INNOVATIVE: Journal Of Social Science Research
Volume 4 Nomor 6 Tahun 2024 Page 9290-9305
E-ISSN 2807-4238 and P-ISSN 2807-4246
Website: https://j-innovative.org/index.php/Innovative

Web-Based Simulation System in Formulating Sustainable Replanting

Muhammad Dominique Mendoza[1✉], Fevi Rahmawati Suwanto[2], Ishaq Matondang[3], Olnes Yosefa Hutajulu[4], Reni Rahmadani[5], Ressy Dwitias Sari[6]
(1,2,3,4,5) Universitas Negeri Medan, (6) Universitas Negeri Jakarta
Email: aenaen@unimed.ac.id[1✉]

Abstrak

Simulasi berbasis web harus mencakup kemampuan untuk mendistribusikan dan menggabungkan model dan hasil simulasi dengan mudah, mirip dengan aksesibilitas dan kemampuan penyusunan publikasi web kontemporer. Kehadiran konten web yang dapat dieksekusi, kapasitas untuk portabel secara universal, pemanfaatan teknologi komponen, dan ketersediaan paket standar tingkat tinggi untuk akses database dan pembuatan antarmuka pengguna grafis merupakan faktor penting yang memfasilitasi implementasi simulasi berbasis web. Pemanfaatan perangkat lunak berbasis komponen memungkinkan penciptaan lingkungan simulasi yang sangat modular, memfasilitasi penggunaan ulang komponen perangkat lunak secara ekstensif. Pengaturan simulasi menghadapi tekanan yang semakin meningkat karena sifat simulasi berbasis web yang luas. Penggabungan teknologi komponen dalam lingkungan simulasi berbasis web memungkinkan perlakuan model simulasi sebagai komponen individual, yang dapat dirakit secara fleksibel untuk membangun model. Selain itu, simulasi berbasis web ini memfasilitasi hubungan dinamis antara input dan output simulasi dengan sistem basis data, sehingga menyediakan sarana yang nyaman dan mudah beradaptasi untuk menyimpan hasil simulasi.

Kata Kunci: *Java, Penanaman Kembali, Berkelanjutan, Simulasi Berbasis Web*

Abstract

The web-based simulation should include the capability to easily distribute and combine simulation models and outcomes, similar to the accessibility and composability of contemporary web publications. The presence of executable web content, the capacity to be universally portable, the utilisation of component technology, and the availability of standard high-level packages for database access and graphical user interface creation are crucial factors that facilitate the implementation of web-based simulation. The utilisation of component-based software enables the creation of simulation environments that are extremely modular, facilitating the extensive reusability of software components. The simulation settings face increased pressures due to the expansive nature of web-based simulations. The incorporation of component technology within the web-based simulation environment enables the treatment of simulation models as individual components, which may be flexibly assembled to construct the model. Furthermore, this web based simulation facilitates the dynamic linkage of simulation inputs and outputs with database systems, so providing a convenient and adaptable means of storing simulation results.

Keywords: *Java, Replanting, Sustainable, Web Based Simulation*

PENDAHULUAN

Pengembangan simulasi penanaman kembali berbasis web akhir-akhir ini telah muncul sebagai jalan yang signifikan untuk memajukan penelitian dan pengembangan di bidang pertanian (Sharma, Tripathi, & Mittal, 2022). Dalam keadaan optimal, simulasi berbasis web harus memungkinkan distribusi dan komposisi model dan hasil simulasi dengan mudah, serupa dengan aksesibilitas dan komposisi publikasi web saat ini. Tidak diragukan lagi, tugas ini memiliki tingkat kompleksitas yang lebih besar dalam praktiknya dibandingkan dengan uraian teoretisnya. Proses *hyperlinking model* simulasi bukanlah tugas yang mudah. Pendekatan alternatif terhadap model hyperlink adalah meningkatkan aksesibilitas dan ketersediaannya di internet. Eksekusi operasi sisi klien difasilitasi oleh materi yang dapat dieksekusi, seperti applet Java (Esquembre, 2004). Simulasi yang mampu memberikan hasil yang detil dan reliabel masih dapat dilakukan pada komputer berperforma tinggi. Salah satu pendekatan untuk mencapai hal ini adalah dengan mengimplementasikan klien tipis yang beroperasi dalam browser web dan menjalin komunikasi dengan *server*. *Server* mengeksekusi model menggunakan teknologi seperti servlet atau aplikasi Java.

Meningkatnya kebutuhan penggunaan simulasi berbasis web mengharuskan penyediaan lingkungan simulasi yang sesuai, yang mampu menghasilkan hasil simulasi yang lebih relevan (Forouzandeh, Tahsildoost, & Zomorodian, 2021). Mengingat jangkauan

global dari model dan kemampuannya untuk menghasilkan hasil simulasi, sangat penting untuk melakukan penelitian di bidang ini. Konektivitas basis data Java memfasilitasi penyimpanan hasil simulasi secara sistematis dalam basis data relasional atau relasional objek. Meskipun JDBC (*Java Database Connectivity*) menangani masalah konektivitas database tingkat rendah, namun kesulitan yang besar tetap ada. Java juga memungkinkan pembuatan antarmuka pengguna grafis (*Graphical User Interface*) yang mudah digunakan untuk menampilkan data yang diambil dari database. Sebagai hasilnya, pengguna tidak perlu memiliki pengetahuan yang luas tentang kriteria pemformatan untuk mengambil hasil simulasi yang disimpan dalam database di internet. Untuk memfasilitasi pengembangan model yang cepat, sangat penting bahwa lingkungan menawarkan alat visual yang memungkinkan desain model simulasi.

Penelitian ini menelusuri banyak aspek yang berkaitan dengan lingkungan simulasi yang digunakan dalam simulasi berbasis web. Hal-hal yang disebutkan di atas ditunjukkan melalui analisis modul fungsional dalam lingkungan simulasi berbasis web (Liu, Jiang, & Jiang, 2020). Iterasi saat ini telah mengalami modifikasi untuk mengintegrasikan metodologi terbaru dalam pengembangan perangkat lunak, yang dikenal sebagai teknologi berbasis komponen. Penelitian ini berpotensi mengarah pada realisasi peningkatan yang diantisipasi dalam produktivitas pengembangan perangkat lunak. Pembangunan sistem perangkat lunak yang rumit akan melibatkan perakitan beberapa komponen, sehingga mengurangi kebutuhan akan kode tingkat rendah yang ekstensif. Sistem yang dibangun menggunakan berbagai komponen menunjukkan peningkatan ekstensibilitas dan memiliki kemampuan untuk beroperasi secara efisien dalam lingkungan yang terdistribusi dan heterogen. Perangkat lunak berbasis komponen umumnya kompatibel dengan lingkungan online, dicontohkan dengan pemanfaatan teknologi seperti Java beans dan Active X. Kerangka kerja yang dikembangkan dengan menggunakan bahasa pemrograman Java dan teknologi Java beans, menawarkan banyak manfaat yang sebanding dengan yang disediakan oleh simulasi berbasis web (Vishwanath, 2023).

Bagian selanjutnya dari bab ini memberikan gambaran umum tentang kegiatan penelitian dan pengembangan yang berkaitan dengan simulasi berbasis web. Penelitian ini juga menyajikan wacana tentang pentingnya teknologi komponen perangkat lunak. Untuk menjelaskan hal ini, sebuah analisis komprehensif dilakukan pada komponen spesifik dari lingkungan simulasi berbasis web. Kerangka kerja simulasi berbasis web ini terdiri dari tiga lapisan hirarki paket Java. Kelompok awal, yang disebut sebagai paket dasar simulasi,

memiliki relevansi minimal dengan simulasi berbasis web; namun demikian, ini tetap merupakan komponen penting untuk sistem simulasi apa pun. Paket-paket penyusunnya dijelaskan secara ringkas untuk memberikan gambaran umum yang komprehensif. Simulasi berbasis web yang dikembangkan termasuk dalam kelompok kedua, memiliki relevansi yang lebih besar karena kemampuannya untuk merepresentasikan model simulasi sebagai Java beans. Hal ini memungkinkan distribusi dan perakitan model-model ini ke dalam federasi model dengan mudah. Paket-paket data lingkungan penanaman kembali, sebagai bagian dari kelompok ketiga, berfungsi untuk memungkinkan pengembangan model simulasi, membuat koneksi dengan database, mengelola eksekusi model atau federasi model secara terkontrol, dan mendukung simulasi berbasis kueri.

METODE PENELITIAN

Penelitian ini menggunakan metode pengembangan Waterfall dalam membuat simulasi replanting berbasis website, pendekatan ini adalah pendekatan terkemuka yang digunakan dalam investigasi simulasi. Teknik Waterfall adalah strategi manajemen proyek yang terdefinisi dengan baik dan sistematis yang biasa digunakan dalam pengembangan perangkat lunak (Farahat, & Defina, 2022). Dalam konteks diskusi ini, teknik ini digunakan untuk pengembangan konten simulasi pertanian demi tercapainya hasil yang sustainable. Fenomena yang sedang dipertimbangkan dibedakan dengan kemajuan linier yang berurutan melalui serangkaian tahapan, di mana setiap tahap dibangun di atas fondasi yang diletakkan oleh pendahulunya.

HASIL DAN PEMBAHASAN

Lapisan pada Proses Simulasi

Pada penelitian ini terdapat tiga paket lapisan yang berbeda, lapisan paling bawah terdiri dari paket antrian, statistik, dan variate. Paket-paket ini memiliki penerapan luas dalam simulasi dan disiplin ilmu terkait lainnya. Paket antrian ditambatkan oleh kelas *Queue*, yang berfungsi sebagai kelas dasar abstrak. Kelas ini berfungsi sebagai dasar untuk kelas turunan: FIFO *Queue*, LIFO *Queue*, *PriorityQueue*, dan *TemporalQueue*. Antrean prioritas dan temporal dapat berfungsi untuk mengatur jalur berdasarkan prioritas, serta memfasilitasi penerapan kejadian masa depan (*Future Event List*) dan algoritme prediksi kejadian di masa yang akan datang.

Paket statistik terdiri dari serangkaian kelas yang dirancang untuk tujuan mengumpulkan dan mengatur data statistik. Kelas Statistik berfungsi sebagai kelas dasar

abstrak dan mencakup banyak metode yang memfasilitasi analisis data statistik dan membantu dalam menghasilkan hasil simulasi. Kelas *SampleStat* dan *TimeStat* berasal dari kelas Statistik. Kelas *SampleStat* digunakan untuk mengumpulkan data statistik dari sampel, yang dicapai melalui metode penghitungannya. Di sisi lain, kelas *TimeStat* digunakan untuk mengumpulkan statistik yang bertahan sepanjang waktu, dicapai melalui metode akumulasinya. Kelas Statistik memiliki kemampuan untuk menghitung beberapa ukuran statistik, termasuk minimum, maksimum, rata-rata, varians, deviasi standar, akar rata-rata kuadrat, dan setengah lebar interval kepercayaan. Paket ini juga menyertakan kelas *BatchStatistic* yang berasal dari kelas *SampleStat*. Kelas ini digunakan untuk tujuan mengumpulkan statistik batch.

Paket variate menawarkan data beragam variasi acak. Kelas Variate berfungsi sebagai kelas dasar yang diperluas oleh semua variate lainnya. Kelas Variate di pemodelan simulasi ini menggunakan generator kongruensial liniernya sendiri, yang dikenal sebagai *LCGRandom*. Namun, dimungkinkan untuk beralih menggunakan kelas Random Java dengan membuat modifikasi pada kode kelas Variate. Selain itu, pemasangan generator nomor acak tambahan adalah tugas yang mudah. Simulasi berbasis web ini mencakup implementasi 14 generator variate acak kontinu dan delapan generator variate acak diskrit. Paket variate di JSIM menawarkan pilihan variabel acak diskrit, termasuk Bernoulli, Binomial, DiscreteProb, Geometric, Hypergeometric, Negative Binomial, Poisson, dan Randi. Variasi acak kontinu yang dapat diakses adalah Beta, Cauchy, ChiSquare, Erlang, Exponential, F Distribution, Gamma, HyperExponential, LogNormal, Normal, dan Weibull.

Tabel 1. Lapisan Paket

Lapisan lingkungan			
paket *model*	paket *query*	paket *runAgency*	paket *qdsAgency*
Lapisan pemrosesan			
paket *event*		paket *proses*	
Lapisan dasar			
paket antrian	paket statistik	paket variate	

Ada empat paket lingkungan berbeda yang tersedia. Lapisan paling atas dari sistem ini menawarkan kerangka kerja yang dapat ditempa untuk melakukan simulasi berbasis web. *Modelpackage* menawarkan antarmuka grafis untuk merancang proses dalam paket proses. Saat simulasi dijalankan, rendering model yang dibuat oleh desainer berpotensi untuk dianimasikan. Paket *Query* terdiri dari kode serbaguna yang memfasilitasi interaksi

antara model dan database. Kerangka kerja ini terdiri dari barisan data yang memfasilitasi interaksi dengan database melalui penggunaan website ini. Paket *runAgency* bertanggung jawab untuk mengelola eksekusi satu atau beberapa model. Dimasukkannya data dalam paket ini memberi pengguna pendekatan yang disederhanakan untuk mengeksekusi model atau federasi model, sehingga mengurangi kebutuhan akan detail yang ekstensif. Paket *qdsAgency* menawarkan metode yang mudah untuk memperoleh dan membuat data simulasi, khususnya untuk simulasi berbasis kueri (Smith, 2021). Paket perangkat lunak yang disebutkan di atas memungkinkan penyimpanan input dan output simulasi dalam database, serta eksekusi model simulasi selama pemrosesan kueri. Fungsionalitas ini memungkinkan analisis hasil simulasi yang kohesif dan lugas, terlepas dari keberadaannya. Dua paket terakhir umumnya dikenal sebagai agen karena dimasukkannya Java beans yang memiliki kemampuan untuk berkolaborasi dan mengambil keputusan, sehingga memenuhi syarat sebagai agen.

Proses Simulasi

Seperti yang telah disebutkan sebelumnya, penyediaan lingkup yang memfasilitasi simulasi berbasis web memerlukan peningkatan persyaratan karena kemungkinan skalanya yang luas. Selain itu, pengguna internet menunjukkan kurangnya kecenderungan untuk terlibat dengan teknologi yang kurang ramah pengguna. Lingkungan simulasi berbasis web yang dikembangkan ini terdiri dari empat paket, yaitu paket *jmodel, jquery, runAgency,* dan *qdsAgency*, yang digunakan untuk mengatasi masalah yang ada.

Paket *jmodel* adalah model visual yang diimplementasikan dan dirancang menggunakan perpustakaan Java Swing. Pembuat model adalah antarmuka pengguna grafis (GUI) yang memberikan pendekatan yang lebih intuitif dan langsung kepada para simulasi untuk membangun model. Pengguna dapat meletakkan item simulasi pada kanvas pembuat model dengan memilih tombol dari toolbar dan kemudian mengklik tempat yang diinginkan pada kanvas untuk menempatkan objek, seperti node server. Paket ini dirancang khusus untuk memfasilitasi pengkodean model secara manual secara cepat dengan memanfaatkan perpustakaan kelasnya yang komprehensif. Namun, pendekatan paling mudah dalam pembuatan model adalah dengan memanfaatkan desainer visual *Wmodel*. Perangkat lunak *Wmodel* menawarkan serangkaian tombol yang memungkinkan pengguna memanipulasi pembuatan model di atas kanvas. Tombol kontrol yang tersedia saat ini ditampilkan pada Tabel 2. Konstruksi model dilakukan menggunakan antarmuka visual dengan mengaktifkan mode desainer melalui pemilihan

tombol. Selanjutnya, setelah aktivasi peristiwa klik mouse, tindakan terkait akan dijalankan pada koordinat yang tepat di dalam kanvas gambar. Jika sistem berada dalam mode "Cuaca", kondisi cuaca baru akan digambarkan di tempat yang ditentukan. Untuk membuat koneksi antara dua node menggunakan mekanisme transport, Anda perlu mengaktifkan mode "Penanaman" dan kemudian memilih dua node yang diinginkan dengan mengkliknya. Tindakan ini akan menghasilkan terciptanya lintasan linier. Untuk menghasilkan kurva, pengguna diharuskan memilih titik yang terletak di luar node, yang kemudian akan berfungsi sebagai titik kontrol.

Tabel 2. Fitur pada *Wmodel*

Tombol	Deskripsi
Cuaca	Memberikan simulasi perkiraan cuaca
Penanaman	Memberikan hasil simulasi penanaman tiap tahun
Keadaan tanah	Memberikan hasil simulasi perubahaan keadaan tanah
Server	Menyediakan layanan untuk entitas yang tiba di node
Fasilitas	Memanggil data dari server dan menambahkan antrian untuk menampung entitas yang baru
Sinyal	Mengubah jumlah unit layanan dalam server
Sumber	Menghasilkan entitas dengan waktu antar-kedatangan yang acak
Akhir	Menghentikan entitas dan mencatat statistiknya
Hubung	Menghubungkan dua node dari 1 ke yang lainnya
Pindah	Memindahkan node ke posisi baru di kanvas
Hapus	Menghapus node atau tepi
Perbaharui	Melihat/mengubah properti node yang dipilih
Hasil	Menjalankan kode Java yang mengimplementasikan model yang dirancang

Kode dalam jmodelpackage dirancang dengan tujuan memfasilitasi ekstensibilitas. Setiap simpul dalam grafik berhubungan dengan poligon, sehingga memudahkan penambahan bentuk baru secara langsung untuk merepresentasikan berbagai jenis simpul. Dengan cara yang sama, dapat diamati bahwa setiap sisi yang ada dalam grafik dapat direpresentasikan sebagai kurva segi empat, sehingga memungkinkan terbentuknya jalur serbaguna yang menghubungkan node yang berbeda.

Paket *jquery* mencakup kumpulan kelas dan *bean* yang memfasilitasi interaksi tanpa batas antara berbagai komponen dan basis data relasional, seperti Mini SQL, serta basis data relasional objek seperti Oracle 8. Simulasi yang dikembangkan memiliki kemampuan

untuk membuat koneksi dengan beberapa sistem manajemen basis data karena ketergantungannya pada konektivitas basis data Java (*Java database connectivity*/JDBC). Secara umum, proses ini dapat disederhanakan dengan memodifikasi spesifikasi koneksi, seperti transisi dari Mini SQL ke Oracle. Jika terjadi skenario yang tidak menguntungkan, mungkin perlu memodifikasi driver JDBC. Namun, perlu dicatat bahwa penyesuaian ini hanya akan berdampak pada sejumlah baris kode yang terbatas, karena API JDBC tetap konsisten apa pun situasinya. Paket ini berisi beberapa hal penting, yang meliputi hal-hal berikut ini.

- *DbQuery bean* adalah komponen yang digunakan dalam konteks kueri basis data. *DbQuery bean* dirancang untuk menangani peristiwa yang berisi kueri SQL. Dengan memanfaatkan kerangka kerja Java Database Connectivity (JDBC), kueri dikirimkan ke database yang ditentukan, dan hasilnya disimpan dalam *AbstractTableModel*. Hal ini memfasilitasi transmisi hasil yang mulus kembali ke pemohon atau tampilan mereka.
- *DbUpdate bean* adalah komponen yang digunakan untuk memperbarui database. *DbUpdate bean* menunjukkan perilaku yang sama dalam merespon kejadian yang melibatkan perubahan SQL. Model *bean* dapat memanfaatkannya sebagai sarana untuk menyimpan hasil dalam *database*. Fitur ini memfasilitasi eksekusi model dengan atau tanpa database, serta kemampuan untuk berpindah antar database dengan mulus karena pembentukan asosiasi yang dinamis antara *bean*.

Dalam kerangka simulasi berbasis website ini, sangat memungkinkan untuk membuat model sebagai Java *bean*, memungkinkan pembentukan federasi model dengan menggabungkan model yang lebih sederhana. Federasi-federasi ini kemudian dapat dilaksanakan secara *seamless*. Selain itu, model dapat dihubungkan secara dinamis dengan *bean* yang berasal dari lingkungan sekitar. Model *bean* dapat diklasifikasikan sebagai komposit, dan memiliki kemampuan untuk digabungkan menjadi federasi model. Bentuk paling dasar dari suatu model disebut sebagai model atom. Konstruksi model atom melibatkan pembuatan digraf terhubung yang terdiri dari sumber soliter dan sink soliter. Sumber mengacu pada entitas produsen, seperti pelanggan, yang melintasi grafik dan pada akhirnya dihapus menggunakan tombol hapus. Sangat penting bahwa semua node yang tersisa memiliki tepi masuk dan keluar.

Model komposit terdiri dari banyak model yang berbagi lingkungan dan bingkai tampilan yang sama. Mengingat bahwa model umum berpotensi diambil dari beberapa sumber, maka dimungkinkan untuk menghasilkan entitas berbeda yang mewakili kategori berbeda (misalnya, simulasi penanaman Kembali pohon karet dan pohon kelapa sawit).

Aliran-aliran yang disebutkan di atas dapat dikategorikan sebagai aliran otonom, persaingan, atau interaktif, bergantung pada ada atau tidaknya node bersama dan peran yang dimainkan oleh masing-masing aliran. Dimasukkannya digraf dengan berbagai sumber dan sumber menimbulkan peringatan yang rumit mengenai kondisi digraf yang terbentuk dengan baik.

Model komposit dapat dibuat melalui integrasi beberapa model sederhana yang sudah ada sebelumnya. Proses ini melibatkan integrasi dua model ke dalam platform *jmodeldesigner*, memungkinkan pengguna untuk membuat koneksi yang diperlukan di antara keduanya. Selanjutnya, kode baru dihasilkan untuk model komposit. Perlu dicatat bahwa proses sintesis model komposit baru menunjukkan tingkat dinamisme yang lebih rendah dibandingkan dengan perakitan federasi model baru. Alasannya adalah karena elemen-elemen menunjukkan tingkat saling ketergantungan yang lebih tinggi dalam model komposit, karena elemen-elemen tersebut memiliki elemen fungsional bersama.

Merakit Model Menjadi Federasi Model

Federasi model dapat dibangun dengan merakit model, tanpa memerlukan pemrograman konvensional. Setiap model dapat disimbolkan dengan ikon dan dieksekusi sebagai Java *bean*. Desainer mempunyai kemampuan untuk memilih dari serangkaian model yang sudah ada sebelumnya untuk membangun federasi model secara dinamis. Keterkaitan antara model individu dalam federasi dibangun melalui pemanfaatan *event* Java *bean*. Ketika suatu entitas dalam model tertentu mendekati *sink*, entitas tersebut dapat memulai peristiwa eksternal yang selanjutnya akan diproses dalam model terpisah. Penciptaan suatu entitas dalam suatu model merupakan konsekuensi langsung dari pengelolaan peristiwa eksternal oleh suatu sumber. Satu model memiliki kemampuan untuk berinteraksi dengan model lain melalui proses memasukkan entitas. Saat ini, diasumsikan bahwa injeksi entitas terjadi pada model target saat ini, dengan penerapan sinkronisasi metode yang sesuai untuk mengurangi terjadinya masalah balapan. Jika waktu tertentu, dilambangkan dengan t, disediakan untuk injeksi entitas, terdapat kemungkinan bahwa t merujuk pada periode di masa lalu yang relatif terhadap model target. Oleh karena itu, penggunaan teknik simulasi paralel diperlukan untuk mengatasi skenario ini (D'Angelo, & Marzolla, 2014).

Interaksi ini memerlukan pembentukan hubungan antara sink dalam satu model dan sumber dalam model lainnya. Pembentukan hubungan ini tidak dicapai melalui cara yang telah dirancang atau dikodekan sebelumnya, melainkan melalui pemanfaatan dinamis alat

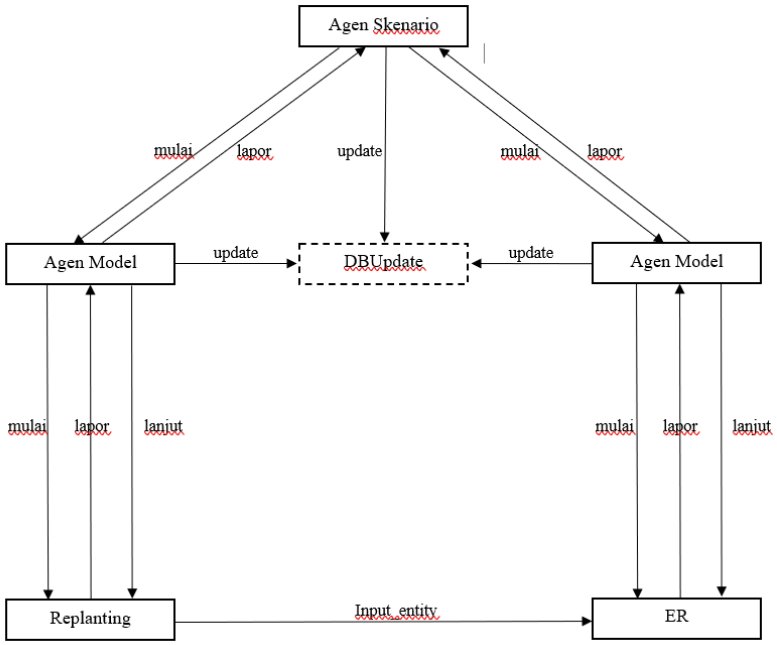
pengembangan visual seperti BeanBox di Beans Development Kit, Java Studio, atau Visual Cafe, dan lain-lain.

Federasi model Penanaman Kembali yang berkelanjutan dapat dianggap sebagai skenario yang cocok untuk membangun hubungan antara kedua model tersebut. Beberapa *entity* dan variabel yang berangkat dari *ReplantPlant* menginput data cuaca, tingkat kesuburan tanah, kelembapan, siklus panen, dll, selanjutnya *user* memilih untuk melakukan simulasi setelah mengisi bobot tiap variable yang ditandai sebagai *triggerProb*. Sesampainya di tahun yang diinginkan, pengguna melakukan asesmen awal oleh *entityTriage* yang bertugas menentukan tindakan yang tepat yaitu apakah tanaman harus di tanam kembali apa masih bisa dilanjutkan dan penilaian ini di simpan oleh database untuk pensimulasian di masa depan.

Prosedur yang ditentukan akan diterapkan pada saat terjadinya suatu peristiwa *random*. Terjadinya peristiwa akan dimulai dan ditransmisikan setelah penangkapan suatu entitas oleh Sink dalam *bean Replant*. Tindakan ini akan mengakibatkan Sumber dalam *Entity Recognition* (ER) *bean* memulai pembuatan dan aktivasi (yaitu, injeksi) entitas baru. Intinya, fungsi *Sink.capture* memanggil metode *Model.triggerEntityEvent*, yang bertanggung jawab untuk membuat dan menyebarkan *EntityEvent* ke semua penerima yang ditunjuk. Penerima ini dapat ditentukan atau dihubungkan secara dinamis dan visual menggunakan kerangka BeanBox. Kelas adaptor, yang dibuat secara otomatis selama penautan peristiwa, berfungsi sebagai *recorder* untuk *EntityEvents*. Setelah menerima sinyal, sistem akan memanggil fungsi *ER.injectEntity*, yang telah ditetapkan sebelumnya selama fase *linkage*. Selanjutnya, prosedur yang disebutkan di atas akan memanggil *Source.startEntity*, sehingga memperkenalkan entitas baru ke dalam sistem (ER) sebagai reaksi terhadap rangsangan eksternal. Entitas yang disuntikkan ini merupakan pelengkap dari entitas apa pun yang biasanya dihasilkan oleh sumber data secara internal. Setelah eksekusi federasi model, diamati bahwa setiap model mengalami animasi dalam bingkainya yang berbeda. Model-model tersebut memiliki kemampuan untuk beroperasi secara mandiri; meskipun demikian, mereka umumnya terlibat dalam interaksi yang difasilitasi oleh peristiwa eksternal yang dihasilkan oleh satu model dan dikelola oleh model lainnya. Pengamatan komprehensif terhadap animasi model yang rumit terbukti menantang, karena sulit untuk sepenuhnya memahami dan memperoleh makna darinya. Dengan memanfaatkan animasi multi-bingkai, akan lebih mudah untuk mengarahkan perhatian ke aspek-aspek tertentu dari simulasi dengan menempatkan bingkai jendela yang relevan di latar depan pada layar.

Subyek diskusi berkaitan dengan *bean ModelAgent*. Peran *ModelAgent* melibatkan eksekusi model pada simulasi berbasis web ini. Model akan dimulai untuk durasi atau beban kerja tertentu. Model akan memberikan laporan termasuk hasil dari proses atau *batch* ini. Agen model selanjutnya akan menentukan apakah akan menyimpulkan atau memperpanjang eksekusi model, berdasarkan metodologi keluaran yang dipilih (Cook, & Wolf, 1998). Keputusan ini mungkin termasuk melakukan proses atau batch lebih lanjut. Teknik ini menunjukkan tingkat modularitas dan fleksibilitas yang tinggi, memungkinkan integrasi agen model baru ke dalam sistem, asalkan mereka mematuhi aturan permainan yang telah ditetapkan. Saat ini, terdapat dua kategori agen model berbeda yang telah berhasil diterapkan. Kategori pertama menggunakan pendekatan replikasi independen, sedangkan kategori kedua menggunakan metode batch mean.

Dalam skenario yang diberikan, subjek diskusinya adalah *Agentbeans*. Tugas utama untuk mengawasi penerapan federasi model terletak pada Agen Skenario. Setelah upaya kolaboratif antara berbagai agen untuk menentukan pendekatan penerapan federasi model, agen skenario akan mengkomunikasikan instruksi kepada agen model mengenai penggunaan metodologi analisis keluaran yang sesuai, seperti metode replikasi independen atau metode *batch*. Selain itu, agen skenario akan menentukan tujuan analisis yang sesuai, termasuk tingkat kepercayaan yang diinginkan dan presisi relatif. Gambar 1 mengilustrasikan contoh struktur eksekusi yang digunakan untuk federasi model *SustainableReplant*. Representasi visual tersebut menggambarkan organisasi spasial dari *Replanting* dan ER *beans* di dalam *bean box*, serta keterhubungan di antara keduanya.



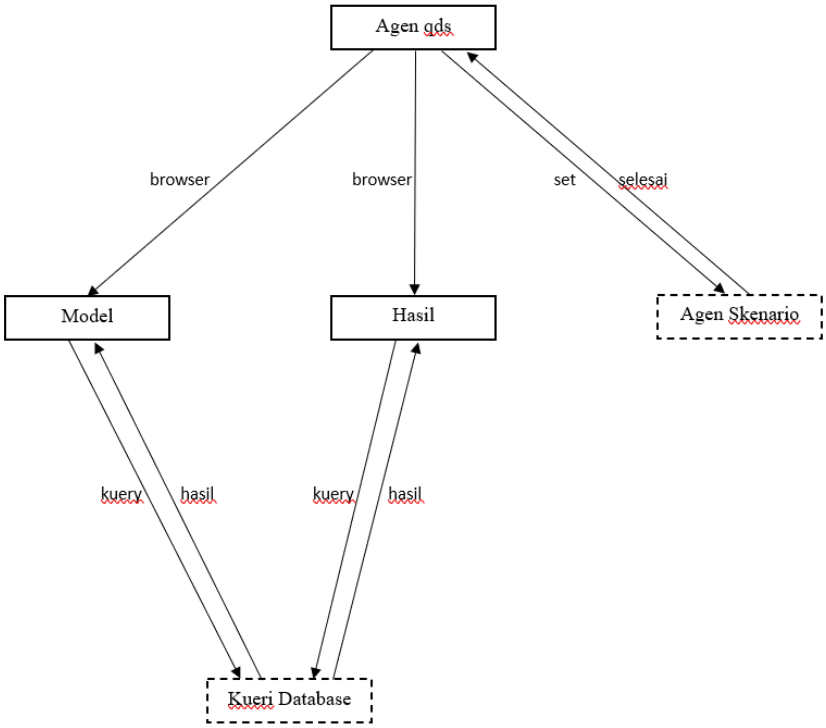
Gambar 1. Struktur Eksekusi *Replanting*

Paket *qdsAgency*

Simulasi dapat diintegrasikan ke dalam sistem lain melalui berbagai cara. Salah satu metodologi tertentu, yang dikenal sebagai simulasi berbasis kueri (*Query Driven Simulation*/QDS), beroperasi berdasarkan prinsip bahwa analis simulasi dan pengguna yang tidak berpengalaman harus menganggap sistem berbasis QDS sebagai sistem informasi yang canggih. Sistem ini memberikan akses yang konsisten terhadap informasi mengenai perilaku sistem yang dipelajari (Weidlich, Mendling, & Weske, 2010). Hal ini menyiratkan bahwa pengguna harus disajikan dengan antarmuka ramah pengguna yang memungkinkan mereka memulai aktivitas rumit dengan memasukkan pertanyaan langsung, seperti melalui formulir.

Proses simulasi sering kali ditandai dengan tuntutan komputasi yang tinggi dan biaya yang signifikan. Oleh karena itu, logis untuk menyimpan hasil simulasi untuk pemanfaatan di masa depan. Sistem manajemen basis data (*Database Management System*/DBMS) menawarkan metodologi yang efektif untuk penyimpanan, modifikasi, dan pengambilan data dalam jumlah besar (Rawat, & Purnama, 2021). Oleh karena itu, disarankan agar sistem QDS menyertakan database untuk tujuan menyimpan hasil simulasi dan model simulasi. Selanjutnya, data tambahan yang berkaitan dengan proses pengambilan keputusan akan dicatat dan dihubungkan dengan hasil simulasi.

Ketika pengguna mengirimkan kueri ke sistem simulasi yang dibangun pada kerangka QDS, sistem awalnya mencoba untuk mengidentifikasi informasi yang diperlukan dalam database, karena mungkin saja informasi ini telah direkam sebagai hasil dari eksekusi sebelumnya. model. Jika data yang diperlukan tersedia, data tersebut diambil dan kemudian diberikan kepada pengguna. Dengan tidak adanya sistem QDS, model yang diperlukan dibuat instance-nya, dieksekusi, dan hasil yang dihasilkan ditampilkan kepada pengguna. Entitas utama yang bertanggung jawab untuk mengaktifkan keterlibatan pengguna dengan sistem QDS adalah kacang QdsAgent. Kacang ModelBrowser dan ResultBrowser memberikan bantuan dalam hal ini. Sesi prototipikal yang melibatkan pengguna dan sistem QDS dapat dijelaskan sebagai berikut.



Gambar 2. *Bean* pada agen qds

Setelah memilih model atau federasi model yang dapat diakses, proses pada web akan dijalankan oleh *ModelBrowser* sebagai respons terhadap permintaan *qdsAgent*. Setelah pengguna memilih model atau federasi model tertentu, *qdsAgent* memungkinkan pengguna menavigasi hasil simulasi terkait dengan memanfaatkan *ResultBrowser*. Dalam pendekatan alternatif, pengguna mempunyai opsi untuk menetapkan skenario spesifik yang diinginkan untuk model atau federasi model. Proses ini dibantu oleh *qdsAgent* yang memperoleh daftar parameter model dari database, yang mencakup nilai default. Setelah penyediaan parameter, *qdsAgent* melakukan kueri database untuk menentukan keberadaan informasi yang berkaitan dengan eksekusi sebelumnya dari model yang ditunjuk atau federasi model dengan nilai parameter yang disediakan. Setelah hasil simulasi tersedia, *qdsAgent* mengambilnya dari database dengan memanfaatkan *ResultBrowser*. Jika kondisi tidak terpenuhi, *qdsAgent* dan Agen Skenario bekerja sama untuk menjalankan model atau federasi model menggunakan parameter yang disediakan. Tindakan melaksanakan tugas akan mengarah pada penyimpanan hasilnya dalam database. Presentasi hasil kini dapat difasilitasi dengan pemanfaatan *ResultBrowser*.

SIMPULAN

Platform simulasi berbasis web menawarkan beberapa keunggulan dibandingkan dengan lingkungan simulasi tradisional. Simulasi ini memfasilitasi penyebaran hasil

simulasi dan model simulasi berdasarkan aksesibilitasnya melalui platform berbasis web, hal ini memberikan peluang untuk meningkatkan aksesibilitas analisis simulasi dalam skala yang lebih luas. Simulasi berbasis web ini memfasilitasi portabilitas global dengan memungkinkan eksekusi model di beragam lingkungan komputasi, pemanfaatan bahasa pemrograman Java bersama dengan koleksi perpustakaan kelasnya yang luas memfasilitasi pencapaian tujuan ini. Pemanfaatan JDBC memfasilitasi transisi yang mulus antara beberapa sistem manajemen basis data. Kerangka kerja simulasi berbasis web ini memfasilitasi pembangunan federasi model yang dinamis, lingkungan dibangun dengan menggunakan teknologi berbasis komponen, khususnya Java *beans*, yang memungkinkan pemanfaatan komponen perangkat lunak yang dapat digunakan kembali. Komponen-komponen ini dapat digabungkan secara dinamis melalui penggunaan alat pengembangan visual. *ModelBeans* memiliki kemampuan untuk membangun koneksi dengan *bean* model lain untuk membangun federasi model (Rathee, & Chhabra, 2020). Selain itu, mereka dapat digabungkan dengan environment beans untuk mengatur eksekusinya dan menyimpan hasilnya dalam database. Simulasi berbasis web ini menawarkan metode yang mudah dan konsisten untuk mengakses hasil simulasi melalui konsep simulasi berbasis kueri. Pengguna menanyakan informasi, kemudian sistem akan menentukan cara yang paling tepat untuk memperoleh data yang diminta, seperti mengakses database atau menjalankan model simulasi. Untuk penelitian yang akan datang dapat memanfaatkan fungsionalitas yang ditawarkan oleh Java API baru untuk meningkatkan estetika visual, fungsionalitas terdistribusi, dan interoperabilitas sistem.

DAFTAR PUSTAKA

Bertacchini, F., Bilotta, E., Demarco, F., Pantano, P., & Scuro, C. (2021). Multi-objective optimization and rapid prototyping for jewelry industry: methodologies and case studies. The International Journal of Advanced Manufacturing Technology, 112, 2943-2959.

Brito, M., & Gonçalves, C. (2019, June). Codeflex: A web-based platform for competitive programming. In 2019 14th Iberian Conference on Information Systems and Technologies (CISTI) (pp. 1-6). IEEE.

Cheliotis, K. (2021). ABMU: an agent-based modelling framework for Unity3D. SoftwareX, 15, 100771.

Cook, J. E., & Wolf, A. L. (1998). Discovering models of software processes from event-

based data. ACM Transactions on Software Engineering and Methodology (TOSEM), 7(3), 215-249.

D'Angelo, G., & Marzolla, M. (2014). New trends in parallel and distributed simulation: From many-cores to cloud computing. Simulation Modelling Practice and Theory, 49, 320-335.

Esquembre, Francisco. "Easy Java Simulations: A software tool to create scientific simulations in Java." Computer physics communications 156.2 (2004): 199-204.

Esquembre, Francisco. "Easy Java Simulations: A software tool to create scientific simulations in Java." Computer physics communications 156.2 (2004): 199-204.

Farahat, A. M., & Defina, D. (2022, October). Novel Adaptive Approach for Applying and Combining Traditional Waterfall and Agile Project Management Methodologies. In Abu Dhabi International Petroleum Exhibition and Conference (p. D041S125R003). SPE.

Forouzandeh, N., Tahsildoost, M., & Zomorodian, Z. S. (2021). A review of web-based building energy analysis applications. Journal of Cleaner Production, 306, 127251.

Goetschalckx, L., Andonian, A., Oliva, A., & Isola, P. (2019). Ganalyze: Toward visual definitions of cognitive image properties. In Proceedings of the ieee/cvf international conference on computer vision (pp. 5744-5753).

Halim, D. K., Ming, T. C., Song, N. M., & Hartono, D. (2019, October). Arduino-based IDE for embedded multi-processor system-on-chip. In 2019 5th International Conference on New Media Studies (CONMEDIA) (pp. 135-138). IEEE.

Kaniayeu, Y., & Rusetski, I. (2020). Swift Object-Oriented Programming Language: Features and Advantages.

Liang, X., Luo, L., Hu, S., & Li, Y. (2022). Mapping the knowledge frontiers and evolution of decision making based on agent-based modeling. Knowledge-Based Systems, 250, 108982.

Liu, C., Jiang, P., & Jiang, W. (2020). Web-based digital twin modeling and remote control of cyber-physical production systems. Robotics and computer-integrated manufacturing, 64, 101956.

Martinez, I., Hafid, A. S., & Jarray, A. (2020). Design, resource management, and evaluation of fog computing systems: a survey. IEEE Internet of Things Journal, 8(4), 2494-2516.

Mendoza, M. D., Hutajulu, O. Y., & Salman, R. (2022, February). Community constraints analysis in the use of solar power plants in Indonesia. In Journal of Physics:

Conference Series (Vol. 2193, No. 1, p. 012079). IOP Publishing.

Mendoza, M. D., Hutajulu, O. Y., Lubis, A. R., Rahmadani, R., & Putri, T. T. A. (2022). PENGARUH PENGGUNAAN MEDIA SOSIAL DALAM PENDIDIKAN TERHADAP PRESTASI AKADEMIK MAHASISWA. Jurnal Teknologi Pendidikan (JTP), 15(2), 68-80.

Rathee, A., & Chhabra, J. K. (2020). Mining reusable software components from object-oriented source code using discrete pso and modeling them as java beans. Information Systems Frontiers, 22(6), 1519-1537.

Rawat, B., & Purnama, S. (2021). MySQL Database Management System (DBMS) On FTP Site LAPAN Bandung. International Journal of Cyber and IT Service Management, 1(2), 173-179.

Roman, R., Zhou, J., & Lopez, J. (2013). On the features and challenges of security and privacy in distributed internet of things. Computer networks, 57(10), 2266-2279.

Sehgal, N. K., Bhatt, P. C. P., & Acken, J. M. (2020). Cloud computing with security. Concepts and practices. Second edition. Switzerland: Springer.

Sharma, V., Tripathi, A. K., & Mittal, H. (2022). Technological revolutions in smart farming: Current trends, challenges & future directions. Computers and Electronics in Agriculture, 107217.

Smith, A. D. (2021). Event detection in educational records: an application of big data approaches. International Journal of Business and Systems Research, 15(3), 271-291.

Srirama, S. N., Dick, F. M. S., & Adhikari, M. (2021). Akka framework based on the Actor model for executing distributed Fog Computing applications. Future Generation Computer Systems, 117, 439-452.

Vishwanath, M. (2023). Adaptive Educational Hypermedia System Architecture Trends: The MATHEMA Case.

Weidlich, M., Mendling, J., & Weske, M. (2010). Efficient consistency measurement based on behavioral profiles of process models. IEEE Transactions on Software Engineering, 37(3), 410-429.

Yan, X., Hu, C., & Sheng, V. S. (2020). Data-driven pollution source location algorithm in water quality monitoring sensor networks. International Journal of Bio-Inspired Computation, 15(3), 171-180.