O-RAN SC 기반 Non-RT RIC과 AI/ML Framework 통합 분석

최호성, 이성진, 유현민, 홍인기
경희대학교
to6044, ssjj3552, yhm1620, ekhong@khu.ac.kr

Hands-On Analysis of O-RAN SC-Based Non-RT RIC and AI/ML Framework

Hoseong Choi, Sunggin Lee, Hyunmin Yoo, Eenkee Hong
Kyunghee University

요 약

본 논문에서는 O-RAN Software Community(O-RAN SC)의 “Creating and Running an ML-based rApp” 가이드를 바탕으로 Non-RT RIC의 Kafka 기반 데이터 파이프라인과 AI/ML Framework의 Kubeflow 기반 MLOps 파이프라인을 직접 실습하고 분석하였다. 특히 CT(통신 기술)에 IT·DT(정보·데이터 기술)의 융합으로 등장한 신규 기술 스택을 학계 연구자 관점에서 체계적으로 정리하였다. 또한, 오픈소스 구동 및 로그 분석 과정을 End-to-End Learning Path 형태로 제시해, 관련 경험이 부족한 연구자도 O-RAN 오픈소스 환경에 빠르게 입문할 수 있는 실용적 가이드를 제공한다.

Ⅰ. 서 론

최근 무선 통신 분야에서는 폐쇄형 RAN(Radio Access Network) 구조의 한계를 극복하기 위한 대안으로 Open RAN(O-RAN)이 주목받고 있다. O-RAN은 기존 통신 기술(Communication Technology, CT)에 정보기술(Information Technology, IT)과 데이터 기술(Data Technology, DT)을 결합한 ICDT 융합 구조를 지향하며, 개방성·클라우드화·지능화를 동시에 실현한다 [1]. O-RAN의 핵심 컴포넌트 중 하나인 Non-Real-Time RAN Intelligent Controller (Non-RT RIC)는 Apache Kafka를 활용해 실시간으로 데이터를 처리하고, Open Network Automation Platform (ONAP)의 Software-Defined Network Controller (SDNC)와 Policy Framework를 이용해 네트워크 정책을 관리한다. 또한, Spring Boot 기반의 마이크로서비스와 Kubernetes, Helm을 활용해 서비스를 운영하고 컨테이너를 관리한다. InfluxDB 및 MinIO를 통해 성능 지표와 비정형 데이터를 관리한다.

이와 더불어 MLOps를 구현하는 O-RAN SC의 AI/ML Framework는 Kubeflow를 사용해 모델을 학습시키고 KServe로 배포한다. Kubernetes를 통한 컨테이너 배포 및 운영 관리, Istio를 이용한 서비스 간 통신, 보안 등 다양한 IT 및 DT 기술을 통합적으로 활용한다.

그러나 O-RAN의 이러한 IT·DT 스택은 전통적 CT 환경에 익숙한 통신공학 연구자·개발자에게 높은 진입 장벽으로 작용한다. 따라서 본 논문은 O-RAN SC에서 공개한 “Creating and Running an ML-based rApp” [2]를 직접 구동하고 분석한 사례를 제공한다. 해당 가이드는 RANPM (RAN Performance Management)와 AI/ML Framework를 통합적으로 다룰 수 있도록 설계되어, O-RAN의 핵심 컴포넌트인 Non-RT RIC과 AI/ML Framework [3]의 연계를 PoC 수준으로 경험할 수 있는 환경을 제공한다. 제안한 실습·분석 결과는 O-RAN 연구자·개발자가 ICDT 융합 생태계에 빠르게 안착할 수 있는 실용 지침을 제공하며, 향후 O-RAN 기반 지능형 네트워크 연구·개발에 기여할 것으로 기대된다.

Ⅱ. 본론

1. RANPM 오픈소스 구성 및 실행 환경

RANPM[4]은 O-RAN SC에서 제공하는 Non-RT RIC용 RAN Performance Management (PM) 모듈로, 데이터 소비자(Data consumer)가 원하는 KPI를 구독·활용할 수 있도록 해주는 확장성 높은 End to End 데이터 파이프라인을 제공한다. 구성 요소는 다음과 같다.

A. Data File Collector

RAN으로부터 데이터 파일을 수집하는 역할을 수행하며, 해당 데이터는 3GPP 표준에 따라 정의된 XML 기반의 PM report 형식으로 제공된다. Java Spring Boot 프레임워크를 기반으로 구현되었으며, PM report 파일은 다양한 프로토콜을 통해 다운로드할 수 있도록 설계되어 있다.

B. File Converter

수집된 PM report는 XML 형식에서 JSON 형식으로 변환되며, 단순한 포맷 변경으로 내부 데이터 구조는 3GPP 표준을 그대로 유지한다.

C. PM Producer

각 Information Job (정보처리 작업, Info-Job)의 목적에 따라 PM 데이터를 필터링해 필요한 데이터만을 소비자에 선택적으로 제공하며, Spring Boot 프레임워크를 기반으로 구현되었다.

D. Information Coordinator Service (ICS)

데이터 생산자(Data Producer)와 데이터 소비자 사이의 결합을 완전히 분리하는 데이터 구독 서비스이다. 데이터 소비자가 특정 요소에 대한 정보를 알 필요 없이 원하는 데이터를 획득할 수 있도록 지원한다. 데이터 소비자는 자신이 필요로 하는 Information Type (정보 유형, Info-Type), 필터링 조건, 전송 주기 등의 파라미터를 지정해 Info-Job을 생성하고, 이를 통해 다양한 데이터 생산자로부터 데이터를 구독할 수 있다.

2. Kafka 기반 Non-RT RIC 데이터 파이프라인 분석

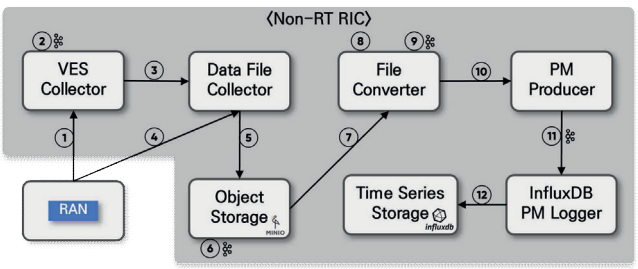

그림 1 RAN PM 데이터 처리 파이프라인의 구성 요소 및 처리 흐름

Non-RT RIC에서는 대용량 실시간 데이터 스트리밍을 위해 Apache Kafka를 메시지 브로커로 채택하고, 운영 편의성 및 모니터링을 위해 Redpanda UI를 함께 활용하였다. 특히 Redpanda UI는 31767 포트를 통해 접속할 수 있으며, 이를 이용해 Kafka 토픽별로 소비자 정보와 메시지 현황

을 실시간으로 확인할 수 있다. Kafka Topic 종류와 흐름은 다음과 같다.

Component	입력 Kafka 토픽	출력 Kafka 토픽
push_qoe_data.sh	-	File-Ready
Data File Collector	File-Ready	file-collected
File Converter	file-collected	json-file-ready-kpadp
PM Producer	json-file-ready-kpadp	pmreports
InfluxDB PM Logger	pmreports	-

표 1 Kafka Topic 목록

먼저 push_qoe_data.sh 스크립트가 RAN 노드에서 PM 리포트 파일을 내려받아 XML로 변환한 뒤 GZIP으로 압축하고, 새로운 파일의 준비 완료를 알리는 File Ready 토픽을 발행한다. 이 스크립트는 본래 그림 1의 1~3번인 VES 이벤트가 수행하던 'RAN Data 생성 → 알림' 구간을 대체하며, 이후 흐름은 모두 Kafka로 자동화된다. Data File Collector는 **File Ready** 토픽을 구독해 다운로드 URL을 받아 MinIO에 XML 원본을 저장하고, 완료 사실을 file-collected 토픽으로 알린다.

이어지는 File Converter 단계에서는 **file-collected** 토픽을 읽어 해당 XML을 JSON으로 변환한다. 그 결과를 MinIO에 .json.gz로 저장한다. 변환이 끝나면 json-file-ready-kpadp 토픽을 발행(동시에 생성되는 json-file-ready-kp는 이후 사용 안 됨)한다. 다음으로 PM Producer가 **json-file-ready-kpadp** 토픽을 구독해 JSON 파일명을 확인하고 MinIO에서 파일을 읽어 온다. PM producer는 ICS에 PmData Info-Type의 생산자로 등록되어 있고, 데이터 소비자가 지정한 필터 조건에 따라 데이터를 선별해 pmreports 토픽으로 전달한다.

마지막으로 데이터 소비자인 influxDB PM Logger는 **pmreports** 토픽을 받아 InfluxDB에 적재한다. 값이 문자열로 저장되므로 UI에서 시각화하려면 float형 변환이 필요하다. 요약하면, 네 개의 Kafka 토픽이 직렬로 연결되어 각 단계의 파일 포맷 변환, 저장 위치, 메시지 필터링을 담당함으로써 RAN PM 리포트의 실시간 가공·배포 체계를 완성한다.

Component	실제 Pod 이름	언어	저장소 버킷
push_qoe_data.sh	-	Shell	-
Data File Collector	dfc-0	Java	rofiles (MinIO)
File Converter	kafka-producer-pm-xml2json-0	Go	pm-files-json (MinIO)
PM Producer	pm-producer-json2kafka-0	Java	pm-files-json (MinIO, 읽기전용)
InfluxDB PM Logger	pmlog-0	Java	pm-log-bucket (Influx)

표 2 RAN PM 구성 요소 세부 정보

3. AI/ML Framework와의 연계 분석

O-RAN SC의 AI/ML Framework는 O-RAN Alliance WG2가 정의한 AI/ML 워크플로우를 실제로 구동하기 위해 구현된 참조 MLOps 시스템이며, 이에 대한 상세한 구조 및 동작 방식은 문헌 [5]에 기술되어 있다.

먼저, RAN PM 모듈과 연동할 때는 외부의 InfluxDB 31812번 NodePort를 통해 AIMLFW가 직접 데이터를 조회하는 방식과, 'enable_dme=true' 설정 후 RANPM의 DME(ICS) 31823번 NodePort를 경유해 RANPM이 대신 데이터 소스를 조회하는 방식의 두 가지 방법을 선택할 수 있다. 현재는 별도의 DME 연동 없이 전자의 방식을 사용하며, 해당 내용은 학습용 Feature Group 생성 시 결정할 수 있다.

현재 Training Manager (TM) 등의 요소가 코드 리팩토링 중이어서, 공식 문서나 가이드에 나와 있는 API 경로와 실제 요청 형식이 일부 상이할 수 있다. 현재는 그림 2처럼 TM Pod를 호출할 때는 포트 번호 뒤에 '/ai-ml-model-training/v1/'를 반드시 포함해야 한다. 또한, Model Management Service (MMS)가 비활성화된 상태(TM의 PostgreSQL을 모델 DB로 사용)에서도 요청 데이터에 modelId와 modelVersion을 함께 전달해야 하기 때문에, 가이드에 명시되지 않았지만 Training Job 생성 전에 모델을 등록해야하며 그림 3의 엔드포인트를 참고한다.

그림 2 Training Job 생성 API

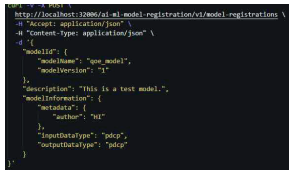

그림 3 Model 등록 API

이후 Non-RT RIC에서 rApp의 라이프사이클을 관리하는 rAppmanager에 학습된 model의 정보가 담긴 rApp package 파일(.csar)을 업로드한다. 이후 ONAP의 Automation Composition Management (ACM)과 Service Management Exposure (SME) 기능을 제공하는 KServe로 rApp instance를 생성해 rApp을 직접 배포하고 Kong을 이용해 이를 추론해볼 수 있다. rApp 배포 및 추론에 대한 구체적인 분석은 후속 연구로 진행될 예정이다.

Ⅲ. 결론

본 연구에서는 O-RAN SC의 실습 사례를 통해 CT 환경에 익숙한 연구자도 IT·DT가 융합된 ICDT 스택을 단계별로 체험할 수 있음을 확인하였다. 이러한 실습·분석 결과는 2018년 설립 이후 Open Fronthaul·E2 인터페이스 중심으로 개방성과 상호운용성 확보에 주력해 온 O-RAN Alliance의 표준화 방향과도 일치한다. 최근에는 AI/ML 기반 지능화가 무선망의 핵심 경쟁력으로 인정받아 AI/ML Framework 및 Non-RT RIC 표준화가 한층 강화되고 있으며, 실제로 O-RAN SC 내에서는 AI/ML Framework가 SMO 내부로 통합되어 Non-RT RIC과 긴밀히 연동되도록 하는 개발·표준화 작업이 활발히 진행 중이다. 네트워크 운영에서 AI/ML이 필수 역량으로 부상함에 따라 에너지 절감과 QoS 보장을 위해 트래픽 예측 및 정책 배포를 통합 관리하는 MLOps의 중요성이 더욱 강조되었으며, 본 연구가 제시한 MLOps 연계 방법은 향후 지능형 무선망 연구·개발 및 실제 네트워크 운영에서 해당 역량을 강화하는 데 실질적인 지침이 될 것으로 기대된다.

ACKNOWLEDGMENT

이 논문은 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원-대학ICT연구센터(ITRC)의 지원(IITP-2025-RS-2021-II212046, 50%)과 과학기술정보통신부 및 정보통신기획평가원의 오픈랜 인력양성 프로그램(연세대) 연구 결과로 수행되었음(IITP-2025-RS-2024-00434743, 50%)

참 고 문 헌

[1] Y. Huang et al., "Validation of Current O RAN Technologies and Insights on the Future Evolution," IEEE J. Sel. Areas Commun., vol.41, no.10, pp.3054 3070, Oct.2023.

[2] O-RAN Software Community, Creating and Running an ML-based rApp, Release K Wiki, accessed May 15, 2025.

[3] O-RAN SC, "aiml-fw-aimlfw-dep," GitHub, https://github.com/o-ran-sc/aiml-fw-aimlfw-dep, accessed May 1, 2025.

[4] O-RAN SC, "nonrtric-plt-ranpm," GitHub, https://github.com/o-ran-sc/nonrtric-plt-ranpm, accessed May 1, 2025.

[5] 최호성 외, "O-RAN 환경에서 MLOps를 통한 AIMLFW 운영 자동화," 한국통신학회 동계학술대회, 2025.