SISTEM INFORMASI DASHBOARD MONITORING C/N PADA PRODUK SCPC DI PT. PI
[*DASHBOARD INFORMATION SYSTEM FOR C/N MONITORING ON SCPC PRODUCTS AT PT. PI*]

Chuan Pratama[1], Junita[2*]
[1,2]Program Studi Teknik Elektro, Universitas Pelita Harapan, Indonesia
*Korespondensi penulis: junita.fti@uph.edu

ABSTRACT

SCPC (Signal Carrier Per Carrier) is a priority VSAT service that requires continuous frequency signal monitoring to ensure network stability. Currently, this monitoring is performed using a web-based decimator device. However, the monitoring data still requires manual processing to create reports and analyses, which is inefficient and time-consuming. This study developed a web-based dashboard monitoring system that automates the processing of C/N (Carrier to Noise) data for SCPC products at PT. PI. The system was tested using a task scheduler for 14 hours with 30-minute intervals, recording a longest processing time of 16.665 seconds with 11,812 data points, 60.6% RAM usage, and 36.9% CPU usage, and a shortest processing time of 4.026 seconds with 3,546 data points, 53.9% RAM usage, and 39.7% CPU usage. The proposed web application successfully displays historical graphs, total remote counts, databases, top 10 under C/N, tables, and reports, as initially designed. The average dashboard loading time to complete data display is approximately 9.894 seconds. This system is expected to improve the efficiency of SCPC monitoring and provide faster, more accurate reporting.
Keywords : *C/N; dashboard; database; python; spectrum analyzer*

ABSTRAK

SCPC (*Signal Carrier Per Carrier*) adalah layanan VSAT prioritas yang memerlukan monitoring sinyal frekuensi secara terus-menerus untuk memastikan kestabilan jaringan. Saat ini, monitoring dilakukan melalui perangkat decimator berbasis web; namun, pengolahan data monitoring menjadi laporan dan analisis masih harus dilakukan secara manual, yang tidak efisien dan memakan waktu. Penelitian ini mengembangkan sistem dashboard monitoring berbasis web yang mampu mengotomatisasi pengolahan data C/N (*Carrier to Noise*) pada produk SCPC di PT. PI. Sistem ini diuji menggunakan task scheduler selama 14 jam dengan interval 30 menit, di mana waktu proses terlama tercatat 16,665 detik dengan 11.812 data, penggunaan RAM 60,6%, dan CPU 36,9%; sementara waktu tercepat adalah 4,026 detik dengan 3.546 data, penggunaan RAM 53,9%, dan CPU 39,7%. Aplikasi web berhasil menampilkan grafik historis, total remote, basis data, top 10 di bawah C/N, tabel, dan laporan sesuai desain awal. Waktu rata-rata yang dibutuhkan dashboard untuk memuat data hingga selesai adalah sekitar 9,894 detik. Pengembangan sistem ini diharapkan dapat meningkatkan efisiensi monitoring SCPC dan memberikan laporan yang lebih cepat dan akurat.
Kata kunci : basis data; C/N; dasbor; penganalisis spektrum; python

e-ISSN 2598-9596

PENDAHULUAN

SCPC (*Signal Carrier Per Carrier*) adalah teknologi yang digunakan untuk menyediakan layanan VSAT (*Very Small Aperture Terminal*) prioritas bagi pelanggan. Setiap pelanggan SCPC mendapatkan sinyal frekuensi tersendiri sehingga menghindari interferensi dari sinyal lainnya. Karena layanan ini bersifat prioritas, monitoring sinyal secara terus-menerus sangat penting untuk memastikan kualitas layanan tetap stabil.

Di PT. PI, monitoring SCPC saat ini dilakukan menggunakan sistem *Carrier Monitoring System* (CMS), yang berbasis spektrum digital bernama decimator. Decimator merupakan perangkat yang dapat memantau spektrum frekuensi dan terhubung ke komputer melalui jaringan LAN. Antarmukanya berbasis web sehingga dapat diakses oleh beberapa pengguna dari berbagai perangkat melalui web browser.

Meskipun decimator memiliki banyak keunggulan dibandingkan perangkat monitoring sebelumnya, ada beberapa keterbatasan dalam sistem saat ini. Data yang dikumpulkan oleh decimator tidak langsung diolah menjadi laporan atau analisis monitoring yang siap digunakan. Pengolahan data masih dilakukan secara manual oleh tim teknis, yang memakan waktu dan rentan terhadap kesalahan. Selain itu, decimator tidak menyediakan fitur untuk memvisualisasikan data dalam bentuk grafik atau tabel yang memudahkan analisis.

Untuk mengatasi masalah ini, penelitian ini bertujuan untuk mengembangkan sistem informasi *dashboard* berbasis web yang mampu mengotomatisasi proses pengumpulan dan pengolahan data monitoring C/N. *Dashboard* ini juga akan menampilkan hasil monitoring dalam bentuk yang mudah dipahami, sehingga pengguna dapat melihat performa SCPC secara real-time dan melakukan analisis lebih cepat. Dengan sistem ini, diharapkan proses monitoring dapat menjadi lebih efisien.

BAHAN DAN METODE

Bahan dan Alat

Spectrum analyzer adalah perangkat yang digunakan untuk memantau sinyal frekuensi pada berbagai band. Salah satu pengembangan terbaru dari perangkat ini adalah decimator, yang merupakan *spektrum analyzer* digital. Decimator tidak memiliki antarmuka fisik seperti tombol atau layar, namun dapat dioperasikan sepenuhnya melalui *web interface.* Kelebihan decimator adalah kemampuannya untuk memantau hingga 100 frekuensi sekaligus melalui 8 port

e-ISSN 2598-9596

input, serta kemampuan untuk memantau frekuensi pada rentang 5 MHz hingga 6,5 GHz (Calian, n.d.).

Gambar 1. *Spectrum Analyzer* Analog (kiri) Decimator (kanan)

Python merupakan bahasa pemrograman yang semakin populer dalam pengembangan perangkat lunak karena kesederhanaan sintaksis dan kekuatan dalam pengolahan data. Python tidak memerlukan memori yang besar saat dijalankan sehingga meringankan beban komputer (Harismawan, 2017). Python memiliki banyak *library* yang mendukung analisis data (Bogdanchikov et al., 2013), salah satunya adalah Pandas. *Library* ini memungkinkan pengolahan data terstruktur dengan lebih mudah, termasuk penggabungan data dari berbagai sumber. *Library* Pandas terdiri atas dua struktur data yaitu Series, yang memiliki satu dimensi struktur data, dan DataFrame, yang memiliki dua dimensi data yang lebih kompleks (Nelli, 2018). *Library* ini memiliki beberapa tools. Dua diantaranya adalah *tools* untuk melakukan filter dan groupby (McKinney dan Pandas Development Team, 2015) yang akan digunakan pada program. Selain Pandas, *library* lain yang digunakan adalah OS (python.org, 2022), Pathlib (Python Documentation, 2019) dan Datetime (Python Software Foundation, 2020). Python mendukung pengembangan aplikasi web melalui framework seperti Dashplotly, yang memungkinkan pembuatan antarmuka pengguna berbasis web untuk menampilkan hasil analisis data.

Pengembangan sistem yang dibuat memerlukan *Database Managemnet Systems* (DBMS). Model database yang popular hingga saat ini adalah SQL (*Structure Query Language*) **(**Watt dan Eng, 2012). Selain SQL ada juga yang disebut dengan NoSQL *(Not only SQL*) database, yang termasuk ke dalam *database management system non-relational*. CouchDB adalah sistem manajemen database non-relasional yang sering digunakan dalam aplikasi web karena fleksibilitasnya dalam menyimpan data semi-terstruktur. CouchDB menggunakan format JSON untuk menyimpan data, yang memudahkan integrasi dengan aplikasi berbasis web. Namun, karena sifatnya yang hanya menyimpan data sementara, data dari CouchDB perlu direplikasi ke penyimpanan permanen, seperti MySQL, untuk keperluan pengolahan data lebih

lanjut. MySQL adalah sistem manajemen database relasional yang sangat populer karena kecepatannya dan kemudahan penggunaannya dalam mengelola data terstruktur (MySQL, n.d.)

Dashplotly adalah framework pengembangan aplikasi web berbasis Python yang dirancang untuk menyajikan data dalam bentuk grafik dan tabel interaktif. Dengan Dashplotly, pengguna dapat membuat dashboard yang menampilkan data secara dinamis dan real-time, sehingga sangat cocok digunakan dalam aplikasi monitoring seperti SCPC. Web-app ini dapat diakses dari berbagai perangkat tanpa memerlukan instalasi aplikasi khusus, hanya melalui browser web.

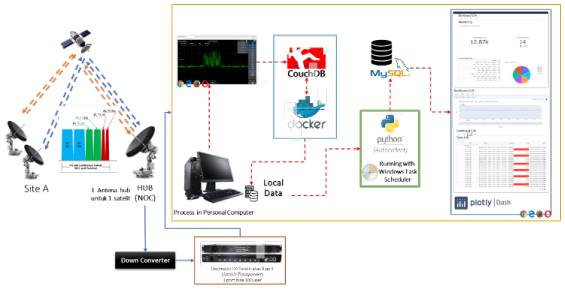
Metode Penelitian

Penelitian ini berfokus pada pengembangan sistem monitoring berbasis web yang mampu mengotomatisasi proses pengumpulan dan pengolahan data dari decimator.

Arsitektur sistem yang dirancang terdiri dari beberapa komponen utama, yaitu decimator, database CouchDB untuk penyimpanan sementara, Python untuk pengolahan data, MySQL untuk penyimpanan data permanen, dan Web-app Dashplotly untuk visualisasi data. Decimator akan mengambil sinyal *carrier* dari antena satelit, yang kemudian disimpan sementara dalam CouchDB. Data ini kemudian diambil dan digabungkan secara otomatis oleh program *autocollect* berbasis Python, yang menggunakan *library* Pandas dan Pathlib untuk mengelola file.

Data yang telah diproses kemudian disimpan dalam MySQL, dan dari sini Web-app akan mengambil data untuk ditampilkan dalam bentuk grafik historis dan tabel top 10 frekuensi dengan nilai C/N terendah.

Program *autocollect* dikembangkan menggunakan *library* Pathlib dan OS dari Python, yang memungkinkan sistem untuk secara otomatis mengambil file CSV dari berbagai folder yang dibuat oleh decimator dan menggabungkannya menjadi satu file. *Library* Pandas digunakan untuk memproses dan membersihkan data, serta menambahkan header yang sesuai pada file CSV.



Gambar 2. Arsitektur Sistem

Pengujian dilakukan menggunakan dua perangkat sistem untuk mengukur kinerja program *autocollect* dan

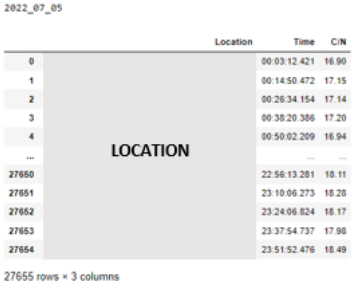
 e-ISSN 2598-9596

dashboard. *Task scheduler* digunakan untuk mengotomatisasi proses pengambilan data dengan interval waktu 30 menit selama 14 jam. Pengujian ini juga melibatkan pengukuran penggunaan RAM dan CPU pada setiap proses.

HASIL DAN PEMBAHASAN

4.1. Hasil Pengujian Program *Autocollect*

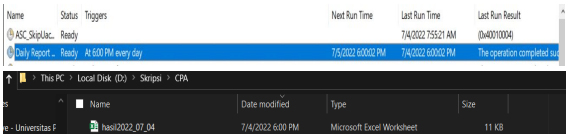
Hasil pengujian menunjukkan bahwa program *autocollect* mampu menggabungkan data dari berbagai folder secara otomatis dengan akurasi yang tinggi. Pada pengujian, data dari decimator dengan total 27.655 baris berhasil digabungkan menjadi satu file CSV dengan tiga kolom utama: lokasi, waktu, dan nilai C/N. Program juga mampu menambahkan header pada file dengan benar.



Gambar 3. Hasil Penggabungan Data dari Program *Autocollect*

4.2. Hasil Pengujian *Task Scheduler*

Pengujian *task scheduler* dilakukan selama selama 14 jam dengan interval 30 menit.



Gambar 4. Konfigurasi *Task Scheduler* (atas), Hasil *Task Scheduler* (bawah)

Task Scheduler dikonfigurasi untuk secara otomatis menjalankan file Python setiap hari pukul 18:00. Pada gambar hasil otomatisasi pengambilan data memakai *Task Scheduler*, yang ada pada Gambar 4, keterangan pada kolom bagian *date modified* menunjukkan kesesuaian dengan waktu yang telah diatur yakni pukul 18:00.

Tabel 1. Hasil Pengujan dengan Interval 30 menit

waktu test 14 jam 30 menit

time_get	time_process	data	CPU	RAM	Interval
9:26	13.5841682	2979	42.4	53.8	0:29
9:56	4.026174	3546	39.7	53.9	0:30
10:26	10.8173766	4095	34	54.2	0:29
10:56	7.5682983	4639	41.8	55	0:30
11:26	9.8548787	5165	38.5	54.9	0:30
11:56	9.8564696	5673	42.1	54.4	0:29
12:26	7.1166253	6168	43.5	54	0:30
12:56	6.833197	6675	39.6	54.3	0:30
13:26	11.6263164	7211	35.1	55.2	0:30
13:56	11.0870575	7741	36.8	55.7	0:30
14:26	13.8667318	8267	34.4	56.4	0:29
14:56	11.215866	8748	41.3	56.7	0:29
15:26	10.8043743	9230	36.7	56.9	0:29
15:56	7.8550793	9779	38.4	58	0:30
16:26	10.4269236	10293	36.8	58.7	0:29
16:56	7.6828604	10805	39.6	60.1	0:30
17:26	12.0854312	11308	33.9	60.5	0:30
17:56	16.6651535	11812	36.9	60.6	0:29
18:26	9.3524381	12308	35.9	61.2	0:30
18:56	13.7559164	12801	37.5	62.4	0:29
19:26	10.0166898	13290	35.8	62.7	0:29
19:56	6.8734969	13776	31.2	63	0:30
20:26	12.1933465	14266	36.6	62.6	0:29
20:56	11.268912	14715	38.8	62.1	0:29
21:26	9.3482759	15176	32.8	63.4	0:30
21:56	12.1843485	15652	41.8	63.3	0:30
22:26	15.652438	16099	40.5	62.7	0:29
22:56	11.7399967	16604	39	63.9	0:30
23:26	14.1665321	17068	38.3	63.2	0:30
23:56	13.7968686	17512	40.7	63.5	
max	16.6651535	17512	43.5	63.9	
min	4.026174	2979	31.2	53.8	
avrg	10.77740804	10446.7	38.01333	58.91	

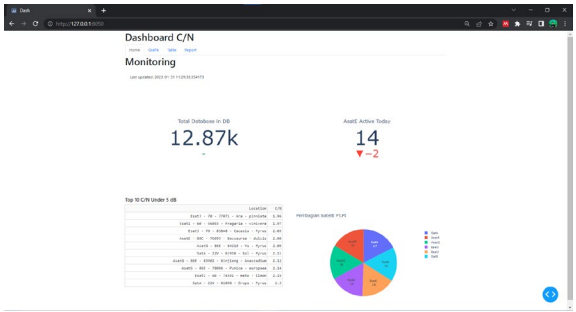
Dari Tabel 1 terlihat bahwa selama 14 jam pengujian dengan interval waktu 30 menit, program mengolah data dengan nilai waktu terlama yaitu 16,665 detik pada

e-ISSN 2598-9596

kondisi data yang diambil sebanyak 11.812, dengan RAM *usage* pada perangkat sistem 60,6% dan CPU *usage* 36,9%. Sedangkan waktu tercepat adalah 4,026174 detik pada kondisi data yang diambil sebanyak 3.546, dengan RAM *usage* pada perangkat sistem 53,9% dan CPU *usage* 39,7%.

4.3. Hasil Pengujian *Web-App*

Dashboard monitoring yang dikembangkan berhasil menampilkan data monitoring dalam bentuk grafik historis dan tabel top 10 frekuensi dengan nilai C/N terendah. Pengujian menunjukkan bahwa *dashboard* dapat memuat data dengan rata-rata waktu 9,894 detik. Hasil visualisasi data dalam bentuk grafik memudahkan pengguna untuk melihat tren perubahan nilai C/N dari waktu ke waktu, sementara tabel top 10 membantu dalam mengidentifikasi frekuensi dengan masalah potensial.



Gambar 5. Tampilan *Web-app* pada Menu *Home*

KESIMPULAN

Penelitian ini berhasil mengembangkan sistem informasi *dashboard* berbasis web yang mampu mengotomatisasi pengolahan dan visualisasi data monitoring C/N pada produk SCPC di PT. PI. Program *autocollect* yang dirancang berhasil menggabungkan data secara otomatis dari decimator dan mengirimnya ke database MySQL untuk keperluan visualisasi di dashboard. Dari pengujian selama 14 jam dengan interval waktu 30 menit, didapatkan waktu terlama untuk mengolah data adalah selama 16,665 detik untuk 11.812 data, dengan RAM *usage* 60,6% dan CPU *usage* 36,9%. Sedangkan waktu tercepat adalah 4,026174 detik untuk 3.546 data, dengan RAM *usage* 53,9% dan CPU *usage* 39,7%. Pengujian pada *Web-App* menunjukkan bahwa *dashboard* dapat memuat data dengan rata-rata waktu 9,894 detik.

Dengan sistem ini, tim monitoring di PT. PI dapat melakukan analisis data dengan lebih cepat dan akurat.

DAFTAR PUSTAKA

Bogdanchikov, A., Zhaparov, M., & Suliyev, R. (2013). Python to learn programming. *Journal of Physics: Conference Series, 423*(1). https://doi.org/10.1088/1742-6596/423/1/012027

Calian. (n.d.). *Decimator D4 RF spectrum and signal analyzer*. Retrieved from https://www.calian.com/products/decimator-d4/

Harismawan, A. F. (2017). Analisis perbandingan performa web service

menggunakan bahasa pemrograman Python, PHP. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2*(1), 237–245. Retrieved from https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/781

McKinney, W., & Pandas Development Team. (2015). Pandas - Powerful Python data analysis toolkit. *Pandas - Powerful Python Data Analysis Toolkit*, 1625.

MySQL. (n.d.). What is MySQL? *MySQL 8.0 Reference Manual*. Retrieved from https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html

Nelli, F. (2018). *Python data analytics*. https://doi.org/10.1007/978-1-4842-3913-1

Python Documentation. (2019). Pathlib - Object-oriented filesystem paths. Retrieved from https://docs.python.org/3/library/Pathlib.html

Python Software Foundation. (2020). Datetime — Basic date and time types. *Python*. Retrieved from https://docs.python.org/3/library/datetime.html

python.org. (2022). os — Miscellaneous operating system interfaces — Python 3.10.4 documentation. Retrieved from https://docs.python.org/3/library/os.html

Watt, A., & Eng, N. (2012). *Database design* (2nd ed.). Retrieved from http://open.bccampus.ca