

ANALISIS DAN DESAIN SISTEM MULTIAGENT PADA SENSOR WIRELESS

Richki Hardi

Program Studi S1 Teknik Informatika
Sekolah Tinggi Teknologi (STITEK) Bontang-Kalimantan Timur
richkihardi@gmail.com

ABSTRAK

Wireless Sensor atau Wireless Sensor Network (WSN) merupakan perangkat embedded kecil yang dipasang di jaringan skala besar yang memiliki kapabilitas penginderaan, komputasi, dan komunikasi. WSN mengkombinasikan teknologi sensor modern, teknologi micro electronic, komputasi, teknologi komunikasi, dan pemrosesan terdistribusi. Implementasi sistem multiagent pada WSN cukup menjanjikan untuk meningkatkan efektifitas dan efisiensi kerja WSN. Namun, penelitian yang dilakukan terkait sistem multiagent di WSN masih parsial dengan kata lain terlalu fokus pada isu-isu tertentu. Paper ini mendeskripsikan penelitian terkait dengan penerapan sistem multiagent di WSN yang memperhatikan berbagai aspek pendukung untuk efektifitas dan efisiensi agent seperti arsitektur organisasi multiagent, itinerary planning, kapabilitas agent, middleware dan platform hardware yang digunakan. Metodologi yang digunakan adalah Ingenias yang berbasis pada agent oriented software engineering.

Kata Kunci: *Agent, Ingenias, Itinerary, Multiagent, Wireless Sensor Network.*

PENDAHULUAN

Perkembangan teknologi sensor semakin pesat dengan kapabilitas tidak hanya pada aspek penginderaan dan signal acquisition, namun memiliki kapabilitas dalam melakukan komputasi dan komunikasi dengan perangkat lainnya. Sensor ini dinamakan sebagai Wireless Sensor Network (WSN) yang juga memanfaatkan teknologi internet sebagai media komunikasinya. WSN memiliki beberapa karakteristik seperti alokasi energy dan bandwidth terbatas, unattended ad hoc deployment, cakupan skala luas, high noise dan fault rate, lingkungan yang dinamis dan tak menentu, serta memberikan dampak pada pengembangan aplikasi yang variatif seperti structural monitoring, bio-habitat monitoring, industrial monitoring, disaster management, military surveillance, dan building security. Karakteristik ini memberikan tantangan bagi pengembangan WSN. Saat ini WSN di-deploy dengan pendekatan client-server.

Menurut Min Chen, et al pendekatan ini merupakan metode tradisional penyebaran data di WSN. Kemunculan event me-trigger node-node sumber di sekitarnya untuk

mengumpulkan dan mengirim data ke sink sendiri. Jumlah aliran data umumnya sama dengan jumlah node-node sumber sehingga menyebabkan konsumsi bandwidth dan energi yang cukup tinggi. Pendekatan ini menyebabkan ketidakseimbangan konsumsi energi di jaringan karena node-node yang lebih dekat dengan sink akan mengirim lebih banyak data, yaitu data miliknya maupun data yang dititipkan dari node lain. Untuk itu, diperlukan pendekatan lain yang dapat menyelesaikan permasalahan di atas, yaitu salah satunya dengan pendekatan sistem mobile agent.

Menurut Stefano Get al pendekatan berbasis mobile agent merupakan alternatif pendekatan arsitektur di WSN. Sink node mengirimkan mobile agent ke area target untuk mengunjungi node satu per satu, lalu data-data sensor dikurangi dan dikumpulkan oleh agent, dan selanjutnya sesuai dengan instruksi mobile agent, data-data tersebut dikirim kembali ke sink. Pendekatan ini menghasilkan satu aliran lalu lintas data daripada banyak aliran lalu lintas data. Pendekatan mobile agent memerlukan perencanaan dan perancangan arsitektur yang baik untuk menjamin efektifitas dan

efisiensi kinerja dari agent-agent yang terlibat di dalamnya. Untuk itu, diperlukan berbagai macam pertimbangan dalam melakukan perencanaan dan perancangan mobile agent tersebut terutama pertimbangan terkait dengan konsep sistem multiagent.

Menurut X. Wang, et al ada beberapa aspek yang perlu dipertimbangkan dalam merancang WSN dengan pendekatan mobile agent, yaitu aspek arsitektur, itinerary planning, perancangan middleware, dan aspek hardware.

PEMBAHASAN

Konsep Dasar Sistem Multiagent

Menurut Michael Wooldridge, agent adalah sistem komputer yang memiliki kapabilitas melakukan aksi yang otonom di lingkungan tertentu untuk mencapai tujuan yang didelegasikan kepadanya. Agent yang intelligent memiliki perilaku reaktif, proaktif, dan sosial. Sistem reaktif adalah sistem yang memelihara interaksi yang sedang berjalan dengan lingkungannya dan memberikan respon terhadap perubahan yang terjadi. Proaktif (goal directed behaviour) berarti agent menghasilkan dan berusaha untuk mencapai goal, tidak hanya didorong oleh event, namun punya inisiatif dan mengenali opportunity. Kemampuan sosial adalah kemampuan untuk berinteraksi dengan agent lain (dapat juga manusia) melalui kerja sama, koordinasi, dan negosiasi atau istilah umumnya adalah kemampuan untuk berkomunikasi. Properti-properti agent lainnya adalah mobility, veracity (ketelitian), benevolence (dapat bekerja sama), rationality, dan learning/Adaption.

Sejalan dengan sistem komputer yang semakin kompleks, diperlukan abstraksi dan metafora yang lebih powerful untuk menjelaskan operasi sistem komputer. Penjelasan low level menjadi tidak praktis sehingga dihasilkan istilah baru bernama Intentionalstance yang merupakan salah satu bentuk abstraksi itu. Intentional notion adalah bentuk tool abstraksi yang memberikan kenyamanan dan cara yang familiar untuk mendeskripsikan, menjelaskan, dan memprediksi perilaku

sistem kompleks. Daniel Dennett mendefinisikan istilah intentional system sebagai entitas-entitas yang perilakunya dapat diidentifikasi oleh metode pengatributan belief, desire, dan rational acumen. Menurutnya agent, sebagai intentional system, merepresentasikan abstraksi powerful lebih lanjut.

Pada dideskripsikan arsitektur Belief, Desire, Intention (BDI) sebagai arsitektur agent yang populer. Salah satu arsitektur BDI yang terkenal adalah Procedural Reasoning System (PRS). Pada sistem PRS, belief merepresentasikan informasi yang dimiliki agent terkait dengan lingkungannya, desire merepresentasikan tugas-tugas yang diberikan kepada agent sesuai dengan obyektif atau tujuan yang harus dicapainya, intention merepresentasikan desire yang agent komitmen lakukan, dan plan menentukan beberapa daftar aksi-aksi yang dapat dilakukan agent untuk mendukung intention-nya. Empat struktur data ini dikelola oleh intrepeter agent yang bertanggung jawab dalam memperbaharui belief dari hasil observasi lingkungan, menghasilkan desire baru sebagai dasar belief baru, dan memilih beberapa set pekerjaan yang harus dilakukan sebagai intention-nya. Selanjutnya intrepeter memilih aksi yang harus dilakukan. Agent yang memiliki karakteristik ini selanjutnya berinteraksi dengan agent-agent lain membentuk sistem multiagent.

Sistem Multiagent, adalah sistem yang terdiri dari agent-agent yang berinteraksi satu sama lain untuk melakukan kerja sama, koordinasi, dan negosiasi. Kerja sama adalah bekerja secara bersama-sama untuk meraih goal bersama, biasanya didorong oleh pernyataan bahwa "Tidak ada satu agent yang bisa mencapai goalnya sendirian" atau "Kerja sama akan menghasilkan hasil yang lebih baik (misalnya, proses berjalan cepat)". Koordinasi adalah mengelola ketergantungan antar aktifitas, sebagai contoh, jika agent ingin menggunakan resource yang tidak di-share, maka perlu dilakukan koordinasi. Beberapa alasan agent perlu koordinasi adalah: (1) goal dari agent-agent dapat menyebabkan konflik di

antara aksi-aksinya, (2) dalam mencapai tujuan, agent dapat saling ketergantungan antar satu agent dengan agent lainnya, (3) agent memiliki kapabilitas dan pengetahuan berbeda, (4) tujuan agent dapat tercapai dengan cepat jika agent berbeda pekerjaan dan kapabilitas bekerja sama secara terkoordinir.

Negosiasi adalah kemampuan untuk mencapai persetujuan terkait dengan kepentingan umum, biasanya melibatkan offer dan counteroffer, dengan kompromi yang dibuat oleh partisipan. Teknik koordinasi untuk alokasi pekerjaan dan sumber daya antar agent dan penentuan struktur organisasi adalah protocol contact net. Pendekatan ini berdasarkan struktur terdesentralisasi di mana agent berperan dalam dua peran, yaitu manajer dan kontraktor.

Dasar pemikiran dari bentuk koordinasi ini adalah jika sebuah agent tidak dapat menyelesaikan masalah yang diberikan kepadanya dengan memanfaatkan sumber daya lokalnya, maka dilakukan dekomposisi masalah ke dalam sub-masalah dan mencoba mencari agent lain yang dapat menyelesaikan sub masalah tersebut. Secara umum penyelesaian sub masalah ini terdiri dari tiga tahap, yaitu pengumuman kontrak oleh agent dinamik. Bahasa pemodelan juga mencakup mekanisme ekstensi sederhana bagi agent melalui relasi inheritance. Tool yang dapat digunakan untuk metodologi ini adalah INGENIAS Development Kit (IDK). Tahapan pengembangan sistem multiagent di WSN menurut Raghavendra adalah sebagai berikut:

1. Identifikasi container dan sumber daya. Pada tahapan ini dilakukan identifikasi perangkat komputasi yang dapat mengeksekusi kode dan mentransmisikan informasi. Perangkat ini berupa sensor beserta perangkat-perangkat tambahan seperti komputer atau fasilitas komunikasi;
2. Identifikasi data yang akan dihasilkan dan menentukan agent yang berperan dalam mengorganisir pemrosesan dan integrasi data yang rumit. Proses

ini akan terus berulang jika masih ada data yang harus diidentifikasi;

3. Menentukan spesifikasi agent yang mengelola adaptasi dinamis jaringan sensor. Tahapan ini mengidentifikasi elemen seperti sumber daya, peran, dan aktor yang memungkinkan terjadinya kegagalan atau perlu di-setup atau di-deploy ulang selama jaringan sensor itu bekerja sehingga dapat ditentukan mekanisme pencarian, evaluasi, dan pergantian elemen tersebut. Tahapan ini juga merancang agent spesifik terkait mekanisme pergantian tersebut yang di dalamnya mencakup agent pengawas yang memonitor elemen elemen tersebut. Proses ini berulang jika masih perlu dikaji sumber daya-sumber daya yang replaceable; dan,
4. Tahap selanjutnya merupakan aktifitas INGENIAS yang tahapan-tahapannya dilakukan secara paralel mulai dari menentukan Platform Independence Module (PIM) yang mendeskripsikan perangkat, agent-agent dan peranannya, informasi yang dipertukarkan, dan interaksinya tanpa mengkaji secara rinci platform yang digunakan. Selanjutnya mengembangkan model yang diperlukan untuk Platform Spesifik Module (PSM) dan pengembangan transformasi yang mendukung penyempurnaan PIM dan PSM yang sudah dimodelkan;
5. Langkah terakhir adalah membangkitkan kode program dari PSM.

Analisis Rame work Pengembangan Sistem Multiagent di Wsn

Bahasa komunikasi agent yang banyak digunakan dan dipelajari adalah Foundation for Intelligent Physical Agents (FIPA) Agent Communication Language (ACL). Menurut Bellefemine Fabio, et al' saat ini FIPA dikembangkan sebagai standar aktifitas IEEE, bernama FIPA-IEEE.

Beberapa fitur pada agent seperti interoperabilitas web service, komunikasi manusia-agent, mobile agent, dan peer-to-peer agent juga dikembangkan.

Java Agent Development Environment (JADE), merupakan agent platform yang mengikuti standar FIPA. JADE menerapkan spesifikasi manajemen agent secara lengkap sebagaimana di definisikan oleh FIPA seperti Agent management System (AMS), Directory Facilitator (DF), Message Transport Service (MTS), Agent Communication Channel (ACC). JADE juga menerapkan FIPA Agent.

Communication Stack mulai dari FIPA ACL untuk struktur pesannya, FIPA SL untuk ekspresi konten pesan, ditambah dukungan untuk interaksi FIPA dan protokol transport. JADE juga mendefinisikan komponen-komponen di luar yang didefinisikan FIPA seperti terdistribusi, fault tolerant, arsitektur kontainer, arsitektur service internal, persistent message delivery, semantic framework, mekanisme keamanan, mobilitas agent, interaksi web service, dan antarmuka grafis.

Beberapa agent framework telah dikembangkan untuk lingkungan desktop. Seiring dengan perkembangan teknologi mobile phone, banyak pengembangan framework serupa untuk diterapkan di lingkungan yang serba memiliki keterbatasan misalnya Java ME CLDC. Menurut Conor Muldoon, et al, saat ini agent framework dibagi ke dalam dua kategori, yaitu untuk lingkungan desktop dan lingkungan CLDC.

Salah satu lingkungan CLDC yang cukup terkenal adalah Agent Factory Micro Edition (AFME). Contoh framework yang serupa adalah JADE LEAP, 3APL-M, SAGE LITE, dan CourgaarME. JADE LEAP, CougaarME, MicroFIPA-OS, dan SAGE Lite merupakan framework pengembangan teknologi agent, namun berbeda dengan AFME, framework tersebut tidak reflective (kemampuan melakukan penalaran tentang dirinya) dan tidak menggunakan bahasa pemrograman abstrak yang berbasis pada teori agent rasional. 3APL-M dengan AFME memiliki kesamaan di mana keduanya memiliki

kapabilitas penalaran, namun 3APL-M tidak memiliki komponen jaringan sedangkan AFME memiliki komponen tersebut melalui message transport service. Ukuran kode program AFME relatif kecil dibandingkan dengan framework lainnya di mana infrastruktur intinya berukuran hanya 77 Kb.

Menurut eksperimen yang dideskripsikan, diperoleh bahwa AFME waktu eksekusinya lebih cepat daripada 3APLM, namun demikian 3APL-M memiliki fitur yang tidak didukung AFME, yaitu 3 APL-M menggabungkan prolog engine di dalamnya.

Rancangan Awal Sistem Multiagent di Wireless Sensor Network

Dalam merancang sistem multiagent langkah pertama yang dilakukan adalah penentuan aturan (rule) dan kebijakan (policy) yang diterapkan di lingkungan sistem. Untuk itu diperlukan konsep organisasi agent, setup arsitektur sistem multiagent di WSN, identifikasi agent-agent yang terlibat dalam organisasi, pendefinisian kapabilitas agent dalam organisasi, dan penentuan platform middle ware dan hardware.

Konsep Organisasi Multiagent Menurut Jorge J, et al, organisasi dipertimbangkan sebagai entitas pertama yang mengajukan goal dan tidak bisa berdiri sendiri karena bergantung pada set agent-agent yang melakukan pekerjaan untuk mencapai goal organisasi. Selanjutnya internal organisasi diperinci dengan goal, workflow, struktur grup, dan sumber daya.

Perlu diangkat karakteristik agent yang sifatnya otonom dengan kata lain agent tidak selalu mengeksekusi pekerjaan yang diminta organisasi. Agent juga bersifat adaptif dan punya potensi untuk mengubah perilakunya sehingga sebuah organisasi harus memperhatikan bahwa agent yang mau diajak kompromi di masa lalu belum tentu dapat berkompromi di masa yang akan datang.

Organisasi membutuhkan agent-agent untuk mengeksekusi pekerjaan dan memproduksi suatu hasil yang diinginkan. Di lingkungan terbuka organisasi dapat saja menemukan agent yang mampu melakukan

pekerjaan yang sama. Kualitas hasil pekerjaan agent bergantung pada beberapa variabel seperti waktu CPU atau ketersediaan memori. Selain itu, agent dapat muncul atau hilang beberapa saat selama waktu hidup organisasi sehingga organisasi harus menyimpan jejak rekam agent yang selanjutnya organisasi dapat secara efektif melakukan kerja sama dengannya.

Untuk menentukan kinerja agent, didefinisikan dua parameter pengukuran agent, yaitu trust dan effectivity. Trust direpresentasikan sebagai riwayat aktivitas agent dan fungsi utilitas, yaitu memahami agent dalam melakukan tugasnya dengan baik. Fungsi utilitas menghasilkan sebuah estimasi trust yang memberi penghargaan kepada sebuah agent berdasarkan riwayat kinerjanya di masa lalu. Riwayat ini terdiri dari apakah agent memenuhi kompromi dengan organisasi atau sebaliknya. effectivity direpresentasikan sebagai nilai dari pekerjaan yang telah dilakukan, kualitas hasil pekerjaan, dan kondisi eksekusi. Kualitas perlu dihitung dengan fungsi utilitas lain dengan mendefinisikan organisasi yang mempertimbangkan parameter terkait dengan goal-nya.

Untuk mencapai organisasi yang sehat perlu didefinisikan fungsi manajemen untuk menambah, menghapus, dan memonitor agent di organisasi. Agent yang diterima di komunitas harus menyediakan informasi yang baik ke agent lain sesuai yang diminta. Tingkat kualitas informasi diukur dengan skema voting dengan mempertimbangkan dokumen terkait agent yang sebelumnya diketahui. Anggota organisasi membantu mendeteksi agent yang tidak mampu bekerja sama.

Prinsip dari mekanisme ini adalah "Semua berjalan baik jika sebagian besar anggota berfikir semua juga baik". Melalui mekanisme ini dihasilkan biaya komputasi rendah. Untuk mencapai goal-nya organisasi harus menjamin bahwa:

1. tidak ada anggota yang tidak berkolaborasi ketika diminta,
2. tidak ada anggota dengan produktifitas rendah,
3. anggota tidak boleh salah menggunakan sumber daya organisasi sehingga mencegah

agent lain melakukan tugasnya, dan

4. agent yang suka konflik dikeluarkan dan tidak pernah diterima lagi.

Konsep lain yang perlu dipertimbangkan dalam merancang agent adalah workflow yang meliputi cara agent melakukan registasi ke organisasi, cara melakukan pemutusan hubungan kerja organisasi dengan agent, dan cara memonitor organisasi. Pada proses registrasi ke organisasi, penerimaan anggota baru ke organisasi ditentukan dua faktor: (1) factor kebutuhan organisasi, yaitu organisasi dapat menerima agent dengan keterampilan atau implementasi baru untuk pekerjaan yang sudah berjalan sehingga meningkatkan kinerja, dan (2) kepercayaan dan efektifitas.

Proses pemutusan hubungan kerja dengan agent dimulai dari anggota organisasi. Anggota organisasi pertama kali menyelesaikan kompromi-kompromi dengan anggota lain. Setelah kompromi dilepas, agent dapat meninggalkan organisasi dengan aman. Pemecatan agent dilakukan apabila agent memiliki perilaku yang tidak sesuai dengan yang diharapkan organisasi. Pengambilan keputusan pemecatan ini bergantung pada set laporan yang dikumpulkan selama agent bekerja di organisasi. Memonitor organisasi berarti bertanya kepada setiap anggota organisasi tentang anggota lainnya.

Mekanisme ini bertujuan untuk mendeteksi apakah agent masih hidup dan mendeteksi apakah agent memenuhi kompromi-kompromi yang sudah ditentukan organisasi. Laporan-laporan yang terkumpul disimpan dan digunakan untuk mendeteksi apakah agent harus dikeluarkan dan apakah agent ini diterima kembali di masa depan. Prinsip "Goal akan dicapai ketika agent-agent yang suka berkonflik dikeluarkan dan agent-agent yang mau berkolaborasi tetap dipelihara" diterapkan dalam organisasi.

Setup Arsitektur Organisasi Multiagent di WSN Pada tahap ini dilakukan penentuan peran agent yang akan diterapkan di WSN. Peran ini mengadopsi usulan yang

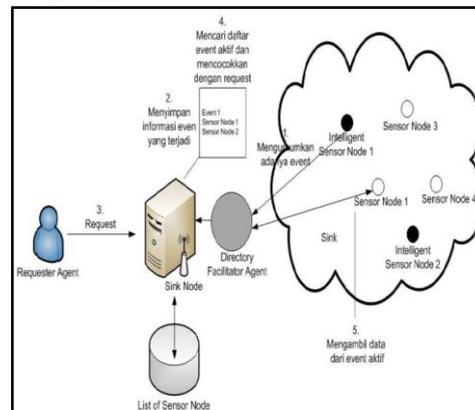
mendefinisikan peran agent ke dalam 7 peran sebagai berikut:

1. intelligent sensor adalah sensor yang memonitor lingkungan. Ketika mendeteksi sebuah event, sensor ini melakukan broadcast ke base station (sink);
2. data centric sensor adalah sensor yang melakukan penginderaan pada lingkungannya dan mengumpulkan data penginderaan di dalam dirinya (lokal);
3. agent otonom adalah agent yang diletakkan di bagian lingkungan yang melakukan penginderaan dan melakukan aksi pada lingkungannya setiap saat. Agent ini melakukan prediksi terhadap apa yang akan dideteksi di masa yang akan datang;
4. mobile agent (MA) adalah tipe agent spesial yang dapat mengeksekusi dirinya secara otonom untuk bermigrasi. Ketika diberangkatkan, agent ini bermigrasi dari sensor node satu ke sensor node lain untuk melakukan pemrosesan data secara otonom. MA mengumpulkan data hop by hop di dalam rentang sensor aktif dan dalam threshold yang ditentukan. Data yang terkumpul diaku mulasi ukurannya;
5. directory facilitator (DF) adalah entitas tempat agentagent mengumumkan kapabilitasnya, melakukan koordinasi aktifitas-aktifitas agent, dan memenuhi permintaan agent-agent yang di bawah koordinasinya. Peran DF adalah menyimpan identitas provider (ID) beserta kapabilitasnya di database lokalnya. Ketika requester meminta preferensi, DF mencocokkannya dengan kapabilitas provider dan memberikan respon kepada requester berupa set event yang sesuai dengan permintaannya;
6. requester agent adalah entitas yang melakukan request informasi dari

sistem untuk kepentingan pengguna;

7. provider agent adalah tipe agent yang menyediakan informasi di dalam sebuah lingkungan aktif melalui agent-agent superiornya. Provider adalah intelligent sensor node dan data centric sensor yang terhubung dengannya. Pada langkah pertama provider harus mendaftarkan keberadaannya ke DF. Lalu, provider ini mengumumkan event aktif.

Arsitektur yang mendeskripsikan peran agent tersebut ditunjukkan pada Gambar 1 berikut ini.



Gambar 1. peran-peran agent

Kapabilitas Agent dalam Organisasi Salah satu kapabilitas agent yang harus didefinisikan adalah kemampuannya dalam melakukan pembelajaran. Salah satu pembelajaran yang dapat dilakukan oleh agent adalah proses deteksi outlier dan proses pengambilan keputusan secara otonom. Salah satu isu penting yang perlu dipertimbangkan dalam WSN adalah outlier. Ketika digunakan untuk memonitoring lingkungan, agent-agent yang diimplementasikan di WSN dapat digunakan untuk melakukan deteksi dan klasifikasi suatu event atau bahkan diintegrasikan dengan sistem control tertentu. Oleh sebab itu, diperlukan realibilitas dan jaminan kualitas informasi yang diterima WSN. Untuk menghindari pengiriman data yang tidak akurat ke base station, diperlukan implementasi analisis data secara real time yang dilakukan di setiap sensor node. Karena kondisi

lingkungan yang keras dan tidak menentu, sensor node dapat saja tidak berfungsi dengan baik atau menghasilkan data yang tidak akurat ke base station. Selain itu, keterbatasan sumber daya seperti pemrosesan yang terbatas, penyimpanan data yang terbatas, bandwidth yang terbatas, dan otonomi agent dapat juga menyebabkan ketidakakuratan data mentah yang dibaca oleh sensor node. Inilah dikenal dengan outlier di mana sensor node membaca data yang menyimpang dari pola data yang biasanya. Kemungkinan sumber outlier ini berasal dari noise, data error yang disebabkan karena kegagalan hardware, atau serangan malicious.

Sebagian besar teknik tersebut membutuhkan kapasitas memory yang besar untuk penyimpanan data dan memerlukan konsumsi energi yang cukup besar. Selain itu teknik tersebut menyebabkan overhead pada komunikasi dan tidak mendukung kapabilitas komputasi terdistribusi. Teknik tersebut dikategorikan ke dalam pendekatan statistical-based, nearest-neighbour-based, clustering-based, classification-based, dan spectral-decomposition-based. Statistical-based dibagi ke dalam dua metode, yaitu parametrik dan nonparametrik. Kedua metode ini menitik beratkan kepada distribusi probabilitas untuk melakukan karakteristik data set atau time seri ketika melakukan evaluasi sampel data mentah secara statistik.

Metode *nearest-neighbour* menitik beratkan pada matrik yang menghitung jarak antara obyek atau sample dengan interpretasi geometrik. *Euclidian norm* dan *Mahalanobis distance* adalah pilihan umum yang banyak digunakan pada metode ini untuk atribut yang *univariate* dan *multivariate*. Pada teknik berbasis *clustering*, *data set* dikelompokkan ke dalam *cluster* yang memiliki atribut yang sama. Sebuah obyek dapat dikatakan sebagai *outlier* jika obyek tersebut berada di luar *cluster* yang ditentukan. Metode berbasis *classification* menitik beratkan pada teknik *machine learning* yang mencoba memahami *dataset* yang tersedia. Kelemahan teknik ini adalah secara komputasi terlalu kompleks untuk diterapkan secara *real time* di *sensor node*. Terakhir,

pendekatan *spectral-decomposition* yang menitik beratkan pada analisis komponen-komponen dasar untuk mengidentifikasi mode-mode normal dari perilaku *dataset*. Pendekatan ini menghasilkan kompleksitas komputasi yang tinggi.

Untuk menentukan metode yang tepat yang dapat diterapkan di *sensor node* yang memiliki sumber daya terbatas, diperkenalkanlah teknik mendeteksi *outlier* secara *real-time* lokal dan teknik mengakomodasi skema-skema yang dapat diterapkan kode program yang ditanam di *sensor node*, dapat menjalankan *thread* otonom yang kooperatif, dan diimplementasikan dalam bentuk *mobile agent*. Pendekatan yang diusulkan adalah pendekatan statistika *univariate* yang diterapkan pada sensor yang melakukan penginderaan terbatas. Pendekatan ini mengasumsikan bahwa variable lingkungan tunggal diobservasi melalui pembacaan sensor dan digunakan untuk menentukan batasan *threshold* untuk operasi kontrol.

Penyimpangan dari batasan ini akan diindikasikan sebagai *outlier*. Metodologi ini biasanya diterapkan dalam bentuk grafik *Shewhart control*. Batas atas (Δ_u) dan batas bawah (Δ_l) pada grafik *Shewhart* adalah garis kritis untuk meminimasi nilai *false outlier* dan *missed detection*. Teori hipotesis statistik dapat digunakan untuk memprediksi suatu nilai *false outer* dan *missed detection*. Misalkan variable monitoring z , yang memiliki deviasi dari titik tengahnya, \hat{z} , adalah *additive error* dan nilai z mengikuti distribusi Gaussian $N(\hat{z}, \sigma^2)$ dengan standar deviasi σ ; lalu probabilitas P di mana z berada dalam interval yang ditentukan dihitung berdasarkan:

$$P\{z < (\hat{z} - c\alpha/2 \sigma)\} = P\{z > (\hat{z} + c\alpha/2 \sigma)\} = \alpha/2 \dots \dots \dots (1)$$

$$P\{(\hat{z} - c\alpha/2 \sigma) \leq z \leq (\hat{z} + c\alpha/2 \sigma)\} = 1 - \alpha \dots \dots \dots (2)$$

di mana $c\alpha/2$ adalah *standard normal deviation* dan α adalah tingkat signifikansi dimana menentukan tingkat *tradeoff* antara nilai *false outlier* dan *missed detection*. Beberapa nilai yang umum digunakan pada *standard normal deviation* meliputi $c\alpha/2 = \{1.0; 1.5; 3.0\}$.

Kemampuan *agent* lainnya yang harus dipertimbangkan adalah kemampuan berkomunikasi dengan lingkungannya tanpa dibantu oleh tutor atau guru untuk pembelajarannya, dikenal dengan *Reinforcement Learning* (RL). Tujuan RL adalah menemukan *policy* di mana memilih sebuah aksi pada suatu *step-time* yang mengantarkannya untuk mendapatkan *reward* terbaik dari lingkungannya. Setiap aksi yang *agent* lakukan, lingkungannya merespon dengan sebuah *reward* yang menunjukkan efektifitas aksi di setiap *step-time*.

Pada aksi yang dilakukan, *agent* tidak menganut benar atau tidaknya aksi yang dilakukan. RL adalah subarea *machine learning* yang menitikberatkan kepada cara sebuah *agent* mengambil aksi di lingkungannya. Di sini *agent* melakukan maksimalisasi pemikiran tentang *reward* untuk jangka panjang. Pada setiap langkah, RL memilih beberapa aksi yang mungkin dilakukan dan menerima *reward* dari lingkungan atas aksi spesifik yang dilakukannya. Aksi terbaik yang harus dilakukan di beberapa *state* tidak pernah diketahui sehingga *agent* harus mencoba beberapa aksi dan urutan aksi yang berbeda serta belajar dari pengalamannya.

RL biasanya dideskripsikan sebagai *Markov Decision Process* (MDP) yang terdiri dari sebuah *agent*, set *state* yang mungkin S , set aksi-aksi yang mungkin $A(S)$ untuk semua *state* S , dan sebuah fungsi *reward* $R(s,a)$ yang menentukan *reward* yang diberi lingkungan atas aksi yang dilakukan *agent*. Fungsi *policy* π mendeskripsikan bagaimana *agent* belajar pada beberapa *time-step* t . *Policy* optimal didefinisikan sebahai π^* . Fungsi *value* $V(s,a)$ mendefinisikan *reward* total yang diharapkan ketika melakukan aksi a pada *state* s jika untuk mencapai *state* berikutnya *agent* mengikuti *policy* optimal π^* . *Agent* harus belajar untuk memperoleh *policy* ini.

Kemampuan terakhir yang perlu dipertimbangkan oleh *agent* terkait *mobile agent* adalah *itinerary planning*. *Itinerary* adalah rute yang dilalui selama migrasi *mobile agent*. Teknik-teknik *itinerary* yang dapat dipertimbangkan menurut meliputi *Local Closest First* (LCF), *Global Closest*

First (GCF), *Genetic Algorithm* (GA), *Near-Optimal Itinerary Design* (NOID), dan *Tree-Based Itinerary Design* (TBID). Parameter yang perlu diukur dalam menentukan kinerja *itinerary planning* adalah *data aggregation cost*, *response time*, dan *estimated network lifetime*. Perlu dipertimbangkan pula pendekatan *multiple mobile agent itinerary planning* (MIP) yang diajukan, sebagai alternative single *mobile agent itinerary planning* (SIP).

Masalah MIP dibagi ke dalam empat tahapan, yaitu pemilihan *visiting central location* (VCL) untuk setiap *mobile agent*, penentuan *node-node* sumber setiap *mobile agent*, penentuan urutan kunjungan ke sumber, dan *iterative framework*. Salah satu algoritma yang dapat digunakan dalam *itinerary planning* adalah *Genetic Algorithm* (GA). Algoritma ini memberikan kinerja yang lebih baik daripada LCF dan GCF. Namun demikian, GA mengkonsumsi waktu komputasi yang cukup lama karena algoritma ini memulai eksekusinya dengan vektor solusi yang random yang memerlukan waktu relatif lama. Meskipun optimasi global akan diperoleh, namun penggunaan algoritma ini bukan solusi yang ringan untuk *sensor node* yang memiliki energi terbatas.

Penentuan Middleware dan Hardware

Mobile agent didefinisikan sebagai entitas *software* yang me-enskapsulasi perilaku dinamis dan punya kemampuan untuk bermigrasi dari satu *node* komputasi ke *node* lainnya untuk menyelesaikan tugas-tugas terdistribusi. *mobile agent* dapat mendukung pemrograman WSN pada tingkat aplikasi, *middleware*, dan *network*. *mobile agent* didukung oleh *Mobile Agent System* (MAS) yang menyediakan API untuk mengembangkan aplikasi berbasis *agent*, *agent server* yang mengeksekusi *agent-agent* dengan menyediakan layanan dasar seperti migrasi, komunikasi, dan pengaksesan sumber daya *node*. Salah satu *middleware* yang akan dipertimbangkan dalam pengembangan system *multiagent* di WSN adalah *Mobile Agent for Sun Spot* (MAPS), dan AFME seperti yang dijelaskan di Subbab IV.

MAPS merupakan *framework* inovasi berbasis Java yang dikembangkan di atas teknologi Sun Spot untuk memfasilitasi pemrograman berorientasi *agent* di WSN. MAPS dirancang berdasarkan kebutuhan:

1. *component-based lightweight agent server architecture* untuk mencegah model kerja sama dan *concurrency* yang berat;
2. *lightweight agent architecture* untuk mengeksekusi dan migrasi *agent* secara efisien;
3. *core service* yang minimal seperti migrasi *agent*, penamaan *agent*, komunikasi *agent*, pengaksesan sumber daya sensor (*sensor, actuator, flash memory*, dan radio);
4. *plug-in based architecture extension*;
5. bahasa Java untuk mendefinisikan perilaku *mobile agent*.

MAPS dapat diinteroperasikan dengan JADE *framework*. JADE-MAPS *gateway* telah dikembangkan untuk memungkinkan *agent* JADE berinteraksi dengan *agent* MAPS. Meskipun JADE dan MAPS menggunakan *platform* Java, keduanya menggunakan metode komunikasi yang berbeda. JADE mengirim pesan berdasarkan standar FIPA menggunakan spesifikasi *Agent Communication Language (ACL)*, sedangkan MAPS membuat komunikasi pesannya sendiri berbasis *event*. Oleh sebab itu JADE-MAPS *gateway* memfasilitasi pertukaran pesan antara *agent* MAPS dengan JADE. *Platform* inter komunikasi MAPS dengan JADE memungkinkan pengembangan lebih lanjut aplikasi terdistribusi berbasis WSN di mana JADE digunakan pada *base station/koordinator/host* dan MAPS diletakkan pada *sensor node*.

Merupakan *agent platform* yang merujuk pada J2ME MIDP ringan *open source* dan menerapkan *Agent Factory Framework* yang ada. AFME ditujukan untuk sistem *wireless pervasive* sehingga secara spesifik AFME tidak dirancang untuk WSN. Namun, karena dukungan J2ME kepada *platform* sensor Sun Spot, AFME dapat diadopsi untuk pengembangan aplikasi WSN berbasis *agent*. AFME berbasis pada paradigma *Belief-Desire-*

Intention di mana di dalamnya *agent-agent* mengikuti siklus *sense-deliberate-act*. AFME mendeskripsikan *agent* melalui *mixed declarative/imperative programming model* bernama *Agent Factory Agent Programming Language (AFAPL)*, berbasis formalisme *logic belief* dan *commitment*. AFAPL digunakan untuk *me-encode* perilaku *agent* dengan menentukan aturan-aturan yang mendefinisikan kondisi-kondisi dimana komitmen diadopsi.

Berbeda dengan MAPS dan AFME yang tidak mendukung migrasi yang kuat, *isolate (weak) migration*, di mana kedua platform ini hanya mendukung migrasi *state* obyek atau data, bukan kode, *Mobile Agent Platform for Sun Spot (MASPOT)*, memungkinkan migrasi kode *agent (strong migration)* sehingga berdampak pada tidak hanya penghematan energi, namun juga adanya penghematan pada sumber daya *memory*. Jika ingin menerapkan platform *mobile agent* di MAPS dan AFME, kode-kode *agent* yang melakukan tugas spesifik harus diinstall terlebih dahulu di *sensor node*.

Sedangkan dengan MASPOT hal ini tidak perlu dilakukan karena MASPOT menyediakan solusi terkait dengan migrasi kode program. Eksperimen yang dilakukan menunjukkan jumlah energi yang dihabiskan selama migrasi kode adalah sebesar 0.03% dari baterai *sensor node*. Selain itu, MASPOT hanya menggunakan 1.5% saja alokasi *flash memory* dari *node* Sun Spot. Namun, *strong migration* mengkonsumsi energi yang lebih besar daripada *weak migration* yang diterapkan oleh MAPS dan AFME.

Perangkat yang digunakan dalam penelitian ini adalah Sun Spot (*Small Programmable Object Technology*). Sun Spot terdiri dari 32-bit ARM9 CPU, 512K memori, 2 Mb *flash storage* dan *wireless networking* berbasis pada ChipCon CC2420 yang mengikuti standar 802.15.4, terintergasi dengan antena, dan beroperasi pada 2.4GHz sampai 2.4835GHz ISM *unlicensed bands*. IC-nya terdiri dari 2.4 GHz RF *transmitter/receiver* dengan *digital direct sequence spread spectrum (DSSS) baseband modem* dengan dukungan MAC. Sensor yang sudah terdapat pada Sun Spot

meliputi sensor temperatur, sensor cahaya, dan *accelerometer* 3D. Sensor lain dapat ditambahkan melalui pin analog I/O yang sudah disediakan. Sensor ini cukup ideal untuk penerapan *multiagent* yang dapat dipasang *middleware* MAPS, AFME, dan MASPOT.

SIMPULAN

Banyak penelitian yang dilakukan terkait usulan penyelesaian masalah yang dihadapi di WSN, namun sebagian besar penelitian tersebut terlalu fokus pada aspek tertentu dan siftnya parsial seperti fokus di optimasi komunikasi, fokus di *cross layer protocol*, atau di *mobile agent* tanpa memperhatikan aspek arsitektur sistem secara keseluruhan dan terintegrasi. Paper ini mendeskripsikan usulan pemodelan system *multiagent* pada WSN yang menjadi solusi untuk efektifitas dan efisiensi kerja WSN yang memiliki keterbatasan, khususnya energi. Dalam pengembangan *multiagent* di WSN perlu memperhatikan aspek-aspek penting yang dapat mendukung kinerja WSN seperti aspek arsitektur, aspek *middleware*, aspek *hardware*, aspek *itinerary planning*, dan aspek kapabilitas *agent-agent* dalam sistem.

Pada aspek arsitektur perlu dirancang struktur organisasi *multiagent*. Di sini diidentifikasi organisasi, peran, dan aturanaturan yang mengikat *agent-agent* dalam menjalankan fungsinya. Organisasi selanjutnya diperinci dengan penerapan konsep *goal*, *workflow*, struktur grup, dan sumber daya. Pada aspek *middleware* perlu diidentifikasi API yang didukung oleh perangkat WSN spesifik. *Middelware* ini menyediakan antarmuka kepada *service-service* yang disediakan di WSN secara *loose coupling* sehingga fokus penelitian ditekankan pada aspek kapabilitas *multiagent*. *Middleware* yang perlu dipertimbangkan dalam penelitian ini adalah AFME yang dijalankan pada perangkat Sun Spot. *Middleware* ini mendukung arsitektur BDI yang merupakan arsitektur yang akan diangkat dalam penelitian ini.

Terakhir, kapabilitas *agent* perlu diidentifikasi seperti kemampuan dalam *itinerary planning*, kemampuan pembelajaran khususnya mendeteksi *outlier*,

dan pengambilan keputusan secara otonom terkait dengan aksi-aksi yang dilakukan. Pekerjaan selanjutnya adalah pengembangan secara lebih terperinci kapabilitas *agent* yang akan terlibat sebagai anggota dalam organisasi di lingkungan WSN. Kapabilitas tersebut adalah melakukan pembelajaran dan menentukan keputusan melakukan aksi tertentu secara otonom dengan menerapkan arsitektur BDI yang dikombinasikan dengan MDP. Selain itu, mempertimbangkan protokol komunikasi standar antar *agent*, mengadopsi *Service Oriented Architecture* (SOA) atau mikro SOA, sehingga *agent* dengan *platform* berbeda dapat salingberkomunikasi adalah langkah selanjutnya yang sangat penting.

DAFTAR PUSTAKA

- Stefano G, Francesco Aiello, Alesio Carbone, Giancarlo Fortino, "Java-Based Mobile Agent Platform for Wireless Network Sensor", the International Multiconference on Computer Science and Information Technology, 2010, pp. 165–172.
- Min Chen, Sergio Gonzalez, and Victor C. M. Leung, "Application and Design Issues For Mobile Agent in Wireless Sensor Networks", University of British Columbia, IEEE Wireless Communication, 2007.
- X. Wang, M. Chen, T.Kwon, H.C. Chao, "Multiple Mobile Agents Itinerary Planning in Wireless Sensor Network : Survey and Evaluation", IET Communications, 2010, www.ietdl.org.
- Ramon Lopes, Flavio Assis. "MASPOT: A Mobile Agent System for Sun SPOT", LaSiD-Distributed Systems Laboratory DCC-Department of Computer Science UFBA-Federal University of Bahia. Salvador, Brazil, Tenth International Symposium on autonomous Decentralized Systems, 2011.

- Michael Wooldridge, “*Introduction to MultiAgent System*”, John Wiley and Sons LTD, 2002
- Charalampos K, Aristides Mpitiopoulos, Damianos G, Grammati Pantziou, “*Effective Determination of Mobile Agent Itineraries for Data Aggregation on Sensor Networks*”, IEEE Transactions On Knowledge And Data Engineering, Vol. 22, No. 12, December 2010.
- Giancarlo F, Stefano Galzarano, Raffaele Gravina, Antonio Guerrieri, “*Agent-based Development of Wireless Sensor Network Applications*”, DEIS– University of Calabria. 2010.
- Paulo Gil, Amâncio Santos, and Alberto Cardoso, “*Dealing With Outliers in Wireless Sensor Networks: An Oil Refinery Application*”, IEEE transactions on control systems technology, 2013.
- Bellefemine Fabio, Giovanni Caire, Dominic Greenwood. “*Chapter 2 Agent Technology Overview. Developing MultiAgent System with JADE*”, John Wiley and Sons LTD, 2007.
- Conor Muldoon, Geogory M. P. O’here, Rem W. Collier, Michael J. O’Grady, “*Toward Pervasive Intelligence: Reflection on The Evolution of Agent Factory Framework*”, Springer Science and Business Media LC, 2009.
- Jorge J. Gómez-Sanz, Juan Pavón, “*Implementing Multi-Agent Systems Organizations with INGENIAS*”, Dep. Sistemas Informáticos y Programación Universidad Complutense de Madrid.
- Khin Haymar S.H, YoungSik Choi, Jong Sou Park, “*The Multi Agent System Solutions for Wireless Sensor Network Applications*”, Springer-Verlag Berlin Heidelberg, 2008.
- Raghavendra V. Kulkarni, Anna Förster, Ganesh Kumar Venayagamoorthy, “*Computational Intelligence in Wireless Sensor Networks: A Survey*”, IEEE Communications Surveys & Tutorials, Vol. 13, No. 1, First Quarter 2011.