Rencana Penelitian Grafika Komputer

Topik: Temporal Code City untuk Visualisasi Evolusi Microservices Berbasis Git History

Anggota:

- Christopher Gijoh (01082240011)
- Evan Laluan (01082240015)
- Christian Oroh (01082240019)

1. Tabel Perbandingan Ringkas Jurnal Ilmiah

NO	JUDUL & TAHUN	PENULIS	FOKUS UTAMA	TEMUAN UTAMA
1	**Visualizing Evolving Software Cities (2020)**	Pfahler, F., et al.	Evolusi visual dalam metafora kota	Mengembangkan konsep di mana "evolusi" adalah warga kelas satu, mengatasi masalah pergerakan gedung yang kacau saat update versi.
2	**Trace Visualization within the Software City Metaphor... (2021)**	Dashuber, V. & Philippsen, M.	Visualisasi Microservices & Tracing	*DynaCity* memvisualisasikan ketergantungan dinamis antar-service sebagai busur (arcs) di atas kota, meningkatkan pemahaman arsitektur hingga 11.7%.
3	**Visual Integration of Static and Dynamic Software Analysis... (2024)**	Krause-Glau, A., et al.	Integrasi Git Workflow & Web-based Viz	Mengintegrasikan visualisasi kota langsung ke dalam *Git hosting services* untuk code review, menggabungkan analisis statis dan dinamis.
4	**From Static Code Analysis to Visual Models of Microservice Architecture (2024)**	Cerný, T., et al.	Analisis Statis untuk Microservices	Menunjukkan bahwa tampilan arsitektur holistik dari sistem cloud-native (Microservices) dapat diturunkan secara efektif hanya dari analisis kode statis.
5	**Githru: Visual Analytics for Understanding Software Development History... (2020)**	Kim, Y., et al.	Visualisasi Metadata Git	Mengusulkan teknik abstraksi graph untuk menangani kompleksitas data Git commit yang masif, yang menjadi basis data utama untuk visualisasi temporal.

2. Ringkasan Kritis dan Analisis Jurnal

Jurnal 1: Visualizing Evolving Software Cities

- **Referensi:** Pfahler, F., Minelli, R., Nagy, C., & Lanza, M. (2020). *2020 Working Conference on Software Visualization (VISSOFT)*.
- **Tujuan:** Mengatasi kelemahan metafora kota tradisional yang statis, di mana perubahan antar versi seringkali menyebabkan pergeseran layout gedung yang membingungkan (*unpredictable layout stability*).
- **Metodologi:** Penulis mengembangkan *m3triCity*, sebuah platform berbasis web yang memperlakukan "waktu" sebagai dimensi utama. Mereka menggunakan algoritma layout yang mempertahankan posisi gedung (kelas) yang tidak berubah dan memvisualisasikan refactoring secara eksplisit.
- **Temuan Utama:** Pendekatan ini memungkinkan *smooth transition* antar revisi kode. Pengguna tidak kehilangan orientasi spasial saat melihat evolusi software dari waktu ke waktu.
- **Relevansi:** Sangat krusial untuk aspek **"Temporal"** dalam riset ini. Jika ingin membuat animasi "gedung tumbuh" berdasarkan Git Commit, harus menggunakan teknik *layout stability* yang dibahas di sini agar kota tidak "berantakan" setiap kali ada commit baru.

Jurnal 2: Trace Visualization within the Software City Metaphor

- **Referensi:** Dashuber, V., & Philippsen, M. (2021). *2020 Working Conference on Software Visualization (VISSOFT)*.
- **Tujuan:** Memvisualisasikan perilaku dinamis dan ketergantungan pada arsitektur Microservices yang seringkali tersembunyi di balik komunikasi jaringan.
- **Metodologi:** Memperkenalkan *DynaCity*, yang menggunakan metafora kota namun menambahkan "arcs" (busur cahaya) antar gedung untuk merepresentasikan *runtime traces* (komunikasi antar service). Kecerahan busur menunjukkan intensitas trafik.
- **Temuan Utama:** Eksperimen terkontrol menunjukkan bahwa developer profesional dapat menyelesaikan tugas pemahaman software 5.84% lebih cepat dan 11.7% lebih akurat dibandingkan metode visualisasi *trace* tradisional.
- **Relevansi:** Ini adalah fondasi untuk aspek **"Microservices"**. Karena Microservices terpecah-pecah, sekadar menggambar gedung tidak cukup. Perlu mengadopsi konsep "visualisasi koneksi/dependensi" (seperti busur/jembatan) untuk menunjukkan hubungan antar *services* yang terpisah repo.

Jurnal 3: Visual Integration of Static and Dynamic Software Analysis in Code Reviews

- **Referensi:** Krause-Glau, A., Damerau, L., Hansen, M., & Hasselbring, W. (2024). *2024 IEEE Working Conference on Software Visualization (VISSOFT)*.
- **Tujuan:** Membawa visualisasi software keluar dari aplikasi *standalone* dan masuk ke dalam *workflow* sehari-hari developer (seperti saat melakukan Pull Request di GitHub/GitLab).
- **Metodologi:** Membangun ekstensi berbasis web pada alat *ExplorViz* yang terintegrasi dengan layanan hosting Git. Alat ini menggabungkan analisis perubahan struktur kode (statis) dengan dampak runtime (dinamis).
- **Temuan Utama:** Integrasi langsung ke Git hosting menghilangkan hambatan "setup data manual" yang sering membuat visualisasi gagal diadopsi di industri.
- **Relevansi:** Jurnal ini sangat baru (2024) dan mendukung argumen tentang pentingnya integrasi **Git**. Ini memberi landasan bahwa visualisasi harus mudah diakses (idealnya web-based/WebGL) dan bisa membaca data langsung dari repositori Git, bukan input manual.

Jurnal 4: From Static Code Analysis to Visual Models of Microservice Architecture

- **Referensi:** Cerný, T., Abdelfattah, A. S., Yero, J., & Taibi, D. (2024). *Cluster Computing*.
- **Tujuan:** Menjawab tantangan degradasi arsitektur pada sistem cloud-native berskala besar dengan cara mengekstraksi tampilan arsitektur secara otomatis.
- **Metodologi:** Menggunakan *Static Code Analysis* (analisis kode tanpa menjalankannya) untuk mendeteksi dependensi antar microservices dan membangun model visual.
- **Temuan Utama:** Pandangan holistik (menyeluruh) dari sistem Microservices dapat dibangun secara efektif dari *codebase*. Ini membantah anggapan bahwa Microservices hanya bisa dipahami lewat *dynamic tracing*.
- **Relevansi:** Penelitian akan menggunakan **Git Log/Source Code** (data statis/historis), bukan memonitor aplikasi yang sedang berjalan (data dinamis/live server). Jurnal ini memvalidasi bahwa pendekatan (menggunakan data statis dari Git untuk memetakan arsitektur Microservice) adalah metode yang valid secara ilmiah.

Jurnal 5: Githru: Visual Analytics for Understanding Software Development History

- **Referensi:** Kim, Y., et al. (2020). *IEEE Transactions on Visualization and Computer Graphics*.
- **Tujuan:** Membantu developer memahami sejarah pengembangan proyek yang memiliki ribuan commit dan cabang (branch) yang rumit.

- **Metodologi:** Penulis membuat *Githru*, sebuah sistem analitik visual yang melakukan rekonstruksi graph, *clustering*, dan *squash merge* pada metadata Git untuk menyederhanakan tampilan sejarah yang kompleks.
- **Temuan Utama:** Alat visualisasi Git yang ada sebelumnya tidak skalabel. Teknik abstraksi data (pengelompokan commit) sangat diperlukan sebelum data divisualisasikan.
- **Relevansi:** Meskipun *Githru* adalah visualisasi 2D (bukan 3D City), jurnal ini penting untuk **Backend Logic**. Saat membuat animasi pertumbuhan kota, tidak bisa merender *setiap* commit satu per satu (terlalu banyak). Perlu menggunakan teknik *clustering* atau *sampling* seperti yang dibahas di jurnal ini agar animasi evolusi kota berjalan mulus dan bermakna.

3. Refleksi Celah Penelitian (Research Gap)

1. **Status Quo:**
    - Visualisasi evolusi (*Time-based*) sudah ada (Pfahler et al., 2020), namun seringkali fokus pada sistem Monolith berorientasi objek.
    - Visualisasi Microservices sudah ada (Dashuber et al., 2021; Cerný et al., 2024), namun sebagian besar bersifat "snapshot" (statis pada satu waktu) atau fokus pada *tracing* performa runtime, bukan evolusi strukturnya.
2. **Masalah (Gap):**
    - Belum ada metode visualisasi yang secara efektif menggabungkan **metafora kota**, **struktur Microservices yang terdistribusi**, dan **dimensi waktu (Git History)** secara bersamaan dalam satu tampilan *real-time*.
    - Developer sulit melihat "Health History" sebuah ekosistem Microservices. Misalnya: *"Service A membengkak (god class) sejak commit bulan lalu, dan ini menyebabkan Service B jadi sering berubah."* Narasi kausalitas ini hilang dalam tool yang ada.
3. **Rencana Judul/Topik Tesis (Kontribusi Kami):**
    - *Judul Usulan:* **"Visualisasi Evolusi Arsitektur Microservices Berbasis Metafora Kota Temporal Menggunakan Data Riwayat Git"**
    - *Kontribusi Teknis:* Membuat purwarupa (prototype) berbasis WebGL yang membaca sekumpulan repositori Git (multi-repo), lalu merender animasi kota di mana:
        - **Distrik** = Service/Repository.

- **Gedung** = File/Class.
- **Tinggi Gedung** = Lines of Code (tumbuh saat commit tambah baris).
- **Warna/Efek** = Recency (menyala merah jika baru saja di-commit/bug fix).
- **Kabel/Jalan** = Dependensi antar service yang muncul/hilang seiring waktu.