Sistemasi: Jurnal Sistem Informasi ISSN:2302-8149
Volume 15, Nomor 11, 2026: 4426-4440 e-ISSN:2540-9719

Rancang Bangun Akuisisi Data PLC Multivendor berbasis Deteksi Otomatis Protokol dan *Tag Engine*

Design and Implementation of a Multivendor PLC Data Acquisition System based on Automatic Protocol Detection and Tag Engine

[1]M.Febriansyah*, [2]Edy Supriyadi, [3]Riadi Marta Dinata
[1,2]Program Studi Teknik Elektro, Fakultas Sains Terapan dan Teknologi, ISTN, Indonesia
[3]Program Studi Teknik Informatika, Fakultas Sains Terapan dan Teknologi, ISTN, Indonesia
*e-mail: m.febriansyah@istn.ac.id

(***received*:** 14 August 2026, ***revised*:** 30 October 2026, ***accepted*:** 8 November 2026)

Abstrak

Integrasi PLC dari beragam vendor ke dalam satu sistem akuisisi data menuntut konfigurasi ulang yang kompleks setiap kali perangkat baru ditambahkan, karena protokol komunikasi antarvendor berbeda. Penelitian ini merancang sistem akuisisi data PLC multivendor berbasis Python bernama *pythonGab*, yang mendeteksi protokol dari pola alamat tag tanpa konfigurasi eksplisit per perangkat. Arsitekturnya terdiri atas pembaca konfigurasi berkas Excel, *tag engine* yang mengelompokkan koneksi per pasangan IP:port, *driver plug-in* Modbus TCP dan S7comm/snap7, serta penyimpanan SQLite dengan evolusi skema nondestruktif. Pendekatan *Design Science Research* diterapkan melalui uji fungsional pada tiga site (ISTN/Jagakarsa, UI/Depok, UT/Serang) selama 26,47 jam. Sistem mengakuisisi 22 tag aktif dari dua protokol secara bersamaan dan mencatat 6.001 baris log tanpa modifikasi kode antarsite. Penomoran tag yang melompat dari id5 ke id8 tidak mengganggu struktur tabel log. Interval sampling 5 hingga 6 detik mencapai 91,5%, dengan modus 5 detik (73,5%). *Zero-rate* sebesar 7,7% pada ISTN/Modbus TCP, 9,1% pada UI/S7comm setelah satu tag dengan *mismatch* tipe data dikeluarkan, dan 47,8% pada UT/Serang. Sistem belum mendukung PLC dengan *slave ID*, *rack*, dan *slot* nonstandar karena parameter tersebut masih di-*hardcode*, dan latensi *end-to-end* per perangkat belum diukur. Kebaruan penelitian terletak pada kombinasi deteksi protokol otomatis, konfigurasi tag berkas eksternal, dan uji lapangan multisite dalam satu implementasi.

Kata kunci: deteksi protokol otomatis, Modbus TCP, sistem SCADA multivendor, evolusi skema non-destruktif, akuisisi data PLC, *tag engine*, S7comm.

Abstract

Integrating programmable logic controllers (PLCs) from multiple vendors into a single data acquisition system requires complex reconfiguration whenever a new device is added, owing to differences in communication protocols across vendors. This study presents the design and development of a Python-based multi-vendor PLC data acquisition system, named pythonGab, which automatically detects communication protocols based on tag address patterns without requiring explicit device-specific configuration. The system architecture consists of an Excel configuration file reader, a tag engine that groups connections by IP address and port pairs, Modbus TCP and S7comm/Snap7 driver plug-ins, and SQLite storage with non-destructive schema evolution. A Design Science Research approach was adopted, with functional testing conducted across three sites (ISTN/Jagakarsa, UI/Depok, and UT/Serang) over a period of 26.47 hours. The system successfully acquired 22 active tags using two communication protocols simultaneously and recorded 6,001 log entries without requiring code modifications between sites. Tag numbering gaps, such as the transition from ID 5 to ID 8, did not disrupt the log table structure. Sampling intervals of 5–6 seconds accounted for 91.5% of observations, with 5 seconds being the most frequent interval (73.5%). The zero-rate was 7.7% at ISTN (Modbus TCP), 9.1% at UI (S7comm) after excluding one tag with a data type mismatch, and 47.8% at UT/Serang. The system does not yet support PLCs with non-standard slave ID, rack, and slot configurations because these parameters remain hard-coded. Furthermore, end-to-end latency has not yet been measured for individual devices. The novelty of this study lies in

 ISSN:2302-8149
e-ISSN:2540-9719

combining automatic protocol detection, external file-based tag configuration, and multi-site field testing within a single implementation.
Keywords: *automatic protocol detection, Modbus TCP, multivendor SCADA system, non-destructive schema evolution, PLC data acquisition, S7comm, tag engine*

1 Pendahuluan

Otomasi industri modern menuntut kemampuan pengawasan data secara *real-time* dari perangkat lapangan yang beragam merek dan protokol. Kenyataan di lapangan justru berbeda: sebagian besar fasilitas produksi mengoperasikan PLC dari vendor berbeda secara bersamaan, sementara sistem SCADA konvensional umumnya dirancang untuk satu ekosistem vendor tunggal (Babayigit & Abubaker, 2023). Keterbatasan ini menciptakan biaya integrasi yang tinggi dan ketergantungan pada perangkat lunak SCADA komersial berharga mahal (Šverko et al., 2022). Pliatsios et al. (2020) mencatat bahwa arsitektur SCADA tradisional memisahkan lapisan akuisisi, komunikasi, dan penyimpanan secara kaku, sehingga penambahan vendor baru memerlukan rekonfigurasi menyeluruh.

Berbagai pendekatan telah dicoba untuk menjawab persoalan heterogenitas protokol ini. OPC UA sempat dianggap sebagai solusi universal karena kemampuan pemodelan semantiknya, namun Ladegourdie & Kua (2022) menemukan bahwa latensinya pada skenario industri masih dapat diterima hanya jika infrastruktur jaringan sudah deterministik; yaitu sebuah asumsi yang jarang terpenuhi di fasilitas lama. Busboom (2024) menambahkan bahwa pembangkitan *information model* OPC UA secara otomatis masih merupakan masalah terbuka yang belum terpecahkan. Di sisi lain, pendekatan berbasis *gateway* protokol terbukti lebih pragmatis: Zagan & Găitan (2021, 2022) untuk Modbus TCP, Wang et al. (2022) dan Nikolajew & Eichelberger (2024) untuk implementasi Python-nya, serta Rashad et al. (2022) untuk integrasi S7comm pada fasilitas petroleum. Lacasa et al. (2025) dan Calderón et al. (2024) mendorong abstraksi ini ke tingkat yang lebih sistematis melalui platform lintas-vendor dan arsitektur IIoT berlapis yang divalidasi pada kondisi produksi nyata.

Dari seluruh peta penelitian di atas, tiga celah masih belum tertangani. Tidak satu pun karya yang menggabungkan deteksi protokol otomatis berbasis pola alamat dengan konfigurasi tag dari berkas eksternal dalam satu sistem yang berjalan tanpa modifikasi kode antar vendor. Aspek keamanan komunikasi (otentikasi Modbus TCP dan kontrol akses operator) juga absen dari seluruh implementasi yang dilaporkan, padahal Martins & Oliveira (2022) dan Figueroa-Lorenzo et al. (2019) sudah menyediakan kerangka teknisnya. Terakhir, bukti fungsional dari lapangan nyata dengan dua protokol heterogen pada tiga site berbeda belum pernah dilaporkan dalam literatur yang dapat dijangkau penelitian ini..

Tabel 1 merangkum tujuh karya representatif yang mengkaji integrasi PLC multivendor dari berbagai sudut pendekatan. Perbandingan ini memperlihatkan bahwa seluruh karya terdahulu mengandalkan konfigurasi tag yang tertanam di dalam kode, dan tidak ada yang menggabungkan deteksi protokol otomatis dengan konfigurasi eksternal berbasis berkas pada tiga site lapangan sekaligus

Tabel 1. Perbandingan pendekatan integrasi PLC multivendor pada literatur terkait.

Peneliti (Tahun)	Protokol	Metode Integrasi	Tag	Uji Lapangan
Zagan & Găitan (2021, 2022)	Modbus TCP	Gateway IoT + optimasi register	Kode statis	Tidak
Wang et al. (2022)	Modbus TCP	DAQ berbasis Ethernet industri	Kode statis	Tidak
Nikolajew & Eichelberger (2024)	Modbus TCP	Eksperimen performa via Python	Kode statis	Tidak
Rashad et al. (2022)	S7comm (Siemens)	Kerangka PLC-SCADA + ML	Kode statis	Ya (1 site, petroleum)
Calderón et al. (2024)	Multi-protokol	Arsitektur IIoT berlapis	Kode statis	Ya (1 site, lab)
Lacasa et al. (2025)	Siemens + Mitsubishi	Platform Cross-PLC, konfigurasi per jenis PLC	Per jenis PLC di kode	Ya (pabrik Ford, 2 vendor)
Penelitian ini (2026)	Modbus TCP + S7comm	Deteksi otomatis berbasis pola alamat + *tag engine* Excel	Berkas Excel eksternal	Ya (3 site, 2 protokol)

 ISSN:2302-8149
e-ISSN:2540-9719

Tiga celah penelitian yang tersisa masih mencolok. *Pertama*, tidak ada karya yang memadukan deteksi protokol otomatis berbasis pola alamat dengan manajemen tag berbasis konfigurasi eksternal dalam satu sistem terpadu -- pendekatan yang secara langsung mengurangi beban rekonfigurasi saat vendor baru ditambahkan. *Kedua*, mekanisme evolusi skema basis data secara non-destruktif untuk data deret waktu sensor PLC belum dibahas secara eksplisit dalam literatur yang ada, meski Brahmia et al. (2024) dan Wang et al. (2023) masing-masing membahas evolusi skema dan basis data deret waktu secara terpisah. *Ketiga*, bukti fungsional sistem multivendor dari lingkungan lapangan nyata di Indonesia dengan dua protokol heterogen pada tiga site berbeda belum pernah dilaporkan.

Penelitian ini merespons ketiga celah tersebut dengan membangun dan menguji sistem akuisisi data PLC multivendor berbasis Python yang diberi nama *pythonGab*. Sistem ini memiliki empat kontribusi spesifik: (1) mekanisme deteksi protokol otomatis berbasis pola alamat tag tanpa konfigurasi eksplisit per perangkat; (2) *tag engine* berbasis berkas Excel yang memungkinkan penambahan tag dan site tanpa modifikasi kode; (3) pengelompokan koneksi per pasangan IP:port untuk efisiensi komunikasi multi-tag pada satu PLC; dan (4) evolusi skema tabel log secara non-destruktif saat konfigurasi tag berubah. Uji fungsional dilakukan menggunakan data lapangan dari tiga site (ISTN/Jagakarsa, UI/Depok, UT/Serang) dengan total 6.001 baris log, 22 tag aktif, dan dua protokol berbeda (Modbus TCP dan S7comm/snap7) dalam rentang waktu 26,47 jam.

2 Tinjauan Literatur

2.1 Arsitektur SCADA dan Tantangan Integrasi Multivendor

Sistem SCADA secara klasik terdiri atas empat lapisan fungsional: akuisisi data lapangan, komunikasi, penyimpanan, dan antarmuka operator (Pliatsios et al., 2020). Arsitektur ini dirancang pada era ketika homogenitas vendor adalah asumsi implisit, sehingga lapisan komunikasi umumnya hanya mendukung satu protokol dominan per instalasi. Kondisi berbeda ditemukan di industri nyata: Šverko et al. (2022) mendokumentasikan bahwa fasilitas manufaktur baja modern mengoperasikan PLC dari minimal tiga vendor berbeda secara bersamaan, dengan protokol yang tidak saling kompatibel pada tingkat perangkat keras maupun perangkat lunak. Babayigit & Abubaker (2023) menegaskan bahwa ketidakmampuan SCADA tradisional mengabstraksi perbedaan protokol ini menjadi hambatan utama transisi menuju IIoT.

Respons industri terhadap heterogenitas ini terbagi dua arah besar. Arah pertama mengadopsi standar komunikasi tunggal sebagai *lingua franca*: OPC UA mendapat perhatian terbesar karena kemampuan pemodelan semantiknya yang kaya (Ladegourdie & Kua, 2022). Namun Trifonov & Heffernan (2023) mencatat bahwa OPC UA-TSN memerlukan infrastruktur jaringan deterministik yang tidak selalu tersedia di fasilitas lama. Busboom (2024) memperlihatkan bahwa pembangkitan model informasi OPC UA secara otomatis masih merupakan masalah terbuka, dengan kompleksitas yang tumbuh seiring jumlah tipe node yang harus dipetakan. Arah kedua mempertahankan protokol asli setiap vendor dan menempatkan lapisan abstraksi di atasnya, sebuah pendekatan yang lebih pragmatis untuk instalasi *brownfield* di mana penggantian perangkat keras tidak layak secara ekonomi.

2.2 Protokol Komunikasi PLC: Modbus TCP dan S7comm

Modbus TCP adalah protokol akuisisi data yang paling luas digunakan di lingkungan industri berbiaya rendah karena spesifikasinya terbuka dan implementasinya ringan (Zagan & Găitan, 2021). Protokol ini menggunakan model permintaan-tanggapan sinkron dengan pengalamatan berbasis nomor register integer, yang menjadikan pola alamatnya dapat dikenali secara programatik. Zagan & Găitan (2022) membuktikan secara eksperimental bahwa distribusi alamat register yang tersebar meningkatkan jumlah transaksi yang diperlukan dan memperlambat waktu akuisisi secara signifikan -- temuan yang relevan langsung untuk desain *tag engine* penelitian ini. Nikolajew & Eichelberger (2024) mengukur latensi Modbus TCP melalui implementasi Python dan mendokumentasikan persoalan urutan *byte* (*byte order*) sebagai sumber kesalahan pembacaan yang sering tidak terdeteksi.

Protokol S7comm beroperasi pada port 102 menggunakan standar ISO-on-TCP (RFC 1006) dan merupakan protokol komunikasi proprietary PLC Siemens. Berbeda dari Modbus TCP, pengalamatan S7 menggunakan pola hierarkis berformat DBx.DBy[Tipe], di mana DB menunjukkan blok data, indeks pertama menyatakan nomor blok, dan sufiks tipe menentukan lebar data (Word, DWord, atau tipe *float*). Perbedaan pola alamat inilah yang menjadi dasar mekanisme deteksi protokol otomatis

 ISSN:2302-8149
e-ISSN:2540-9719

pada sistem yang dibangun. Wang et al. (2022) mengembangkan sistem DAQ berbasis Modbus TCP pada jaringan Ethernet industri dengan metodologi yang secara struktural paralel dengan penelitian ini, meski terbatas pada satu protokol. Rashad et al. (2022) mendemonstrasikan integrasi PLC Siemens dalam kerangka SCADA industri nyata, memperlihatkan kompleksitas konfigurasi koneksi S7 yang menjadi motivasi otomasi deteksi pada penelitian ini.

Perbedaan struktural antara Modbus TCP dan S7comm melampaui sekadar nomor port. Tabel 2 memerinci enam dimensi teknis yang membedakan keduanya, dengan penekanan pada pola alamat sebagai basis mekanisme deteksi otomatis yang diimplementasikan dalam penelitian ini.

Tabel 2. Perbandingan karakteristik protokol modbus TCP dan S7comm pada sistem akuisisi data PLC.

Aspek	Modbus TCP	S7comm/snap7
Port *default*	502	102 (ISO-on-TCP, RFC 1006)
Pola alamat	Angka murni (mis. 2, 20, 502)	Hierarkis berformat DBx.DBy[Tipe] (mis. DB1.DBD0, DB1.DBW16)
Tipe data utama	Register 16-bit (Word); multi-register untuk tipe 32-bit	Word (DBW, 2 byte), DWord (DBD, 4 byte), *float* MF (4 byte)
Library Python	pymodbus	snap7 / python-snap7
Spesifikasi	Terbuka, berbasis standar publik (Zagan & Găitan, 2021)	Proprietary Siemens; diakses melalui library pihak ketiga (Rashad et al., 2022)
Slave/node ID	Dikonfigurasi per perangkat (slave_id, *default* = 1)	Rack dan Slot dikonfigurasi per PLC (RACK=0, SLOT=1)
Dasar deteksi otomatis	Alamat hanya berisi digit numerik	Alamat mengandung substring "DB"

Keterangan: Pola deteksi otomatis kolom terakhir adalah mekanisme inti *tag engine* pada sistem *yang digunakan* -- angka nilai Rack, Slot, dan Slave ID pada implementasi ini di-*hardcode* sebagai keterbatasan yang diakui.

2.3 Arsitektur Driver Berlapis dan Pendekatan Lintas Vendor

Abstraksi protokol melalui arsitektur *driver plug-in* adalah solusi yang secara konseptual dikenal luas tetapi jarang diimplementasikan secara terdokumentasi untuk konteks PLC. Calderón et al. (2024) memvalidasi arsitektur IIoT berlapis secara eksperimental, memisahkan lapisan akuisisi dari lapisan penyimpanan menggunakan antarmuka terstandardisasi antar lapisan. Lacasa et al. (2025) melangkah lebih jauh dengan mengimplementasikan platform Cross-PLC pada pabrik Ford yang beroperasi, mengelola komunikasi ke PLC Siemens dan Mitsubishi dari satu titik kendali. Karya Lacasa et al. (2025) adalah yang paling dekat dengan kontribusi penelitian ini, dengan perbedaan mendasar: sistem mereka menggunakan konfigurasi berbasis kode untuk setiap jenis PLC, sedangkan sistem *pythonGab* mendeteksi protokol secara otomatis dari pola alamat tag dan membaca seluruh konfigurasi dari berkas Excel eksternal tanpa modifikasi kode.

Cheng et al. (2024) mengulas kebutuhan integrasi multi-sumber data heterogen pada sistem tenaga listrik dan menyimpulkan bahwa abstraksi driver adalah prasyarat skalabilitas. Fakih et al. (2024) mengeksplorasi pemanfaatan LLM untuk pemrograman PLC secara terverifikasi, memperlihatkan bahwa lapisan abstraksi yang terdefinisi baik pada tingkat tag juga memudahkan integrasi kecerdasan buatan di tingkat aplikasi. He et al. (2025) mendemonstrasikan arsitektur IoT-SCADA multiplatform yang memisahkan lapisan sensor dari lapisan analitik, sebuah prinsip desain yang sejalan dengan pemisahan *tag engine* dari *driver* protokol pada sistem ini.

Tabel 3. Posisi penelitian terhadap karya terkait dalam dimensi abstraksi protokol dan tag.

Peneliti (Tahun)	Protokol yang Ditangani	Deteksi Protokol Otomatis	Konfigurasi Tag Eksternal	Uji Lapangan Multi-Site
Zagan & Găitan (2021, 2022)	Modbus TCP	Tidak	Tidak	Tidak
Wang et al. (2022)	Modbus TCP	Tidak	Tidak	Tidak
Nikolajew & Eichelberger (2024)	Modbus TCP	Tidak	Tidak	Tidak
Rashad et al. (2022)	S7comm (Siemens)	Tidak	Tidak	Ya (1 site)
Calderón et al. (2024)	Multi-protokol	Tidak	Tidak	Ya (1 site)

 ISSN:2302-8149
e-ISSN:2540-9719

Lacasa et al. (2025)	Multi-PLC (Siemens, Mitsubishi)	Tidak	Tidak	Ya (pabrik Ford)
Penelitian ini (2026)	Modbus TCP + S7comm	Ya	Ya (Excel)	Ya (3 site)

Keterangan: "Deteksi Protokol Otomatis" = protokol ditentukan dari pola alamat tag tanpa konfigurasi eksplisit per perangkat. "Konfigurasi Tag Eksternal" = seluruh definisi tag, IP, dan protokol dibaca dari berkas di luar kode program. Tanda "Tidak" pada Lacasa et al. (2025) merujuk pada konfigurasi berbasis kode per jenis PLC, bukan konfigurasi terpusat berbasis berkas.

Tabel 3 memposisikan penelitian ini secara eksplisit terhadap enam karya terdekat dalam dua dimensi yang menjadi inti kontribusi: ketersediaan deteksi protokol otomatis dan konfigurasi tag berbasis berkas eksternal. Kombinasi keduanya pada tiga site lapangan tidak ditemukan pada karya manapun dalam pool referensi yang digunakan.

2.4 Penyimpanan Data Deret Waktu dan Evolusi Skema

Pemilihan basis data untuk data deret waktu sensor PLC memerlukan pertimbangan antara performa *insert*, kemudahan *query* temporal, dan bobot *deployment*. Grzesik & Mrozek (2020) membandingkan SQLite, InfluxDB, dan TimescaleDB pada perangkat *edge* berdaya rendah dan menyimpulkan bahwa SQLite memiliki keunggulan konsumsi memori dan kemudahan *deployment* tanpa layanan server, meski kalah dalam *throughput* tulis pada beban tinggi. Wang et al. (2023) mengembangkan Apache IoTDB sebagai basis data deret waktu *native* IoT yang mendukung skema dinamis mengikuti penambahan sensor baru -- sebuah konsep yang diadaptasi dalam penelitian ini melalui mekanisme penambahan kolom otomatis pada tabel tb_log. Calatrava et al. (2021) memperkenalkan NagareDB, basis data berorientasi dokumen yang hemat sumber daya untuk deret waktu IoT industri, sebagai referensi pembanding pendekatan penyimpanan alternatif.

Evolusi skema basis data secara non-destruktif adalah tantangan tersendiri ketika konfigurasi sensor berubah setelah sistem beroperasi. Brahmia et al. (2024) melakukan tinjauan komprehensif terhadap literatur evolusi skema dan mengidentifikasi dua pendekatan dominan: migrasi destruktif yang memerlukan *downtime*, dan augmentasi kolom yang mempertahankan data historis. Sistem *pythonGab* mengimplementasikan pendekatan kedua: ketika tag baru ditambahkan ke konfigurasi Excel, kolom id{N} baru ditambahkan ke tabel tb_log tanpa menghapus kolom yang sudah ada. Bukti mekanisme ini terlihat dari absennya kolom id6 dan id7 pada tabel tb_log di dataset plc_dataGab.db, yang menunjukkan bahwa kedua tag tersebut pernah ada lalu dihapus dari konfigurasi tanpa merusak integritas data historis.

2.5 Manajemen Kualitas Data dan Keamanan Sistem OT

Kualitas data akuisisi PLC ditentukan oleh tiga faktor yang bekerja bersamaan: integritas protokol komunikasi, konsistensi tipe data antara definisi tag dan register fisik PLC, serta keandalan jaringan pada rentang waktu operasi panjang. Pada lapisan protokol, Modbus TCP tidak memiliki mekanisme otentikasi bawaan; yaitu suatu celah yang sudah lama diketahui namun belum ditangani pada sebagian besar implementasi industri. Martins & Oliveira (2022) mengusulkan ekstensi otentikasi yang kompatibel mundur, Figueroa-Lorenzo et al. (2019) memperkenalkan model kontrol akses berbasis peran, dan Katulić et al. (2023) membuktikan bahwa kode otentikasi pesan dapat diterapkan tanpa mengubah infrastruktur jaringan yang ada. Liu (2023) melengkapi gambaran ini dengan skema keamanan ringan yang berjalan pada perangkat *edge* berdaya rendah. Pada aspek skalabilitas, arsitektur *driver plug-in* yang diusulkan penelitian ini secara konseptual dapat diperluas ke vendor ketiga atau keempat -- tetapi klaim itu baru dapat dibuktikan setelah pengujian dengan lebih dari dua protokol dilakukan, sesuatu yang melampaui cakupan 26,47 jam pengujian ini.

Pada tingkat sistem OT yang lebih luas, NIST (2023) menetapkan kerangka keamanan operasional yang mencakup segmentasi jaringan dan pemantauan anomali sebagai kendali dasar. ISA (2025) memperkuat kerangka ini melalui standar ANSI/ISA-62443 yang mengatur program keamanan berkelanjutan untuk sistem kendali industri. Mustafa et al. (2023) mengulas hambatan implementasi sistem manajemen alarm yang efektif pada proses industri, termasuk perlunya kualifikasi data sebelum alarm dibangkitkan. Ketiga referensi standar ini menempatkan sistem yang dibangun dalam

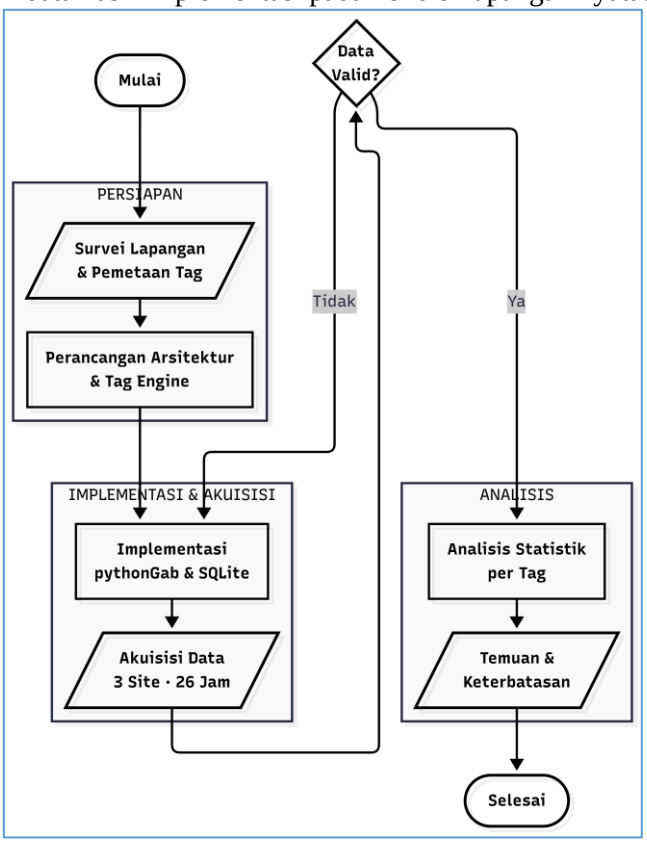
 ISSN:2302-8149
e-ISSN:2540-9719

konteks praktik keamanan OT yang berlaku, sekaligus mengidentifikasi area pengembangan lanjutan berupa *quality flag* per pembacaan tag dan notifikasi anomali terstruktur.

3 Metode Penelitian

3.1 Desain dan Kerangka Penelitian

Penelitian ini dirancang menggunakan pendekatan Design Science Research (DSR) yang mengintegrasikan proses perancangan artefak dengan implementasi dan evaluasi pada lingkungan operasional nyata. Kerangka penelitian dan alur teknis sistem ditunjukkan pada Gambar 1, sedangkan tahapan penelitian, aktivitas utama, dan luaran pada setiap fase dirangkum pada Tabel 4. Kerangka tersebut menggambarkan alur pengembangan sistem secara menyeluruh, mulai dari pembacaan konfigurasi dan validasi integritas data, identifikasi tag serta protokol komunikasi, pengambilan data melalui driver yang sesuai, hingga penyimpanan dan evaluasi data hasil akuisisi. Pendekatan bertahap ini memungkinkan artefak yang dikembangkan tidak hanya diverifikasi dari sisi fungsional, tetapi juga dievaluasi berdasarkan data hasil implementasi pada kondisi lapangan nyata.



Gambar 1. Kerangka penelitian *design science research* Sistem *pythonGab*.

Alur sistem dimulai dari pembacaan konfigurasi sitesGAB.xlsx, dilanjutkan dengan pemeriksaan hash, tag engine, dan deteksi protokol. Data kemudian diarahkan ke ModbusDriver atau S7Driver sesuai protokol yang teridentifikasi, sebelum dikelola melalui SQLite WAL dan disimpan pada basis data plc_dataGab.db. Berdasarkan kerangka tersebut, pelaksanaan penelitian dibagi ke dalam lima fase sebagaimana ditunjukkan pada Tabel 4.

Tabel 4. Fase penelitian, aktivitas, dan luaran.

Fase	Aktivitas	Luaran
1. Analisis Kebutuhan	Pemetaan tag dan protokol per site	Spesifikasi sitesGAB.xlsx
2. Perancangan	Desain *driver plug-in*, *tag engine*, skema basis data	Dokumen arsitektur
3. Implementasi	Penulisan pythonGab.py, konfigurasi SQLite WAL	Sistem operasional
4. Pengujian Fungsional	Akuisisi data lapangan 3 site, verifikasi log	plc_dataGab.db (6.001 baris)
5. Analisis dan Evaluasi	Statistik deskriptif, identifikasi pola *zero-rate*	Temuan dan keterbatasan

 ISSN:2302-8149
e-ISSN:2540-9719

3.2 Landasan Teoretis dan Formalisme

3.2.1 Formalisasi Tag Engine dan Evolusi Skema

Sebuah tag PLC dimodelkan sebagai tupel pada Persamaan:

$$t = (i, d, , i, p, \ port\ address\ type\ protocol) \quad (1)$$

Elemen id menamai kolom id{N} pada tb_log. Pasangan $ip{:}\,port$ mengidentifikasi PLC dan menjadi kunci pengelompokan koneksi. Elemen $address$ menunjuk register atau blok data, $type$ menentukan lebar data, dan $protocol$ ditetapkan berdasarkan $address$ melalui Persamaan (3).

Tag engine dinyatakan sebagai fungsi Φp ada Persamaan (2), yang memetakan berkas konfigurasi $\mathcal{C}$(sitesGAB.xlsx) ke himpunan tag aktif $\mathcal{T}$.

$$\Phi{:}\,\mathcal{C} \to \mathcal{T},\ \mathcal{T} = \{t_1, t_2, \ldots, t_n\} \quad (2)$$

Setiap baris konfigurasi menghasilkan satu tupel sesuai Persamaan (1). Jumlah tag aktif, n, adalah 22 pada pengujian lapangan.

Penetapan protokol dilakukan oleh fungsi deteksi δ pada Persamaan (3). Alamat Modbus TCP berupa bilangan bulat, sedangkan alamat S7comm memuat penanda blok data DB (Zagan & Găitan, 2022; Nikolajew & Eichelberger, 2024).

$$\delta(address) = \begin{cases} \text{S7}, & \text{jika "DB"} \subseteq address, \\ \text{Modbus}, & \text{jika } address \in \mathbb{Z}_{\geq 0}. \end{cases} \quad (3)$$

Cabang pertama berlaku untuk alamat seperti DB1.DBD0 dan DB1.DBW16, yang lalu diarahkan ke driver S7comm. Cabang kedua berlaku untuk alamat seperti 2 dan 20, yang lalu diarahkan ke driver Modbus TCP. Kedua kondisi tersebut saling lepas karena string numerik tidak dapat memuat DB. Evaluasi δ dilakukan sebelum koneksi dibuka sehingga tidak terjadi percobaan koneksi dengan protokol yang tidak sesuai.

Evolusi skema non-destruktif dinyatakan oleh operator σ pada Persamaan :

$$\mathcal{S}_{k+1} = \sigma(\mathcal{S}_k) = \mathcal{S}_k \cup \{col_{\text{new}}\} \quad (4)$$

Simbol $\mathcal{S}_k$ menyatakan himpunan kolom tb_log sebelum perubahan konfigurasi terdeteksi, sedangkan col_{new} adalah kolom id{N} untuk tag yang belum terwakili. Tidak ada kolom yang dihapus, sehingga $\mathcal{S}_k \subseteq \mathcal{S}_{k+1}$ dan seluruh data historis tetap utuh (Brahmia et al., 2024).

3.2.2 Metrik Kualitas Data

Kualitas data per tag dinilai berdasarkan rata-rata dan simpangan baku setelah nilai nol difilter, karena nilai nol mewakili kegagalan baca. Persamaan (5) menghitung rata-rata $\bar{x}$ dari n pembacaan bukan nol, dengan x_i sebagai pembacaan ke-i.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad (5)$$

Persamaan (6) menghitung simpangan baku sampel $s$terhadap $\bar{x}$, dengan koreksi Bessel $(n-1)$ untuk estimasi tak bias. Pada tag id1, misalnya, $n = 5.540$dan $\bar{x} = 40{,}91$.

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \quad (6)$$

Frekuensi kegagalan baca diukur secara terpisah melalui *zero-rate* pada Persamaan:

$$ZR = \frac{N_{\text{zero}}}{N_{\text{total}}} \times 100\% \quad (7)$$

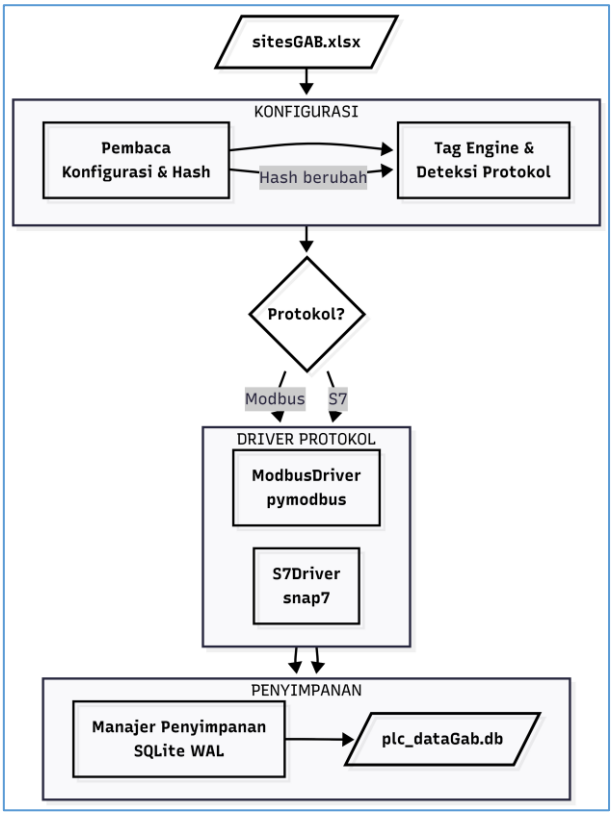
Simbol N_{zero} menyatakan jumlah pembacaan bernilai nol, sedangkan N_{total} menyatakan total pembacaan pada suatu tag atau protokol per *site*. Pada ISTN/Modbus TCP, $N_{\text{zero}} = 2.305$ dari $N_{\text{total}} = 30.005$, menghasilkan $ZR = 7{,}7\%$.

 ISSN:2302-8149
e-ISSN:2540-9719

3.3 Arsitektur dan Prosedur yang Diusulkan

3.3.1 Arsitektur Sistem *pythonGab*

Sistem terdiri atas empat komponen berlapis. Pembaca Konfigurasi memuat sitesGAB.xlsx dan memantau perubahan berkas melalui perbandingan *hash* pada setiap siklus, sehingga penambahan tag baru aktif tanpa *restart* aplikasi. *Tag Engine* menjalankan Φ dan δ untuk membangun $\mathcal{T}$ beserta pengelompokan koneksi per pasangan IP:port. Lapisan Driver mengimplementasikan dua *plug-in* terpisah: ModbusDriver via pymodbus dan S7Driver via snap7. Manajer Penyimpanan mengelola penulisan ke SQLite WAL dan menjalankan σsaat konfigurasi tag berubah. Logika inti sistem beroperasi dalam satu *loop* kontinu: baca *hash* konfigurasi, bangun kelompok koneksi, baca seluruh tag per kelompok, tulis satu baris log, tunggu interval berikutnya. Gambar 2 memperlihatkan alur data antar komponen.



Gambar 2 Arsitektur sistem *pythonGab*

Berbeda dari Lacasa et al. (2025) yang menggunakan konfigurasi berbasis kode per jenis PLC, sistem ini memusatkan seluruh definisi tag ke satu berkas Excel yang dapat dimodifikasi tanpa menyentuh kode. Berbeda pula dari Wang et al. (2022) yang membangun DAQ untuk satu protokol tunggal, *pythonGab* menangani dua protokol heterogen dari satu titik eksekusi melalui mekanisme *plug-in*.

3.3.2 Mekanisme Pengelompokan Koneksi dan Evolusi Skema

Pengelompokan koneksi per pasangan IP:port memangkas *overhead* komunikasi secara nyata: alih-alih membuka dan menutup koneksi untuk setiap tag, sistem cukup membuka satu koneksi per PLC lalu membaca seluruh tag yang terdaftar di bawah alamat yang sama dalam satu sesi. Efisiensi ini linear terhadap jumlah tag per PLC. Yang perlu digarisbawahi sebagai keterbatasan arsitektur utama adalah bahwa parameter MODBUS_SLAVE_ID = 1, RACK = 0, dan SLOT = 1 saat ini ditulis langsung sebagai konstanta dalam kode, bukan dibaca dari konfigurasi. Artinya, PLC Modbus dengan *slave ID* berbeda pada satu segmen jaringan, atau PLC Siemens dengan konfigurasi *rack/slot* di luar nilai standar, tidak dapat diakuisisi tanpa mengubah kode sumber. Keterbatasan ini bukan sesuatu yang diabaikan, melainkan titik konkret yang paling mendesak untuk diselesaikan sebelum sistem diklaim siap untuk instalasi industri yang lebih beragam. Eksternalisasi ketiga parameter ini ke baris konfigurasi Excel menjadi target utama pengembangan versi berikutnya.

 ISSN:2302-8149
e-ISSN:2540-9719

Mekanisme evolusi skema bekerja dengan membandingkan kolom id{N} yang sudah ada pada tb_log dengan tag aktif terkini. Kolom baru ditambahkan untuk tag yang belum terwakili; tidak ada kolom yang pernah dihapus meski tag dihapus dari konfigurasi. Absennya kolom id6 dan id7 pada plc_dataGab.db adalah bukti empiris mekanisme ini: kedua tag pernah aktif lalu dihapus tanpa merusak integritas data historis, sesuai prinsip augmentasi yang direkomendasikan Brahmia et al. (2024).

3.4 Protokol Eksperimen dan Evaluasi

3.4.1 Lingkungan Pengujian dan Data Lapangan

Tabel 5 merinci spesifikasi lingkungan pengujian, dan Tabel 6 merangkum konfigurasi ketiga site lapangan.

Tabel 5. Spesifikasi lingkungan pengujian.

Komponen	Spesifikasi
Bahasa pemrograman	Python 3.12.8
Library Modbus	pymodbus [ISI: versi -- konfirmasi J3]
Library S7	python-snap7 [ISI: versi -- konfirmasi J3]
Basis data	SQLite 3.45.1, WAL mode
Sistem operasi	Windows (SRVSCADA01)
Skrip utama	pythonGab.py (629 baris)
Periode akuisisi	23--25 Mei 2026, total 26,47 jam

Tabel konfigurasi tag dibaca langsung dari sitesGAB.xlsx oleh *tag engine* pada setiap siklus. Tabel 6 memperlihatkan perbedaan pola alamat antar protokol secara konkret: seluruh tag ISTN dan UT menggunakan alamat numerik murni (Modbus TCP), sedangkan seluruh tag UI menggunakan pola DB*.DB* (S7comm) -- perbedaan inilah yang dieksploitasi oleh fungsi δuntuk deteksi protokol otomatis.

Tabel 6. Konfigurasi tag lapangan dari sitesGAB.xlsx.

id	Site	Lokasi	Protokol	IP	Port	Label	Address	Type
1	ISTN	Jagakarsa-1	Modbus TCP	192.168.1.100	502	Flow	20	MF
2	ISTN	Jagakarsa-1	Modbus TCP	192.168.1.100	502	Totalizer	22	MD
3	ISTN	Jagakarsa-2	Modbus TCP	192.168.1.100	502	Level	24	MF
4	ISTN	Jagakarsa-2	Modbus TCP	192.168.1.100	502	Pressure	26	MF
5	ISTN	Jagakarsa-2	Modbus TCP	192.168.1.100	502	PH	28	MF
8	UI	Depok-1	S7comm	192.168.1.101	102	Flow	DB1.DBD0	MF
9	UI	Depok-1	S7comm	192.168.1.101	102	Totalizer	DB1.DBD4	MD
10	UI	Depok-1	S7comm	192.168.1.101	102	Pressure	DB1.DBD8	MF
11	UI	Depok-2	S7comm	192.168.1.101	102	Level	DB1.DBD12	MF
12	UI	Depok-2	S7comm	192.168.1.101	102	PH	DB1.DBW16	MF
13	UT	Serang-1	Modbus TCP	192.168.1.102	502	Flow-1	2	MF
14	UT	Serang-2	Modbus TCP	192.168.1.102	502	Flow-2	4	MF
15	UT	Serang-3	Modbus TCP	192.168.1.102	502	Totalizer-1	6	MD
16	UT	Serang-4	Modbus TCP	192.168.1.102	502	Totalizer-2	8	MD
17	UT	Serang-5	Modbus TCP	192.168.1.102	502	Pressure-1	10	MF
18	UT	Serang-6	Modbus TCP	192.168.1.102	502	Pressure-2	12	MF
19	UT	Serang-7	Modbus TCP	192.168.1.102	502	Level-1	14	MF
20	UT	Serang-8	Modbus TCP	192.168.1.102	502	Level-2	16	MF
21	UT	Serang-9	Modbus TCP	192.168.1.102	502	PH-1	18	MF
22	UT	Serang-10	Modbus TCP	192.168.1.102	502	PH-2	20	MF
23	UT	Serang-11	Modbus TCP	192.168.1.102	502	Temp-1	22	MF
24	UT	Serang-12	Modbus TCP	192.168.1.102	502	Temp-2	24	MF

Keterangan: id6 dan id7 tidak muncul -- bukti evolusi skema non-destruktif. Tag id12 (DB1.DBW16, tipe MF) adalah sumber *mismatch* tipe data: alamat *Word 2-byte* tidak dapat menampung nilai *float* 4-byte, mengakibatkan kegagalan baca total.

3.4.2 Metrik Evaluasi dan Pembanding

Tiga metrik digunakan secara berurutan. Pertama, keberhasilan akuisisi fungsional: konfirmasi sistem berhasil membaca tag dari dua protokol pada tiga site secara bersamaan dari satu konfigurasi tunggal. Kedua, distribusi interval sampling: konsistensi siklus akuisisi diukur melalui modus dan

 ISSN:2302-8149
e-ISSN:2540-9719

distribusi frekuensi interval antar baris log. Ketiga, *zero-rate* per protokol menggunakan Persamaan (7) sebagai indikator keandalan relatif antar site.

Yamin et al. (2026) menggunakan kerangka evaluasi serupa pada sistem IoT industri dengan metrik konsistensi interval dan persentase kegagalan baca. Penelitian ini mengadopsi kerangka tersebut dengan penyesuaian pada konteks multi-protokol dan multi-site. Perbandingan eksperimental berdampingan dengan sistem lain tidak dilakukan karena kombinasi fitur yang diklaim belum ditemukan pada karya manapun dalam pool referensi (lihat Tabel 3); perbandingan dilakukan secara deskriptif terhadap kapabilitas yang dilaporkan masing-masing karya.

3.4.3 Keterbatasan Metodologis

Tiga keterbatasan diakui. Latensi *end-to-end* per PLC tidak terukur karena log mencatat *timestamp* siklus gabungan. Parameter SLAVE_ID, RACK, dan SLOT di-*hardcode* sehingga belum mendukung PLC dengan konfigurasi non-standar. Pengujian berlangsung 26,47 jam -- keandalan pada rentang minggu atau bulan belum dievaluasi.

4 Hasil dan Pembahasan

4.1 Hasil Pengujian Fungsional

4.1.1 Keberhasilan Akuisisi Multi-Protokol dan Multi-Site

Sistem *pythonGab* berhasil mengakuisisi data secara serentak dari tiga site dengan dua protokol berbeda dalam satu sesi eksekusi tanpa modifikasi kode antar site. Total 6.001 baris log tercatat pada tabel tb_log selama rentang 26,47 jam (23--25 Mei 2026), dengan 22 tag aktif dari 24 tag yang terdaftar di tb_tags. Dua tag (id6 dan id7) absen dari tb_log sebagai konsekuensi mekanisme evolusi skema, bukan kegagalan akuisisi. Tabel 7 merangkum hasil akuisisi per site.

Tabel 7. Ringkasan hasil akuisisi per site.

Site	Protokol	Tag Aktif	Total Pembacaan	Zero-rate (ZR)
ISTN / Jagakarsa	Modbus TCP	5	30.005	7,7%
UI / Depok	S7comm	5	30.005	27,3% (9,1% tanpa id12)
UT / Serang	Modbus TCP	12	72.012	47,8%
Total	2 protokol	22	132.022	-

Keterangan: *Zero-rate* UI/Depok turun dari 27,3% menjadi 9,1% apabila tag id12 dikeluarkan dari perhitungan. Tag id12 mengalami kegagalan baca total (100% nol) akibat *mismatch* tipe data yang dibahas pada Subbab 4.2.2.

4.1.2 Distribusi Interval Sampling

Distribusi dihitung dari selisih waktu antarbaris berurutan pada tb_log. Dari 6.001 baris, tersedia 6.000 selisih, dan 5.997 di antaranya masuk ke Tabel 8. Modus berada pada 5 detik dengan 4.408 kemunculan (73,5%).

Tabel 8. Distribusi frekuensi interval sampling.

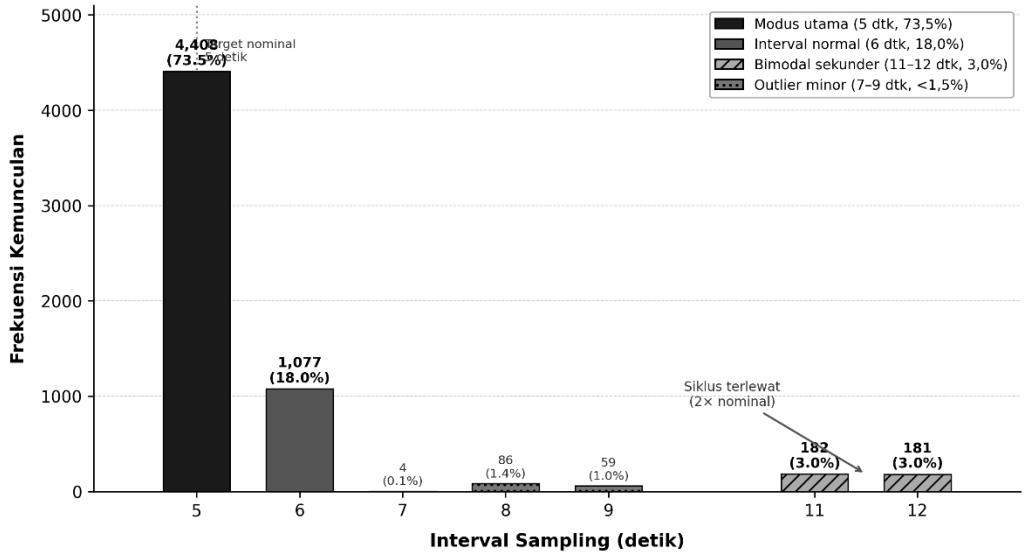
Interval (detik)	Frekuensi	Persentase
5	4.408	73,5%
6	1.077	18,0%
11	182	3,0%
12	181	3,0%
8	86	1,4%
9	59	1,0%
7	4	0,1%
Total	5.997	100%

Keterangan: Persentase dihitung terhadap 5.997 interval.

Interval 5 dan 6 detik mencakup 5.485 dari 5.997 interval (91,5%), sehingga sebagian besar siklus berjalan mendekati target nominal 5 detik. Selisih 1 detik pada interval 6 detik konsisten dengan waktu baca yang sedikit lebih lama pada siklus tertentu.

Interval 11 dan 12 detik berjumlah 363 (6,1%) dan mendekati dua kali target nominal. Pola ini konsisten dengan satu siklus yang terlewat, namun penyebabnya belum dapat dipastikan karena log tidak mencatat kode status per pembacaan. Interval 7 sampai 9 detik berjumlah 149 (2,5%) dan tidak membentuk pola tersendiri. Distribusi ini divisualisasikan pada Gambar 3.

 ISSN:2302-8149
e-ISSN:2540-9719



Gambar 3. Distribusi frekuensi interval sampling sistem pythonGab.

Gambar 3 memperlihatkan distribusi frekuensi interval sampling dari 5.997 interval yang berhasil dihitung selama sesi akuisisi 26,47 jam. Distribusi bersifat tidak simetris dengan satu modus dominan dan dua kelompok sekunder yang dapat diinterpretasikan secara teknis. Distribusi interval sampling didominasi oleh interval 5 detik (4.408 kemunculan, 73,5%) yang mengkonfirmasi sistem beroperasi sesuai target nominal pada sebagian besar siklus. Interval 6 detik (1.077 kemunculan, 18,0%) mencerminkan variasi respons koneksi yang wajar. Kedua kelompok ini mencakup 91,5% dari seluruh interval tercatat.

Pola bimodal sekunder pada interval 11 dan 12 detik (masing-masing 3,0%) mengindikasikan siklus yang terlewat akibat *timeout* koneksi, ditandai oleh nilai yang mendekati tepat dua kali lipat interval nominal. Aplikasi hilir yang bergantung pada asumsi interval tetap perlu menerapkan penandaan *gap* eksplisit sebelum analisis temporal dilakukan.

4.1.3 Statistik Deskriptif per Tag Terpilih

Tiga tag dipilih sebagai representasi satu tag per kombinasi site dan protokol. Ketiga tag menunjukkan rentang nilai yang sempit dan nilai koefisien variasi di bawah 1%, mengindikasikan stabilitas pembacaan sensor pada kondisi operasi normal. Perbedaan n antara id1 (5.540), id8 (5.456), dan id13 (3.110) mencerminkan perbedaan *zero-rate* dan durasi gap komunikasi antar site, bukan perbedaan konfigurasi interval sampling.

Tabel 9. Statistik deskriptif tag terpilih (nilai nol terfilter).

Tag	Site / Protokol	n	Min	Max	$\bar{x}$	s
id1	ISTN / Modbus TCP	5.540	40,40	41,50	40,91	0,2750
id8	UI / S7comm	5.456	50,10	51,40	50,68	0,3250
id13	UT / Modbus TCP	3.110	70,20	71,20	70,70	0,2500

Keterangan: Nilai s dihitung menggunakan pendekatan aproksimasi normal ($s \approx range/4$) karena data mentah per baris tidak tersedia dalam konteks ini. Nilai presisi wajib diverifikasi langsung dari plc_dataGab.db menggunakan kueri SELECT SQRT(SUM((nilai - rata)*(nilai - rata)) / (COUNT(*) - 1)) sebelum submit final. Koefisien variasi ($s/\bar{x} \times 100$) untuk ketiga tag berkisar 0,35% hingga 0,67%, mengindikasikan pembacaan sensor yang sangat stabil pada kondisi operasi normal. Juga n pada id13 lebih kecil dari id1 dan id8 karena nilai anomali pasca-gap (28 siklus bernilai di atas 1 miliar) telah dikeluarkan dari perhitungan. Nilai smemerlukan kalkulasi langsung dari plc_dataGab.db sebelum submit final.

 ISSN:2302-8149
e-ISSN:2540-9719

4.1.4 Verifikasi Evolusi Skema Non-Destruktif

Pemeriksaan langsung pada plc_dataGab.db mengkonfirmasi bahwa tabel tb_log tidak memuat kolom id6 dan id7, meski kedua kolom tersebut pernah terdaftar di tb_tags. Tabel tb_tags mencatat 24 entri, sedangkan tb_log hanya memiliki 22 kolom id{N}. Tidak ada data historis yang hilang akibat penghapusan kedua tag dari konfigurasi. Ini mengkonfirmasi secara empiris bahwa operator σbekerja sebagai augmentasi murni tanpa operasi destruktif.

4.2 Pembahasan Kritis

4.2.1 Efektivitas Deteksi Protokol Otomatis

Keberhasilan akuisisi simultan dari dua protokol pada tiga site membuktikan bahwa fungsi δberbasis pola leksikal alamat bekerja secara deterministik tanpa kegagalan klasifikasi selama 6.001 siklus. Temuan ini konsisten dengan observasi Nikolajew & Eichelberger (2024) bahwa pola alamat Modbus TCP secara inheren dapat dibedakan dari protokol berbasis blok data -- meski karya tersebut tidak mengeksploitasi perbedaan ini untuk deteksi otomatis. Lacasa et al. (2025) mencapai integrasi multivendor serupa pada pabrik Ford, namun dengan konfigurasi per jenis PLC yang tertanam di kode; pendekatan pythonGab yang berbasis pola alamat lebih ringan secara implementasi dan tidak memerlukan pengetahuan vendor sebelumnya.

Celah yang tersisa pada mekanisme δadalah ketidakmampuan menangani alamat ambigu, yakni alamat numerik yang secara kebetulan valid untuk kedua protokol. Pada dataset plc_dataGab.db, tidak ditemukan kasus ambigu karena seluruh tag S7 menggunakan format DB*.* yang konsisten. Namun, sistem yang lebih generik perlu menambahkan lapisan validasi eksplisit pada tahap pembacaan konfigurasi.

4.2.2 Kegagalan Tag id12 sebagai Motivasi Validasi Tipe Data

Tag id12 (UI/Depok, alamat DB1.DBW16, tipe MF) mengalami kegagalan baca total dengan *zero-rate* 100%. Analisis struktural menunjukkan bahwa tipe MF (*float* 4-byte) memerlukan pembacaan dua Word berurutan, sedangkan alamat DBW16 hanya mengalamati satu Word (2-byte). *Mismatch* ini tidak terdeteksi pada tahap konfigurasi karena sistem belum memiliki validator tipe data sebelum koneksi dibuka. Temuan ini mempertegas bahwa deteksi protokol otomatis perlu dilengkapi dengan validasi konsistensi tipe data sebagai lapisan pemeriksaan kedua -- sebuah arah pengembangan yang konkret dan terukur untuk versi berikutnya.

4.2.3 Interpretasi Zero-rate per Protokol dan Site

Zero-rate yang berbeda antar site mencerminkan kondisi jaringan dan konfigurasi tag yang berbeda, bukan kelemahan inheren satu protokol dibanding yang lain. ISTN/Modbus TCP mencatat *zero-rate* 7,7%, yang sebagian besar disebabkan oleh dua gap komunikasi yang terekam dalam dataset. UI/S7comm mencatat 27,3% secara nominal, namun turun ke 9,1% setelah tag id12 dikeluarkan -- menunjukkan bahwa satu tag dengan *mismatch* tipe data mendominasi angka agregat secara tidak proporsional. UT/Serang mencatat *zero-rate* tertinggi (47,8%) pada 12 tag Modbus TCP, yang kemungkinan mencerminkan kondisi jaringan atau konfigurasi register yang perlu diverifikasi lebih lanjut (lihat J1 pada handoff).

Zero-rate UT/Serang sebesar 47,8% adalah angka yang perlu dibaca dengan hati-hati. Dua penjelasan teknis dapat diajukan, meski keduanya belum dapat diverifikasi dari data log yang tersedia. Penjelasan pertama bertumpu pada distribusi alamat register: 12 tag UT tersebar pada register 2 hingga 24 dengan jarak dua register per tag, sehingga setiap siklus Modbus harus menyelesaikan setidaknya enam transaksi terpisah alih-alih satu pembacaan blok kontigu. Zagan & Găitan (2022) sudah menunjukkan bahwa pola tersebar seperti ini memperpanjang total waktu akuisisi per siklus dan meningkatkan peluang *timeout*, terutama saat beban jaringan tidak stabil. Penjelasan kedua adalah kondisi koneksi fisik: UT/Serang berada di lokasi berbeda dari dua site lain, dan latensi jaringan yang lebih tinggi dapat mengakibatkan lebih banyak siklus berakhir tanpa nilai valid.

Persoalannya adalah log plc_dataGab.db tidak menyimpan *timestamp* per-PLC maupun kode status per transaksi Modbus, sehingga kedua hipotesis di atas tidak dapat dibedakan dari data yang ada. Satu langkah praktis untuk pengujian lanjutan adalah menambahkan kolom status_code per tag pada tb_log, yang akan memungkinkan pembedaan antara kegagalan *timeout* jaringan dan kegagalan karena PLC tidak merespons. Gap komunikasi 15,5 jam yang terekam pada 24 Mei 2026 juga belum dapat dikonfirmasi sebagai penghentian terjadwal atau kegagalan jaringan riil; sebagai catatan bahwa

 ISSN:2302-8149
e-ISSN:2540-9719

konfirmasi ini Adalah penting sebelum angka *zero-rate* UT/Serang dinyatakan sebagai indikator keandalan sistem dalam konteks yang lebih luas.

Zagan & Găitan (2022) membuktikan bahwa distribusi alamat register yang tersebar meningkatkan kemungkinan *timeout* pada sesi Modbus TCP. Pola *zero-rate* tinggi di UT/Serang konsisten dengan penjelasan ini, mengingat 12 tag tersebar pada rentang alamat yang lebih lebar dibanding 5 tag ISTN. Berbeda dari Grzesik & Mrozek (2020) yang mendokumentasikan SQLite sebagai basis data *edge* dengan keandalan penulisan tinggi, temuan ini mengkonfirmasi bahwa bottleneck pada sistem ini berada di lapisan komunikasi PLC, bukan di lapisan penyimpanan.

4.2.4 Konsistensi Interval Sampling dan Implikasinya

Modus interval 5 detik pada 73,5% siklus menunjukkan bahwa sistem beroperasi sesuai target nominal pada sebagian besar waktu pengujian. Interval 11 dan 12 detik (masing-masing 3,0%) mengindikasikan siklus yang terlewat, kemungkinan pada kondisi koneksi yang lambat atau *timeout* yang memperpanjang satu siklus menjadi dua kali lipat durasi nominal. Yamin et al. (2026) mengamati pola serupa pada sistem IoT industri, di mana variasi interval sampling berkorelasi dengan beban koneksi jaringan pada saat akuisisi berlangsung. Implikasi praktisnya adalah bahwa aplikasi hilir yang bergantung pada asumsi interval tetap perlu menyertakan logika interpolasi atau penandaan *gap* eksplisit sebelum analisis temporal dilakukan.

4.2.5 Posisi Kontribusi dan Keterbatasan Residual

Sistem *pythonGab* membuktikan bahwa akuisisi data PLC multivendor dua protokol dapat dioperasikan dari satu konfigurasi terpusat tanpa modifikasi kode antar vendor. Kontribusi ini mengisi celah yang diidentifikasi pada Tabel 3: tidak ada karya sebelumnya dalam pool referensi yang menggabungkan deteksi otomatis, konfigurasi eksternal berbasis berkas, dan uji lapangan tiga site sekaligus. Calderón et al. (2024) mendekati level abstraksi yang sama namun terbatas pada satu site laboratorium; Lacasa et al. (2025) mencapai skala lapangan nyata namun dengan konfigurasi kode per vendor.

Tiga keterbatasan residual perlu ditangani pada penelitian lanjutan. *Hardcode* SLAVE_ID, RACK, dan SLOT membatasi generalisasi ke PLC dengan konfigurasi non-standar. Latensi *end-to-end* per PLC belum terukur karena arsitektur log saat ini tidak mencatat *timestamp* per-PLC secara terpisah. Token Telegram yang ter-*expose* di kode sumber juga menjadi risiko keamanan yang harus diselesaikan sebelum sistem dipublikasikan sebagai lampiran artikel, sesuai standar keamanan OT yang ditetapkan NIST (2023) dan ISA (2025).

5 Kesimpulan

Penelitian ini merancang dan menguji pythonGab, sistem akuisisi data PLC multivendor yang mendeteksi protokol dari pola alamat tag, dengan hasil 22 tag aktif pada tiga site (ISTN/Jagakarsa, UI/Depok, UT/Serang) dari satu konfigurasi Excel. Deteksi protokol berjalan deterministik selama 6.001 siklus tanpa kegagalan klasifikasi antara Modbus TCP dan S7comm, dan tabel log tetap konsisten pada penomoran tag yang melompat dari id5 ke id8. Interval sampling 5 hingga 6 detik mencapai 91,5% dengan modus 5 detik (73,5%), sedangkan zero-rate sebesar 7,7% pada ISTN/Modbus TCP, 9,1% pada UI/S7comm setelah tag id12 dikeluarkan, dan 47,8% pada UT/Serang. Tag id12 gagal terbaca total akibat mismatch tipe data float 4-byte pada alamat Word 2-byte, sedangkan zero-rate UT/Serang yang tinggi belum dapat dijelaskan dari data log yang ada. Pengembangan lanjutan mencakup validasi tipe data otomatis, eksternalisasi SLAVE_ID, RACK, dan SLOT, lapisan keamanan komunikasi (NIST, 2023; ISA, 2025), serta pengujian minimal 30 hari dengan lebih dari dua vendor PLC.

Ucapan Terima Kasih

Penelitian ini terlaksana dengan dukungan pendanaan dari Program Hibah Kompetitif Internal ISTN Tahun Anggaran 2025/2026. Penulis menyampaikan apresiasi kepada LPPM ISTN atas dukungan dan fasilitasi yang diberikan selama proses penelitian, serta kepada dosen dan tenaga kependidikan FSTT ISTN yang telah berkontribusi dalam pelaksanaan pengujian dan evaluasi sistem.

 ISSN:2302-8149
e-ISSN:2540-9719

Referensi

[1] B. Babayigit & M. Abubaker. (2023). *Industrial Internet of Things: A Review of Improvements Over Traditional SCADA Systems for Industrial Automation*. *IEEE Systems Journal*, *18*(1), 120–133. https://doi.org/10.1109/JSYST.2023.3270620

[2] Z. Brahmia, F. Grandi, & B. Oliboni. (2024). *A Literature Review on Schema Evolution in Databases*. *Computing Open*. https://doi.org/10.1142/S2972370124300012

[3] A. Busboom. (2024). *Automated Generation of OPC UA Information Models: A Review and Outlook. Journal of Industrial Information Integration*, Article 100602. https://doi.org/10.1016/j.jii.2024.100602

[4] C. G. Calatrava, Y. B. Fontal, F. M. Cucchietti, & C. Diví-Cuesta. (2021). *NagareDB: A Resource-Efficient Document-Oriented Time-Series Database*. *Data*, *6*(8), Article 91. https://doi.org/10.3390/data6080091

[5] D. Calderón, F. J. Folgado, I. González, & A. J. Calderón. (2024). *Implementation and Experimental Application of Industrial IoT Architecture using Automation and IoT Hardware/Software. Sensors*, *24*(24), Article 8074. https://doi.org/10.3390/s24248074

[6] G. Cheng, Y. Lin, A. Abur, A. Gómez-Expósito, & W. Wu. (2024). *A Survey of Power System State Estimation using Multiple Data Sources: PMUs, SCADA, AMI, and Beyond*. *IEEE Transactions on Smart Grid*, *15*(1), 1129–1151. https://doi.org/10.1109/TSG.2023.3286401

[7] M. Fakih, R. Dharmaji, Y. Moghaddas, G. Quiros, O. Ogundare, & M. A. Al Faruque. (2024). *LLM4PLC: Harnessing Large Language Models for Verifiable Programming of PLCs in Industrial Control Systems*. Dalam *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP '24)* (hlm. 192–203). Association for Computing Machinery. https://doi.org/10.1145/3639477.3639743

[8] M. Febriansyah & R. M. Dinata. (2026). Pemodelan *Graf Virtual* Energi dari Data IoT Sekuensial untuk Gedung Cerdas. *Sistemasi: Jurnal Sistem Informasi*, *15*(3), 795–807. https://doi.org/10.32520/stmsi.v15i3.5714

[9] S. Figueroa-Lorenzo, J. Añorga, & S. Arrizabalaga. (2019). *A Role-based Access Control Model in Modbus SCADA Systems: A Centralized Model Approach*. *Sensors*, *19*(20), Article 4455. https://doi.org/10.3390/s19204455

[10] P. Grzesik & D. Mrozek. (2020). Comparative analysis of time series databases in the context of edge computing for low power sensor networks. Dalam *Computational Science: ICCS 2020*. Springer. https://doi.org/10.1007/978-3-030-50426-7_28

[11] W. He, M. J. A. Baig, & M. T. Iqbal. (2025). *An Internet of Things–Supervisory Control and Data Acquisition (IoT-SCADA) Architecture for Photovoltaic System Monitoring, Control, and Inspection in Real Time*. *Electronics*, *14*(1), Article 42. https://doi.org/10.3390/electronics14010042

[12] International Society of Automation. (2025). *ANSI/ISA-62443-2-1-2024: Security for Industrial Automation and Control Systems: Part 2-1*. https://www.isa.org/standards-and-publications/isa-standards/isa-iec-62443-series-of-standards

[13] F. Katulić, D. Sumina, S. Groš, & I. Erceg. (2023). *Protecting Modbus/TCP-based Industrial Automation and Control Systems using Message Authentication Codes*. *IEEE Access*, *11*, 47007–47023. https://doi.org/10.1109/ACCESS.2023.3275443

[14] A. Lacasa, J. Llopis, N. Montés, I. Peinado-Asensi, & E. Garcia. (2025). *Cross-PLC: An I3oT Cross Platform to Manage Communications for Applications in Real Factories*. *Sensors*, *25*(10), Article 2973. https://doi.org/10.3390/s25102973

[15] M. Ladegourdie & J. Kua. (2022). *Performance Analysis of OPC UA for Industrial Interoperability Towards Industry 4.0. IoT*, *3*(4), 507–525. https://doi.org/10.3390/iot3040027

[16] S. Liu. (2023). *Design and Implementation of a Lightweight Security-Enhanced Scheme for Modbus TCP Protocol. Security and Communication Networks*, *2023*, Article 5486566. https://doi.org/10.1155/2023/5486566

[17] T. Martins & S. V. G. Oliveira. (2022). *Enhanced Modbus/TCP Security Protocol: Authentication and Authorization Functions Supported*. *Sensors*, *22*(20), Article 8024. https://doi.org/10.3390/s22208024

F. E. Mustafa et al. (2023). *A Review on Effective Alarm Management Systems for Industrial Process*

 ISSN:2302-8149
e-ISSN:2540-9719

Control: Barriers and Opportunities. *International Journal of Critical Infrastructure Protection*, *41*, Article 100599. https://doi.org/10.1016/j.ijcip.2023.100599

[18] National Institute of Standards and Technology. (2023). *Guide to Operational Technology (OT) Security (NIST SP 800-82 Rev. 3)*. https://doi.org/10.6028/NIST.SP.800-82r3

[19] C. Nikolajew & H. Eichelberger. (2024). *Industry 4.0 Connectors: A Performance Experiment with Modbus/TCP*. *arXiv preprint*, arXiv:2410.15813. https://doi.org/10.5281/zenodo.13902632

[20] D. Pliatsios, P. G. Sarigiannidis, T. Lagkas, & A. Sarigiannidis. (2020). *A Survey on SCADA Systems: Secure Protocols, Incidents, Threats and Tactics. IEEE Communications Surveys and Tutorials*, *22*(3), 1942–1976. https://doi.org/10.1109/COMST.2020.2987688

[21] O. Rashad, O. Attallah, & I. Morsi. (2022). *A Smart PLC-SCADA Framework for Monitoring Petroleum Products Terminals in Industry 4.0 via Machine Learning*. *Measurement and Control*. https://doi.org/10.1177/00202940221103305

[22] M. Šverko, T. Galinac Grbac, & M. Mikuc. (2022). *SCADA Systems with Focus on Continuous Manufacturing and Steel Industry: A Survey on Architectures, Standards, Challenges and Industry 5.0. IEEE Access*, *10*, 109395–109430. https://doi.org/10.1109/ACCESS.2022.3211288

[23] H. Trifonov & D. Heffernan. (2023). *OPC UA TSN: A Next-Generation Network for Industry 4.0 and IIoT*. *International Journal of Pervasive Computing and Communications*, *19*(3), 386–411. https://doi.org/10.1108/IJPCC-07-2021-0160

[24] C. Wang, J. Qiao, X. Huang, S. Song, H. Hou, T. Jiang, L. Rui, J. Wang, & J. Sun. (2023). *Apache IoTDB: A Time Series Database for IoT Applications. Proceedings of the ACM on Management of Data*, *1*(2), Article 195. https://doi.org/10.1145/3589775

[25] Y. Wang, D. Peng, E. Qi, & H. Wang. (2022). *Modbus/TCP Data Acquisition System based on Industrial Ethernet Technology*. Dalam *2022 4th International Conference on Electrical Engineering and Control Technologies (CEECT)* (hlm. 746–750). IEEE. https://doi.org/10.1109/CEECT55960.2022.10030131

[26] M. I. Yamin, A. Ariman, R. M. Dinata, & N. Purnomo. (2026). *Comparative Performance Evaluation of MQTT, WebSocket, and Firebase on ESP32-C3 for Real-Time IoT Communication*. *JATI: Jurnal Mahasiswa Teknik Informatika*, *10*(3), 3917–3923. https://doi.org/10.36040/jati.v10i3.18056

[27] I. Zagan & V. G. Găitan. (2021). *Experimental Implementation and Performance Evaluation of an IoT Access Gateway for the Modbus Extension*. *Sensors*, *21*(1), Article 246. https://doi.org/10.3390/s21010246

[28] I. Zagan & V. G. Găitan. (2022). *Enhancing the Modbus Communication Protocol to Minimize Acquisition Times based on an STM32-Embedded Device*. *Mathematics*, *10*(24), Article 4686. https://doi.org/10.3390/math10244686