한국컴퓨터그래픽스학회
Korea Computer Graphics Society Vol. 32, No. 4, P. 13~22

전력 설비 감시 시스템의 웹 표준 전환을 위한 SPA 아키텍처 설계 및 구현

윤세진[1] 김준호[1] 조인호[1] 조용만[2] 김선정[1]*

[1]한림대학교 컴퓨터공학과 [2](주)인터포

[1]{M25051, M25050, M25052, sunkim}@hallym.ac.kr [2]interfo@daum.net

Design and Implementation of SPA Architecture for Web Standard Transition of Power Facility Monitoring Systems

Sejin Yun[1] Junho Kim[1] Inho Jo[1] Yongman Cho[2] Sun-Jeong Kim[1]*

[1]Dept. of Computer Engineering, Hallym University [2]INTERFO Co., Ltd

요 약

본 연구는 기존 클라이언트-서버 기반 SCADA 시스템의 노후화와 폐쇄성 문제를 해결하기 위해, Vue.js 기반의 고성능 웹 SCADA UI 모듈을 설계 및 구현하였다. 이를 위해 이벤트 기반의 단일 페이지 애플리케이션 아키텍처와 컴포넌트 기반 개발 방법론을 도입하여, 시스템의 유연성을 확보하고 유지보수 효율성을 극대화하였다. 웹 환경에서의 대용량 실시간 데이터 처리 시 발생하는 렌더링 병목 현상을 해결하고자 가상 스크롤 및 비동기 배치 처리 기법을 적용하였으며, 클라이언트 단의 보안 위협을 방지하기 위한 정규화 및 인증 로직을 내재화하였다. 실험 결과, 초기 로딩 속도 0.8초, 대용량 렌더링 속도 0.9초를 기록하여 목표 성능인 1.0초 이내를 달성하였으며, 보안 취약점 0건을 통해 시스템의 안전성을 검증하였다. 본 연구는 최적화된 웹 기술을 통해 산업용 시스템의 요구 성능을 충족하는 한국형 웹 관제 모델의 실현 가능성을 입증하였다는 데 의의가 있다.

Abstract

This study designs and implements a high-performance Web SCADA UI module based on Vue.js to address the issues of obsolescence and the closed nature of existing Client-Server (CS) based SCADA systems. To achieve this, an event-driven Single Page Application (SPA) architecture and Component-Based Development (CBD) methodology were adopted to ensure system flexibility and maximize maintenance efficiency. To resolve rendering bottlenecks occurring during large-scale real-time data processing in a web environment, virtual scrolling and asynchronous batch processing techniques were applied; additionally, input sanitization and authentication logic were internalized to prevent client-side security threats. Experimental results showed an initial loading speed of 0.8 seconds and a large-scale data rendering speed of 0.9 seconds, successfully achieving the target performance of under 1.0 second. Furthermore, the safety of the system was verified by confirming zero security vulnerabilities. This study is significant in that it demonstrates the feasibility of a Korean-style web control model that satisfies the rigorous performance requirements of industrial systems through optimized web technologies.

키워드: 스카다(SCADA), 웹 기반 시스템, 데이터 시각화, UI 모듈, 렌더링 최적화

Keywords: SCADA, Web-based System, Data Visualization, UI Module, Rendering Optimization

*corresponding author: Sun-Jeong Kim / Dept. of Computer Engineering, Hallym University (sunkim@hallym.ac.kr)

Received : 2026.02.09./ Review completed : 1st 2026.03.22. 2nd 2026.05.26./ Accepted : 2026.06.06.
DOI : 10.15701/kcgs.2026.32.4.13
ISSN : 1975-7883(Print)/2383-529X(Online)

1. 서론

국가 기간 산업 및 전력망 운영의 핵심인 SCADA (Supervisory Control And Data Acquisition) 시스템은 설비의 감시와 제어를 위해 높은 신뢰성과 실시간성을 요구한다[1, 2]. 전통적으로 이러한 시스템은 보안과 성능 보장을 위해 폐쇄적인 인트라넷 환경 하에서 클라이언트-서버 (CS: Client-Server) 아키텍처로 구축되어 왔다. 그러나 현재 국내 전력 관제 현장에서 운용 중인 다수의 시스템은 구축된 지 10년 이상 경과된 노후화된 상태로, 유지보수 측면에서 심각한 한계에 봉착해 있다[3]. 가장 큰 문제는 CS 기반 시스템이 ActiveX와 같은 비표준 기술이나 특정 운영체제(OS)에 종속된 네이티브 애플리케이션 형태로 개발되었다는 점이다. 이는 시스템의 유지보수 비용을 지속적으로 증가시킬 뿐만 아니라, 모바일 기기나 태블릿 등 다양한 단말 환경에서의 접근을 원천적으로 차단하여 운영의 유연성을 저해한다. 또한, 텍스트 위주의 경직된 사용자 인터페이스(UI)와 제한적인 상호작용성은 대형 관제 상황실에서 장시간 모니터링 업무를 수행하는 운영자에게 높은 인지 부하를 유발하여, 비상 상황 시 신속한 의사결정을 방해하는 요인으로 작용하고 있다(그림 1).

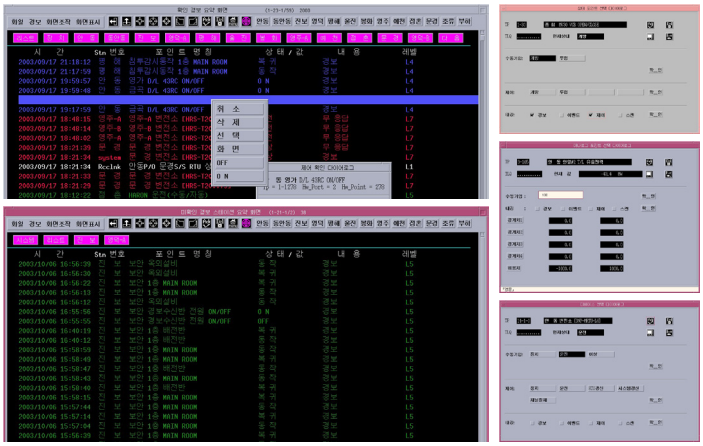


Figure 1: Example of a legacy foreign-made SCADA HMI.

이러한 문제를 해결하기 위해 최근 산업계에서는 HTML5, CSS3, JavaScript 등 웹 표준 기술을 적용한 웹 기반 SCADA 시스템으로의 전환(Digital Transformation)을 시도하고 있다[4, 5, 6]. 웹 기반 시스템은 별도의 클라이언트 설치가 필요 없고, OS에 독립적인 운영이 가능하다는 장점이 있다. 그러나 수천 개의 관제 포인트에서 밀려오는 대용량 시계열 데이터를 실시간으로 처리해야 하는 SCADA 시스템을 웹 환경으로 이식하는 과정에는 중대한 기술적 난제가 존재한다. 웹 브라우저의 렌더링 엔진은 DOM (Document Object Model)을 기반으로 동작하는데, 초당 수십~수백 건의 데이터가 갱신되는 고빈도 데이터 유입 환경에서는 잦은 DOM 조작(Manipulation)이 심각한 렌더링 병목 현상을 초래한다. 이는 화면의 프레임 드롭(Frame Drop)이나 멈춤 현상으로 이어져 실시간 관제 업무의 안정성을 훼손할 수 있다. 또한, 개방형 네트워크인 웹 환경의 특성상 CS 환경 대비 크로스 사이트 스크립팅(XSS) 등 보안 위협에 노출될 가능성이 높아, 이에 대한 구조적인 방어책 마련이 필수적이다[7].

본 연구는 상기한 기술적 제약을 극복하고, 국내 전력 운영 환경에 최적화된 고성능 웹 기반 SCADA UI 모듈을 설계 및 구현하는 것을 목적으로 한다. 본 연구에서는 기존 CS 시스템의 성능을 웹 환경에서 구현하기 위해 Vue.js 기반의 SPA(Single Page Application) 아키텍처를 도입하고, 렌더링 성능 최적화를 위한 알고리즘을 제안한다.

구체적으로 본 연구의 범위는 다음과 같다. 첫째, 사용자 권한 관리(RBAC), 시스템 운영, 데이터 시각화, 알람 모니터링 등 SCADA의 4대 핵심 기능을 컴포넌트 단위로 모듈화하여 재사용성을 확보한다. 둘째, 대용량 데이터 처리에 따른 렌더링 지연을 해소하기 위해 가상 스크롤(Virtual Scrolling) 기법과 requestAnimationFrame 기반의 비동기 배치 처리(Batch Processing) 알고리즘을 적용하고 그 유효성을 검증한다. 셋째, 프론트엔드 레벨에서의 보안성을 강화하기 위해 입력값 정규화(Sanitizer) 및 JWT 인증 인터셉터 설계를 내재화한다.

2. 시스템 설계 및 디자인 원칙

2.1 이벤트 기반의 CBD 아키텍처 설계

기존 클라이언트-서버 기반 SCADA 시스템은 화면을 갱신할 때마다 전체 리소스를 다시 호출하는 구조적 비효율성으로 인해, 데이터 변동성이 높은 전력 관제 환경에서 불필요한 트래픽과 렌더링 지연을 유발하였다. 이를 근본적으로 해결하기 위해 본 연구에서는 이벤트 기반의 단일 페이지 애플리케이션(SPA) 아키텍처를 설계하였다[8, 9]. 이는 시스템 초기 구동 시 실행에 필요한 정적 자산과 애플리케이션 셸을 미리 로드하고, 실제 운영 중에는 웹소켓을 통해 수신되는 경량화된 데이터 스트림만을 비동기로 처리하여 렌더링 부하를 최소화하는 구조이다. 또한, 시스템의 재사용성과 유지보수성을 엔터프라이즈 수준으로 확보하기 위해 컴포넌트 기반 개발(CBD) 방법론을 적용하였다[10](그림 2). 복잡하고 다양한 형태의 관제 화면을 최소 단위의 기능 요소로 분해하여 버튼, 그리드, 차트 등을 독립적인 모듈로 정의하고, 이를 상향식으로 조립하여 대시보드를 구성하는 설계를 채택하였다. 이러한 구조는 특정 설비의 관제 로직이나 디자인이 변경되더라도 전체 시스템을 재컴파일할 필요 없이 해당 컴포넌트만 독립적으로 수정 및 배포가 가능하게 하여 유지보수의 유연성을 보장한다.

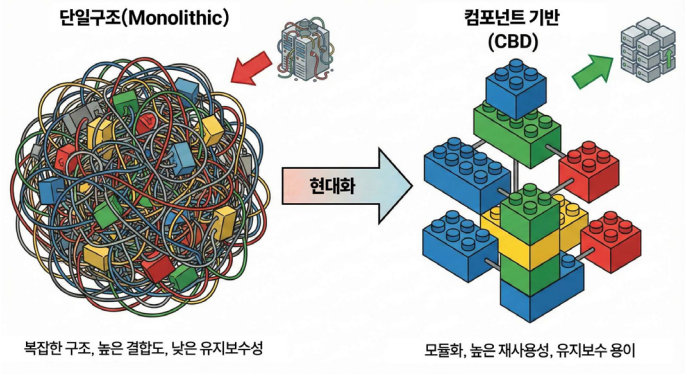


Figure 2: Application of the Component-Based Development (CBD) methodology. Comparison between Monolithic and Component-based architectures. Maintainability is maximized through modularization.

2.2 이기종 프로토콜 추상화를 위한 어댑터 패턴 적용

SCADA 시스템은 다양한 제조사의 원격 단말 장치(RTU) 및 센서로부터 데이터를 수집하기 때문에, 백엔드 데이터 모델이 파편화되는 문제가 빈번하게 발생한다. 본 연구에서는 이러한 이기종 데이터 구조가 프론트엔드 UI 계층에 직접적인 의존성을 갖지 않도록 서비스 계층에 어댑터 패턴을 적용하여 데이터 통신을 추상화하였다.

구체적으로 텔레메트리 서비스 모듈 내에 데이터 변환 메서드를 구현하여, 서버로부터 수신된 원본 데이터를 UI 렌더링에 최적화된 표준 객체 모델로 변환하는 전처리 과정을 중앙화하였다. 예를 들어, 서버에서 전송하는 정수형 상태 코드를 UI 상태 관리를 위한 불변 객체인 열거형으로 매핑하고, 비정형 메타데이터를 정규화된 태그 구조로 재가공한다. 이러한 설계는 향후 백엔드 API 규격이 변경되거나 시뮬레이션 환경과 실제 운영 환경이 교차하는 상황에서도 비즈니스 로직의 수정 없이 어댑터의 매핑 규칙만 변경함으로써 시스템의 안정성을 유지할 수 있게 한다.

2.3 대용량 데이터 동기화를 위한 중앙 집중식 상태 관리

수천 개의 관제 포인트가 실시간으로 갱신되는 환경에서, 개별 컴포넌트 간에 데이터를 직접 전달하는 방식은 상태 불일치 오류와 렌더링 성능 저하를 야기한다. 이를 방지하기 위해 본 연구에서는 전역 상태 관리 패턴을 도입하여 데이터 흐름을 단방향으로 통제하였다. 특히 초당 수백 건 이상의 고빈도 데이터가 유입되는 상황에서 데이터 검색 및 갱신 속도를 보장하기 위해, 상태 저장소의 자료구조를 일반적인 배열이 아닌 해시 맵 구조로 설계하였다.

이를 통해 특정 설비의 상태 조회 및 갱신 알고리즘의 시간 복잡도를 획기적으로 단축시켰으며, 객체의 불필요한 재생성을 억제하여 메모리 관리 효율을 높이고 가비지 컬렉션에 의한 프레임 드롭 현상을 구조적으로 방지하였다. 수신된 알람 데이터는 즉시 저장소에 반영되며, 반응성 시스템을 통해 이를 구독하고 있는 대시보드, 알람 리스트, GIS 맵 등 다수의 컴포넌트가 동시에 동기화되도록 설계하였다.

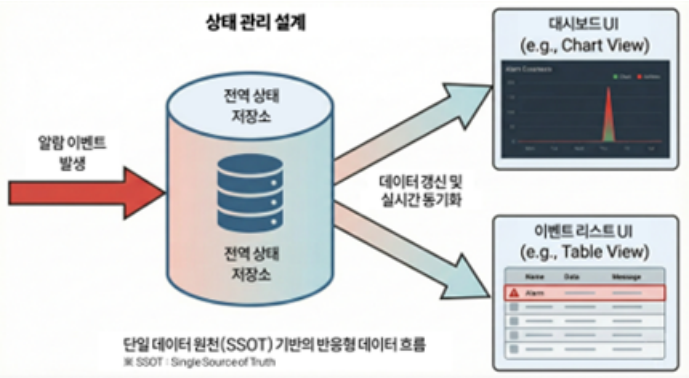


Figure 3: Real-time data synchronization flow between multiple modules via global state management. By adopting a global state management pattern, an organic data flow was designed to ensure that the dashboard and event list react simultaneously when an alarm occurs.

2.4 인지 공학 기반의 상황실 UX/UI 최적화

단순한 심미적 디자인을 넘어, 24시간 무중단 운영되는 관제 상황실의 특수성을 반영하여 운영자의 인지 부하를 최소화하는 인지 공학적 가이드라인을 수립하고 시스템에 반영하였다[11].

첫째, 시각적 계층 구조의 최적화를 위해 다크 모드 기반의 고대비 색상 시스템을 적용하였다. 배경과 정보 영역의 명도 대비를 극대화하여 장시간 주시 시 눈의 피로도를 저감시키고, 알람 등급에 따른 색상 코딩을 고정하여 비상 상황 인지 속도를 향상시켰다.

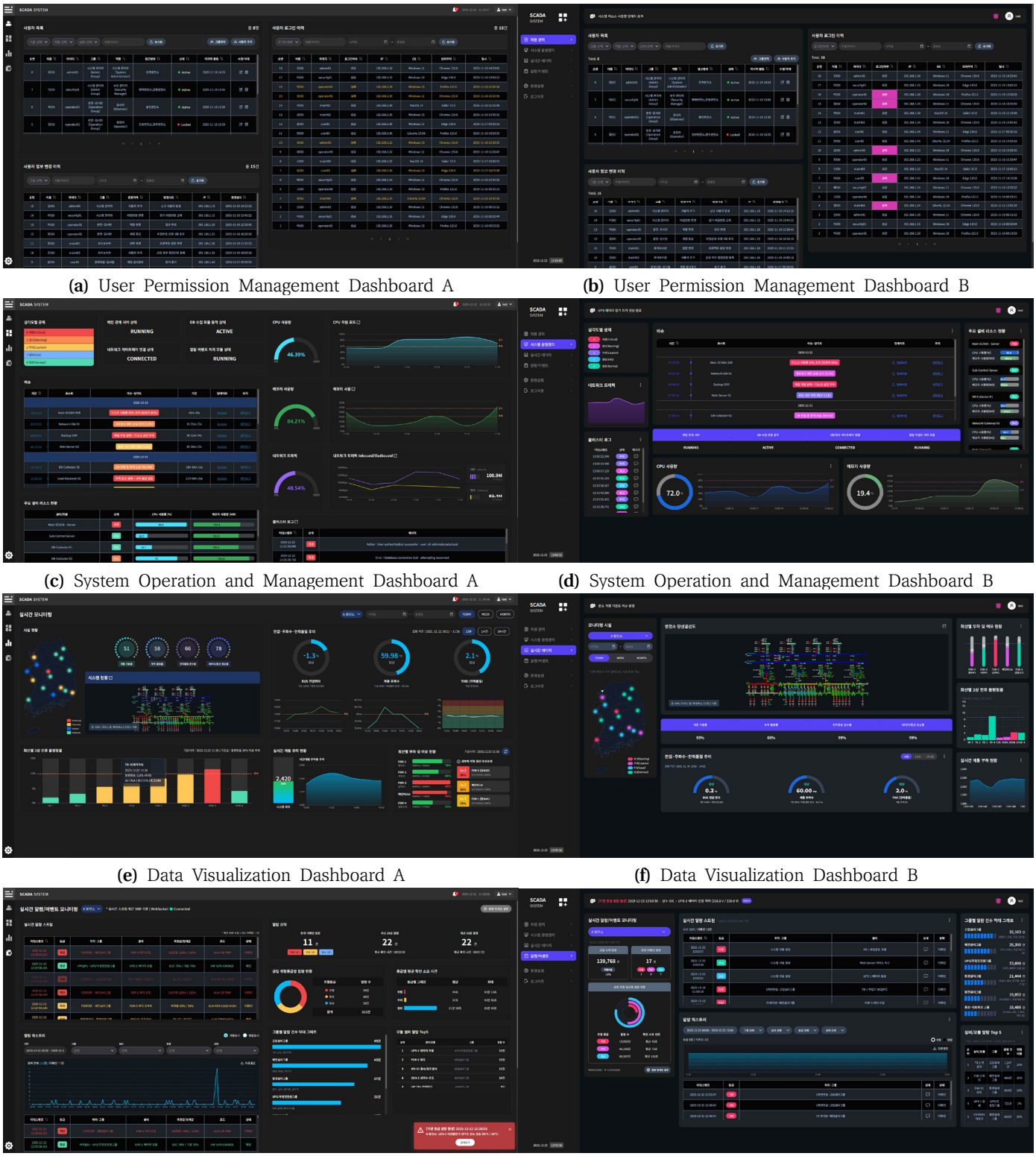
둘째, 대형 비디오 월 환경에서의 가독성을 보장하기 위해 유동적인 반응형 레이아웃 대신 고정 픽셀 기반의 그리드 시스템을 채택하였다. 이는 해상도 변화에 따른 정보 위치의 이동을 원천 차단하여, 운영자가 특정 설비 정보의 위치를 공간적으로 기억하고 신속하게 접근할 수 있도록 지원한다.

셋째, 정보 접근 경로를 최적화하기 위해 '3-Click Rule'을 엄격히 적용하여, 계층화된 설비 구조 속에서도 최하위 단말 장치의 상세 정보까지 3단계 이내의 조작으로 도달할 수 있는 내비게이션 구조를 설계하였다.

3. 고성능 UI 모듈 구현

3.1 시스템 개발 환경 및 아키텍처

본 연구에서는 기존 SCADA 시스템의 폐쇄성을 극복하고 웹 표준 환경에서의 유연한 운영을 보장하기 위해, HTML5 표준 기술과 Vue.js 프레임워크를 기반으로 시스템을 구축



(a) User Permission Management Dashboard A

(b) User Permission Management Dashboard B

(c) System Operation and Management Dashboard A

(d) System Operation and Management Dashboard B

(e) Data Visualization Dashboard A

(f) Data Visualization Dashboard B

(g) Alarm and Event Monitoring Dashboard A

(h) Alarm and Event Monitoring Dashboard B

Figure 4: Development of four types of web-based SCADA UI modules specialized for power facility monitoring systems. UI modules for core functions were developed to maximize User Experience (UX), including (a-b) User Permission Management, (c-d) System Operation/Management, (e-f) Data Visualization, and (g-h) Alarm/Event Monitoring. Two types of UI templates for each module (totaling eight types) were designed and published to respond flexibly to field conditions.

하였다[9]. 화면 전체를 새로 고침하지 않고 필요한 데이터만 비동기로 갱신하는 단일 페이지 애플리케이션 구조를 채택하여, 실시간 감시 업무에 필수적인 빠른 반응 속도를 확보하였다.

시스템 설계에는 컴포넌트 기반 개발 방법론을 적용하였다. 이는 복잡한 관제 시스템의 기능을 독립적인 단위로 모듈화하여 개발하는 방식으로, 향후 설비 확장이 필요하거나 유지보수 이슈가 발생했을 때 해당 모듈만을 수정하거나 교체할 수 있어 운영 효율성이 매우 높다. 본 시스템은 크게 사용자 접근을 제어하는 권한 관리, 전반적인 시스템 상태를 확인하는 운영 관리, 설비 데이터를 그래픽으로 표현하는 시각화, 그리고 장애 상황을 통보하는 알람 감시 등 4가지 핵심 모듈로 구성된다(그림 4).

본 시스템 개발에 사용된 구체적인 기술 스택 및 라이브러리 사양은 <표 1>과 같다. 특히 최신 웹 표준 기술인 Vue.js v3.5.18과 상태 관리 라이브러리인 Pinia v3.0.3을 적용하여 시스템의 유연성과 유지보수 효율성을 확보하였으며, 실시간 텔레메트리 데이터 시각화를 위해 Chart.js와 ECharts를 혼합 사용하였다.

Table 1. System Development Environment

구분	내용 및 사양
운영체제(OS)	Windows 11 Pro
개발 프레임워크	Vue.js v.3.5.18
사용 언어	JavaScript (ES6+), HTML5, CSS3
상태 관리 도구	Pinia v.3.0.3
실시간 통신	WebSocket (Native API)
시각화 라이브러리	Chart.js v4.5.1, ECharts v5.5.0
개발 도구 (IDE)	Visual Studio Code v1.103

3.2 핵심 기능 모듈 구현

각 기능 모듈은 전력 관제 환경의 특수성을 반영하여 다음과 같이 구현되었다.

첫째, 사용자 권한 관리 모듈은 역할 기반 접근 제어 개념을 적용하여 설계되었다. 관리자, 운영자, 일반 사용자 등 업무 특성에 따라 메뉴 접근 권한과 제어 권한을 차등 부여함으로써, 비인가 사용자에 의한 오조작이나 중요 데이터의 유출을 시스템적으로 방지하도록 구현하였다.

둘째, 데이터 시각화 모듈은 수치로 된 텔레메트리 데이터를 운영자가 직관적으로 이해할 수 있도록 다양한 차트와 그리드 형태로 변환하여 제공한다. 특히 실시간으로 변화하는 전압, 전류 등의 데이터를 시각화하기 위해 웹 표준 캔버스 기술을 활용한 차트 라이브러리를 적용하였으며, 이를 통해 대량의 데이터 포인트가 갱신되더라도 부드러운 애니메이션 효과를 제공하도록 하였다.

셋째, 알람 모니터링 모듈은 현장의 이상 징후를 즉각적으로 전달하는 핵심 기능이다[12]. 서버로부터 수신된 알람 메시지를 리스트 형태로 출력하며, 알람의 중요도에 따라 색상을 구분하여 표시함으로써 운영자가 위급 상황을 우선적으로 인지할 수 있도록 구현하였다. 또한, 발생한 알람에 대한 이력 조회 기능을 제공하여 사후 분석이 가능하도록 하였다.

3.3 실시간 데이터 처리 및 상태 관리

SCADA 시스템의 가장 중요한 요구사항인 데이터의 실시간성을 보장하기 위해 웹소켓 통신 기술을 적용하였다[13]. 이는 클라이언트가 서버에 주기적으로 데이터를 요청하는 방식이 아니라, 서버가 데이터 변경 시점에 즉시 클라이언트로 정보를 전송하는 방식이므로 불필요한 네트워크 트래픽을 줄이고 데이터 지연 시간을 최소화할 수 있다. 수신된 다량의 데이터가 여러 화면에 일관성 있게 표시되도록 하기 위해 전역 상태 관리 기법을 도입하였다. 설비의 상태 정보나 알람 내역과 같은 핵심 데이터는 중앙 저장소에서 통합 관리되며, 데이터가 갱신되면 이를 참조하고 있는 모든 화면 요소가 자동으로 동기화된다(그림 3). 이를 통해 개별 화면 간의 데이터 불일치 문제를 원천적으로 해결하고 시스템의 신뢰성을 높였다.

3.4 렌더링 최적화 기술 적용

대규모 전력 설비에서 발생하는 대용량 데이터를 웹 브라우저 상에서 지연 없이 처리하기 위해 보고서에 명시된 렌더링 최적화 기술을 적용하였다.

우선 가상 스크롤 기법을 도입하여 대용량 리스트 처리 성능을 개선하였다[14]. 수많은 알람 이력이나 로그 데이터를 화면에 표시할 때, 전체 데이터를 한 번에 그리는 대신 현재 사용자가 보고 있는 화면 영역에 해당하는 데이터만 선별적으로 렌더링하는 방식이다. 이를 통해 브라우저가 처리해야 할 그래픽 요소를 최소화함으로써, 데이터 양이 늘어나더라도 화면 스크롤이 끊기지 않고 부드럽게 동작하도록 구현하였다. 또한, 대량의 데이터가 일시에 유입될 때 발생할 수 있는 화면 멈춤 현상을 방지하기 위해 비동기 배치 처리 방식을 적용하였다. 수신된 데이터를 즉시 화면에 반영하지 않고 일정 단위로 묶어서 순차적으로 처리함으로써, 브라우저의 연산 부하를 분산시키고 안정적인 프레임 속도를 유지하도록 최적화하였다.

Algorithm 1 Frame-synced Asynchronous Batch Rendering

```
procedure UpdateTelemetryBatch(packetList)
    if isProcessing = True then                          ▷ 중복 실행 방지
        return
    end if
    isProcessing ← True                                  ▷ 프로세스 잠금
    ScheduleTask(via requestAnimationFrame)              ▷ 주사율 동기화 예약
     Asynchronous Update Block:
    for each packet in packetList do
        Update GlobalStateMap using packet.deviceID
    end for
    isProcessing ← False                                 ▷ 프로세스 잠금 해제
end procedure
```

제안하는 비동기 배치 처리 알고리즘의 세부 절차는 [Algorithm 1]과 같다. 본 알고리즘은 웹소켓을 통해 수집된 데이터 패킷을 즉시 렌더링하지 않고, 시스템 상태 플래그를 통해 경합을 제어한다. 이후 브라우저의 차기 렌더링 주기에 맞추어 데이터를 일괄 병합함으로써 메인 스레드의 점유율을 최적화하고 가비지 컬렉션 부하를 최소화하였다.

3.5 클라이언트 보안 기능 구현

개방형 웹 환경에서의 보안 취약점을 보완하기 위해 클라이언트 영역에서의 보안 기능을 강화하였다. 사용자 입력값에 대한 검증 로직을 적용하여 악성 스크립트가 포함된 데이터가 시스템에 유입되지 않도록 차단하는 입력값 정규화(Sanitizer) 기능을 구현하였다[7]. 또한, 시스템 접속 시 발급되는 인증 토큰을 활용하여 모든 데이터 요청 시 정당한 사용자인지를 검증하는 인증 인터셉터 구조를 적용하였다. 이를 통해 허가받지 않은 외부의 접근 시도를 차단하고, 웹 애플리케이션의 보안 무결성을 확보하였다.

4. 성능 평가 및 검증

4.1 실험 환경 및 방법

제안된 시스템이 산업 현장의 대용량 데이터 처리 환경에서도 목표 성능을 만족하는지 검증하기 위해, 실제 SCADA 데이터 통신 규격과 유사한 구조의 가상 데이터 생성기(Mock Data Generator)를 개발하여 테스트를 수행하였다. 파이썬(Python) 스크립트를 통해 전압, 전류, 전력량 등 주요 전력 설비의 시계열 데이터를 포함한 1,000건 이상의 레코드를 생성하였으며, 이를 기반으로 초당 다수의 이벤트가 유입되는 부하 상황을 시뮬레이션하였다.

성능 평가의 객관성과 재현성을 보장하기 위해 구축된 실험 환경의 하드웨어 및 소프트웨어 세부 사양은 <표 2>와 같다. 고성능 연산이 가능한 Intel Core i7-13700K 및 32GB RAM 환경에서 Google Lighthouse v100.0.0.4와 Chrome DevTools를 활용하여 시스템의 응답 속도와 렌더링 성능을 정밀 측정하였으며, OWASP ZAP v2.17.0을 통해 보안 무결성을 검증하였다.

Table 2. Experimental Environment

구분	상세 사양
CPU	Intel Core i7-13700K (3.4GHz, 16-Core)
메모리(RAM)	32GB (Samsung DDR5-5600 16GB × 2)
그래픽 (GPU)	NVIDIA GeForce RTX 4070 (12GB)
웹 브라우저	Google Chrome v141.0.7390.122
성능 측정 도구	Google Lighthouse v.100.0.0.4, Chrome DevTools
보안 진단 도구	OWASP ZAP v.2.17.0
테스트 데이터	1,000건 이상의 전력 설비 시계열 가상 데이터 (Mock Data)

성능 측정의 객관성을 확보하기 위해 Google Lighthouse와 Chrome DevTools의 Performance 탭을 측정 도구로 활용하였으며, 보안성 검증을 위해 국제 웹 보안 표준 기구의 자동 진단 도구인 OWASP ZAP을 사용하였다[14, 7].

4.2 정량적 성능 측정 결과

시스템의 응답 속도와 렌더링 성능에 대한 측정 결과는 다음과 같다.

첫째, 초기 로딩 속도는 웹 애플리케이션이 실행된 후 사용자가 실제 콘텐츠를 볼 수 있게 되는 시점인 LCP(Largest Contentful Paint)를 기준으로 측정하였다. 5회 반복 측정 결과 평균 0.8초를 기록하였으며, 이는 당초 목표치인 1.0초를 20% 상회하는 성능이다(그림 5). 이는 SPA 아키텍처의 특성을 살려 초기 구동에 필요한 필수 리소스만을 효율적으로 로드하도록 최적화한 결과로 분석된다.

둘째, 화면 전환 속도는 운영자가 메뉴를 클릭하여 다른 관제 화면으로 이동할 때 소요되는 시간을 측정하였다. 측정 결과 평균 0.46초를 기록하여 목표치인 0.5초 이내를 충족하였다(그림 6). 이는 컴포넌트 재사용성을 높이고 불필요한 DOM 재생성을 억제한 설계가 화면 간 이동 시 발생하는 브라우저의 연산 부하를 효과적으로 감소시켰음을 시사한다.

셋째, 데이터 시각화 렌더링 성능은 1,000건 이상의 관제 데이터를 차트와 그리드에 동시에 표출할 때 소요되는 렌더링 시간을 측정하였다. 부하 테스트 결과 0.9초의 렌더링 시간을 기록하여 목표치인 1.0초 이내를 달성하였다. 특히 가상 스크롤과 비동기 데이터 처리 기술을 적용함으로써, 데이터 양이 증가함에 따라 렌더링 시간이 선형적으로 급증하는 문제를 해결하고 안정적인 성능을 유지함을 확인하였다.

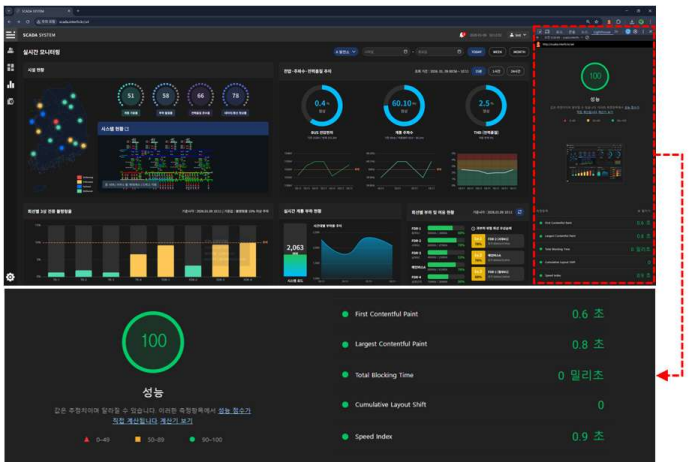


Figure 5: Measurement results of initial UI loading speed. Google Lighthouse was utilized for objective performance verification, focusing on Largest Contentful Paint (LCP) as a key indicator of perceived loading speed. The test measured the time required for the first text or image content to appear on the screen. The result recorded 0.8 seconds, achieving a 20% improvement over the initial target of 1.0 second.

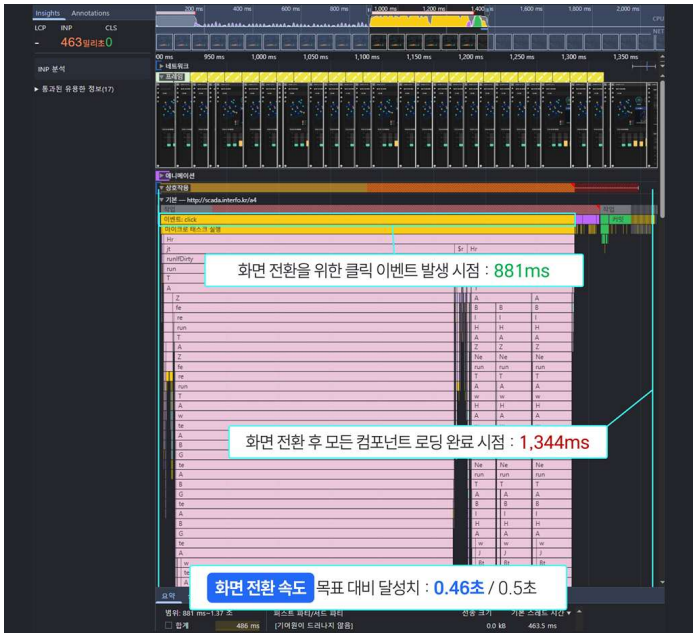


Figure 6: Measurement results of screen transition speed. The timeline was tracked from the moment the click event triggered the transition until the completion of all component loading. The result recorded 0.46 seconds, demonstrating immediate responsiveness without latency. User experience was enhanced by optimizing the interval between the event trigger (881 ms) and rendering completion (1,344 ms).

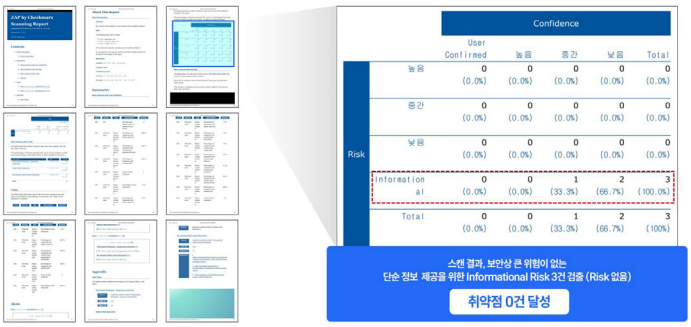


Figure 7: Inspection results of UI security vulnerabilities. A security scan was performed on the entire UI module using OWASP ZAP, an automated diagnostic tool from the international web security standard organization. The results showed zero actual vulnerabilities in the High, Medium, and Low risk categories (three detected 'Informational' items posed no security threat). This validates the web security integrity and safety essential for the deployment of industrial systems.

4.3 보안성 검증 결과

개방형 웹 환경에서의 보안 안전성을 검증하기 위해 OWASP ZAP을 활용하여 SQL 인젝션(Injection), 크로스 사이트 스크립팅(XSS) 등 주요 웹 취약점에 대한 정밀 스캔을 수행하였다[7]. 검사 결과, High, Medium, Low 등급의 유효 취약점은 단 한 건도 발견되지 않았으며(0건), 단순 정보 제공 수준의 알림(Informational) 3건만이 검출되었다(그림 7). 이는 클라이언트 사이드에 내재화된 입력값 정규화 로직과 보안 인터셉터 설계가 실질적인 방어 기제로 작동하고 있음을 입증한다.

5. 논의

5.1 웹 기술 기반 산업용 시스템의 실현 가능성

본 연구의 핵심 과제는 대용량 실시간 데이터를 처리해야 하는 SCADA 시스템을 웹 표준 기술로 전환함에 있어, 기존 CS(Client-Server) 기반 시스템 대비 성능적 열위를 어떻게 극복하느냐에 있었다. 실험 결과, 초기 로딩 속도 0.8초, 대용량 데이터 렌더링 속도 0.9초를 기록하여 산업 현장에서 요구하는 목표 성능(1.0초 이내)을 안정적으로 달성하였다.

이는 웹 브라우저의 DOM 조작 비용이 높다는 통념과 달리, 가상 스크롤이나 비동기 배치 처리와 같은 적절한 최적화 알고리즘이 적용된다면 웹 환경에서도 고빈도 시계열 데이터를 지연 없이 처리할 수 있음을 실증적으로 보여준다. 특히 하드웨어 자원을 직접 제어하는 네이티브 애플리케이션 없이도, 브라우저의 연산 자원만으로 60fps 수준의 부드러운 스크롤과 실시간 시각화가 가능하다는 점은 향후 전력 관제 시스템의 플랫폼이 웹 중심으로 이동할 수 있는 기술적 토대가 마련되었음을 시사한다.

5.2 개방형 네트워크 환경에서의 보안 무결성

폐쇄망에서 운용되던 SCADA 시스템이 웹 환경으로 전환될 때 가장 우려되는 점은 보안 취약성이다. 본 연구에서는 이를 보완하기 위해 별도의 보안 장비에만 의존하지 않고, 애플리케이션 설계 단계에서부터 보안 코드를 내재화하는 전략을 취했다.

OWASP ZAP 점검 결과 유효 취약점 '0건'을 달성한 것은, 클라이언트 레벨에서의 입력값 정규화와 인터셉터 기반의 인증 토큰 관리가 실질적인 방어 기제로 작동했음을 의미한다. 이는 웹 기반 SCADA 시스템이 개방형 네트워크의 이점을 취하면서도, 설계의 견고함에 따라 산업용 시스템이 요구하는 엄격한 보안 무결성을 충족할 수 있음을 확인시켜 주었다. 다만, 본 검증은 알려진 웹 취약점 패턴에 대한 방어 능력을 확인한 것으로, 지능형 지속 위협(APT)과 같은 고도화된 네트워크 공격에 대해서는 추가적인 인프라 보안 대책과의 연계가 필요할 것으로 판단된다.

5.3 연구의 한계

본 연구를 통해 고성능 웹 SCADA UI 모듈을 구현하였으나, 다음과 같은 몇 가지 한계점이 존재한다.

첫째, 본 시스템은 대형 관제 상황실의 비디오 월 환경에 최적화된 '고정형 그리드' 레이아웃을 채택하였다. 이는 운영자에게 일관된 정보 위치를 제공하는 데에는 유리하나, 현장 점검자가 사용하는 모바일이나 태블릿 등 다양한 해상도의 디바이스에 대응하기에는 유연성이 부족하다. 따라서 향후 다양한 단말 환경을 지원하기 위한 반응형 웹(Responsive Web) 또는 프로그레시브 웹 앱(PWA) 기술로의

확장이 요구된다.

둘째, 현재 구현된 시스템은 실시간 데이터의 '감시'와 '제어'에 초점이 맞춰져 있다. 축적된 이력 데이터를 분석하여 설비의 고장을 사전에 예측하거나, 이상 징후를 조기에 탐지하는 지능형 분석 기능은 포함되어 있지 않다. 향후 연구에서는 본 시스템에서 수집된 데이터를 AI 모델과 연동하여, 단순 모니터링을 넘어선 예지보전 시각화 플랫폼으로 고도화하는 연구가 수행되어야 할 것이다.

6. 결론

국가 전력망을 포함한 핵심 기반 시설의 감시 제어 시스템(SCADA)은 안정성을 최우선으로 하는 보수적인 운영 환경 탓에, IT 기술의 발전 속도에 비해 시스템의 현대화가 지체되어 왔다. 특히 기존 CS 기반 시스템의 폐쇄성과 비표준 기술 의존도는 유지보수의 비효율성을 초래하고, 다양한 관제 요구사항을 수용하는 데 걸림돌로 작용해 왔다.

본 연구에서는 이러한 문제를 해결하기 위해 HTML5 웹 표준 기술과 Vue.js 기반의 SPA 아키텍처를 도입하여, 국내 전력 운영 환경에 최적화된 고성능 웹 SCADA UI 모듈을 설계하고 구현하였다. 연구의 핵심은 웹 브라우저가 갖는 렌더링 성능의 한계를 극복하는 데 있었다. 이를 위해 가상 스크롤을 통한 DOM 가상화, 비동기 배치 처리, 그리고 해시 맵 기반의 상태 관리 최적화 등 공학적 기법을 적용하여 대용량 실시간 데이터를 지연 없이 시각화하는 프론트엔드 엔진을 완성하였다.

성능 평가 결과, UI 초기 로딩 속도 0.8초, 화면 전환 속도 0.46초, 대용량 데이터 렌더링 속도 0.9초를 기록하여 산업용 시스템이 요구하는 1.0초 이내의 응답 속도를 안정적으로 달성하였다. 또한, OWASP ZAP을 이용한 보안성 검증에서도 유효 취약점 0건을 기록하여, 설계 단계에서 내재화된 클라이언트 보안 로직의 유효성을 입증하였다.

본 연구의 학술적 의의는 범용 웹 기술만으로도 수천 개의 태그가 실시간으로 갱신되는 산업용 SCADA 시스템의 요구 성능을 충족할 수 있음을 정량적으로 규명한 데 있다. 산업적으로는 외산 소프트웨어 의존도가 높은 국내 전력 제어 시장에서, 한국형 관제 프로세스와 UX를 반영한 국산 UI 표준 모델을 확보함으로써 기술 자립의 기반을 마련했다는 점에서 기여하는 바가 크다.

향후 연구로는 본 과제에서 확립한 프론트엔드 아키텍처를 기반으로, 현장 점검자가 태블릿 등 모바일 기기에서도 설비 상태를 확인할 수 있도록 반응형 웹(Responsive Web) 또는 PWA(Progressive Web App) 형태로 시스템을 경량화하여 확장할 계획이다. 아울러, 단순 감시를 넘어 설비의 고장을 사전에 예측할 수 있도록 AI 기반의 예지보전 알고리즘과 연동된 지능형 시각화 모듈로 고도화하는 후속 연구를 진행하고자 한다.

감사의 글

본 과제(결과물)는 2025년도 교육부 및 강원특별자치도의 재원으로 강원RISE센터의 지원을 받아 수행된 지역혁신중심 대학지원체계(RISE)의 결과입니다.(2025-RISE-10-009)

References

[1] V. C. Gungor et al., "Smart Grid Technologies: Communication Technologies and Standards," in IEEE Transactions on Industrial Informatics, vol. 7, no. 4, pp. 529-539, Nov. 2011, doi: 10.1109/TII.2011.2166794.

[2] M. Sverko, T. G. Grbac and M. Mikuc, "SCADA Systems With Focus on Continuous Manufacturing and Steel Industry: A Survey on Architectures, Standards, Challenges and Industry 5.0," in IEEE Access, vol. 10, pp. 109395-109430, 2022, doi: 10.1109/ACCESS.2022.3211288.

[3] 한종선 (2025). SCADA SYSTEM 해외 기술동향 및 표준화 설명. The Proceedings of the Korean Institute of Illuminating and Electrical Installation Engineers, 39(3), 19-23.

[4] Kim, J., Moon, H., Kim, S., Choi, S., & Paek, W. (2025-07-16). Web-based HMI Development in MVDC distribution network operation system. 대한전기학회 학술대회 논문집, 부산.

[5] Abbas, H. A. (2015). Efficient Web-Based SCADA System. arXiv preprint arXiv:1501.05725.

[6] Chi, C. P., Hai, T. T., Huy, L. T., Le Ba, L., Le, N., & Trong, N. H. (2024). Platform software for building web scada applications. Creative Approaches Towards Development of Computing and Multidisciplinary IT Solutions for Society, 529-548.

[7] A. Sajid, H. Abbas and K. Saleem, "Cloud-Assisted IoT-Based SCADA Systems Security: A Review of the State of the Art and Future Challenges," in IEEE Access, vol. 4, pp. 1375-1384, 2016, doi: 10.1109/ACCESS.2016.2549047.

[8] R. N. V. Diniz-Junior et al., "Evaluating the performance of web rendering technologies based on JavaScript: Angular, React, and Vue," 2022 XVLIII Latin American Computer Conference (CLEI), Armenia, Colombia, 2022, pp. 1-9, doi: 10.1109/CLEI56649.2022.9959901.

[9] Ollila, R. ., Mäkitalo, N. ., & Mikkonen, T. . (2022). Modern Web Frameworks: A Comparison of Rendering Performance. Journal of Web Engineering, 21(03), 789-814. https://doi.org/10.13052/jwe1540-9589.21311

[10] Land, R., Sundmark, D., Lüders, F., Krasteva, I., Causevic, A. (2009). Reuse with Software Components - A

Survey of Industrial State of Practice. In: Edwards, S.H., Kulczycki, G. (eds) Formal Foundations of Reuse and Domain Engineering. ICSR 2009. Lecture Notes in Computer Science, vol 5791. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-04211-9_15

[11] B. Shneiderman, "The eyes have it: a task by data type taxonomy for information visualizations," Proceedings 1996 IEEE Symposium on Visual Languages, Boulder, CO, USA, 1996, pp. 336-343, doi: 10.1109/VL.1996.545307.

[12] L. Depetris et al., "Web-Based SCADA System to Control and Monitor Electrical Grids' Distribution Transformers and Medium Voltage Reclosers, Based on IoT," 2023 XX Workshop on Information Processing and Control (RPIC), Oberá, Argentina, 2023, pp. 1-6, doi: 10.1109/RPIC59053. 2023.10530667.

[13] V. Pimentel and B. G. Nickerson, "Communicating and Displaying Real-Time Data with WebSocket," in IEEE Internet Computing, vol. 16, no. 4, pp. 45-53, July-Aug. 2012, doi: 10.1109/MIC.2012.64.

[14] Stefan Huber and Lukas Demetz. 2019. Performance Analysis of Mobile Cross-platform Development Approaches based on Typical UI Interactions. In Proceedings of the 14th International Conference on Software Technologies (ICSOFT 2019). SCITEPRESS - Science and Technology Publications, Lda, Setubal, PRT, 40-48. https://doi.org/10.5220/0007838000400048

〈저자소개〉

윤세진
- 2025년 한림대학교 콘텐츠IT 학사
- 2025년~현재 한림대학교 컴퓨터공학 석사
- 관심분야: 증강현실, 가상현실, HCI, UI/UX
- https://orcid.org/0009-0001-2914-5217

김준호
- 2025년 한림대학교 콘텐츠IT 학사
- 2025년~현재 한림대학교 컴퓨터공학 석사
- 관심분야: 컴퓨터그래픽스, 가상/증강현실, NPR
- https://orcid.org/0009-0009-3799-9691

조인호
- 2025년 한림대학교 콘텐츠IT 학사
- 2025년~현재 한림대학교 컴퓨터공학 석사
- 관심분야: 가상현실, 인공지능
- https://orcid.org/0009-0006-2443-3961

조용만
- 2002년 강원대학교(강릉캠퍼스) 일반대학원 컴퓨터 공학과 석사
- 2005년 강원대학교(강릉캠퍼스) 일반대학원 컴퓨터 공학과 박사수료
- 현 메인비즈 강원연합회 이사
- 현 (주)인터포 대표이사
- 관심분야: 인공지능, 가상융합콘텐츠, 디지털트윈
- https://orcid.org/0009-0001-1900-8470

김선정
- 1996년 고려대학교 컴퓨터학과 학사
- 1998년 고려대학교 컴퓨터학과 석사
- 2003년 고려대학교 컴퓨터학과 박사
- 2000년 이스라엘 Tel-Aviv 대학교 연구원
- 2004년 독일 RWTH-Aachen 대학교 박사후연구원
- 2005년~현재 한림대학교 컴퓨터공학과 교수
- 관심분야: 컴퓨터그래픽스, 가상/증강현실, 모바일게임 및 앱 개발
- https://orcid.org/0000-0002-8663-4578