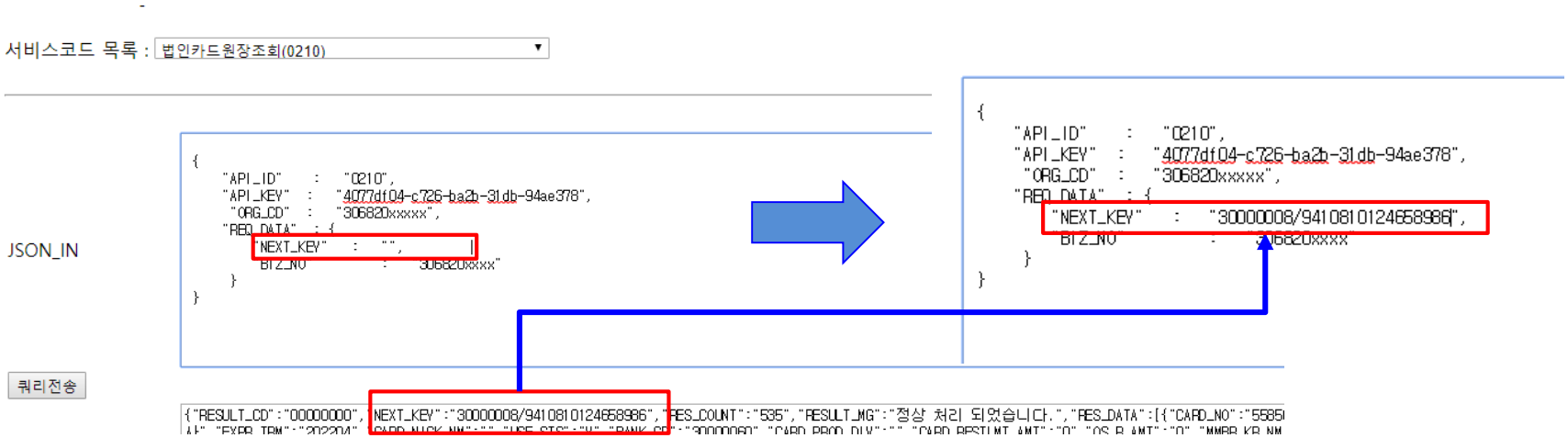
- http://www.convertstring.com/ko/EncodeDecode/UrlEncode 접속
  → '여기에 URL 인코딩하고자하는 텍스트를 붙여 넣습니다' 아래 부분에 JSON_IN의 오른쪽 출력값 붙여 넣기
  → URL 인코딩! 버튼 클릭
  → '여기에 URL 인코딩 된 텍스트를 복사' 아래 부분에 출력된 내용 전체 복사

③ 메모장에 적힌 http://webankdev.appplay.co.kr/gateway.do?JSONData= 바로 뒤에 띄어쓰기 없이 출력된 내용 붙여넣기
④ 메모장 전체 복사하여 크롬(웹브라우져) URL에 붙여넣고 엔터
⑤ 결과 확인 http://jsonviewer.stack.hu/ 에 접속 ④번 결과 전체 복사
⑥ 결과 확인 페이지에 붙여넣기〉 Format tab으로 이동

❖ **URL인코딩(base64) : Protocol로 사용하는 http URL에는 의미를 가진 문자가 있음 해당문자를 URL에 명령어로 인식하지 못하도록 암호화 하는 방법**

4. 리바운드 전문

전문통신 특성상 응답을 무한정 길게 전송이 불가능 하기 때문에 일정 건 별로 페이징 하여 반복하여 전문을 호출 합니다.



예제 법인카드 목록처럼 결과 헤더 부분에 NEXT_KEY가 값이 있는 경우 다음 페이지 존재하기 때문에 같은 전문에
NEXT_KEY 값을 그대로 넣어 다시 요청 하여야 합니다.

5. 예제소스 (for java)

```
vLog("0411", "BEGIN");

JSONObject apiCommon = new JSONObject();
JSONObject reqData = new JSONObject();
JSONObject apiResult = new JSONObject();

setInputValue(apiCommon, "API_KEY", "e11111ea-b248-84d0-0cfb-111111111111");
setInputValue(apiCommon, "ORG_CD", "2148635102");
setInputValue(apiCommon, "API_ID", "0411");
setInputValue(reqData, "BIZ_NO", "2148635102");
setInputValue(reqData, "START_DATE", "20180824");
setInputValue(reqData, "END_DATE", "20180824");
setInputValue(reqData, "REQ_CNT", "100");
setInputValue(apiCommon, "REQ_DATA", reqData);

String data = apiCommon.toString();

try {
    int req = 0;

    vLog(String.format("%03d", ++req)+" Req", data);

    apiResult = execute(data);
    String strNextKey = getNextKey(apiResult);
    vLog(String.format("%03d", req)+" Req", "NextKey:"+strNextKey);

    JSONArray resTotal = getResData(apiResult);
   /* 리바운드 처리*/
    while (strNextKey != null) {

        setInputValue(reqData, "NEXT_KEY", strNextKey);
        removeInputValue(apiCommon, "REQ_DATA");
        setInputValue(reqData, "REQ_CNT", "");
        setInputValue(apiCommon, "REQ_DATA", reqData);
```

```
        vLog(String.format("%03d", ++req)+" Req", data);

        JSONObject resTemp = execute(apiCommon.toString());

        strNextKey = getNextKey(resTemp);
        if (strNextKey == null){
            apiResult = resTemp;
        }

        vLog(String.format("%03d", req)+" Req", "NextKey:"+strNextKey);

        JSONArray resData = (JSONArray) resTemp.get("RES_DATA");

        resTotal = appendList(resTotal, resData);
    }

    /*조회 결과의 반복부 배열(RES_DATA) 크기를 이용한 Looping 처리  */
    for (int row = 0; row < resData.size(); row ++ ) {
        JSONObject ONE_REC = (JSONObject)resData.get(row);
        String strCardNo = (String)getObjValue(ONE_REC, "CARD_NO");
        String strApvDt = (String)getObjValue(ONE_REC, "APV_DT");

        // DB 처리 부분 (생략)
    }

    setInputValue(apiResult, "RES_DATA", resTotal);

    makeFile(apiNums, apiResult.toString());

    vLog("Result Json", apiResult.toString());
} catch (Exception e) {

}

vLog("0411", "END");
```

감 사 합 니 다