センサネットワークの限定されたリソースにおける セキュアな協調システムの開発

Design of Secured Cooperative Multi-Agent System in Limited Resources for Sensor Network

大林真人 † 西山裕之 ‡ 溝口文雄 ‡
Makoto OBAYASHI, Hiroyuki NISHIYAMA and Fumio MIZOGUCHI

† 東京都立産業技術研究所 情報科学グループ
Information Science Group, Tokyo Metropolitan Technology Industrial Research Institute
‡ 東京理科大学 理工学部
Faculty of Science and Technology, Tokyo University of Science
obayashi.makoto@tiri.metro.tokyo.jp, nisiyama@ia.noda.tus.ac.jp, mizo@ia.noda.tus.ac.jp

本研究では，センサネットワークを用いて双方向性を持つユビキタスな環境を構築するためのセンサノード協調システムを開発する．協調システムは，複数ロボットの動的協調システムとして開発されたマルチエージェントの概念を適用させることにより，各ノードが持つ限定されたリソース上において動作するミドルウェアと動作記述言語として開発される．さらに，ミドルウェア内部にセキュリティ機能を組込むことにより，各種同期および動的サービス提供における安全性を確保する．

1 はじめに

センサネットワークは，多種多様なセンサを搭載した通信機能を持つ多数のノードによって構成されるシステムであり，実環境の様々な状態情報を容易に取得することを可能とするものである [3]．ここで，システムの目的が，広範囲に配置されたセンサ情報を取得するだけであるならば，センサネットワーク内に存在する特定のステーションへデータをルーティングするだけで，要求される機能を満たすことができるだろう．しかしながら，使用者が能動的にセンサネットワーク内の特定ノードに対してサービスを要求する場合には，双方向性が必要とされる．また，異なる機能を持ったノードによって構成されるヘテロジニアスな環境では，実世界のイベントに応じた相互処理を実現するために，動的に増減する不特定多数のノードを正しく認識させ，協調や競合の解消を行うためのフレームワークが必要となる．従来，このような問題に対しては，マルチエージェントによる解決手法が挙げられる．分散して動作するセンサノード間の協調やサービス解決，リソースの競合解消を実現するフレームワークとしては，マルチエージェントシステムによるファシリテータやメディエータの手法が適切であり，これらを実現する優れた手法も提案されている．また，Agent UML[6][2] は，UML による記述を拡張することによってエージェント技術の設計を行うことを実現している．また，オブジェクト指向言語を拡張させ，エージェント指向言語を実装する手法も提案されている [5]．しかしながら，従来の手法をセンサネットワークに適用するに際しては，使用可能な計算機リソースが大幅に制限されるため，協調や交渉の手続きに際して，一般の OS や開発言語を使用することが困難である．また，個人情報や生体情報に基づいた協調動作が行われる場合には，一連の手続きにおいて通信パケットの暗号化や認証の機構が要求される．このため，センサネットワークに適合した手法およびシステムを実装する必要が生じる．しかしながら，通常の計算機で使用される暗号および認証のアルゴリズムを，センサノードが持つ限られたリソース内で実装することは事実上不可能である．

この問題に対して，我々は，センサノードの限定されたリソースで動作するマルチエージェントシステムを開発し，ノード間における自律協調動作を実現させる．本研究によって開発されるシステムは，我々の先行研究において自律ロボットや分散デバイスの動的協調システムとして開発されたロボット協調動作記述言語 [4] におけるフレームワークをセンサネットワークに適用することによって実現される．また，

アドホックネットワークによって構築されるセンサネットワーク上の通信パケットに対して，暗号化および認証の機構を導入することによって，エージェント間協調におけるセキュアな相互通信を実現する．そして，評価実験を通じて，本システムの有効性を検討する．

2 設計方針

2.1 動作記述言語の開発

本研究において開発されるマルチエージェント記述言語”TinyMRL”(Tiny Multi-agent Robot Language) では，一つのセンサノードの動作は述語の集合によって構築される．図 1 に記述例を示す．各述語は，GHC(Guarded Horn Clause) による記述形式と同様に，述語名，条件節，実行節の 3 部から構成される．述語は，述語名および引数の数によって分類される．すなわち，同じ述語名と同じ数の引数を持つ全ての述語は同一のグループに分類され，2 つ以上の同じグループの述語が同時に呼び出されることは無い．呼び出される述語は，条件節における定義式が，呼び出し元による引数の値およびセンサノードの内部状態が一致するものだけが呼び出される．ここで，条件に適合する述語が複数存在する場合には，条件節内における定義式の数にしたがって優先度が決定され，最も多い条件を持つ述語が最高優先度として処理される．また，最高優先度となる述語が複数存在する場合には，その中から一つの述語が無作為に選択される．これにより，各述語の実行節の最後に自身の述語を再帰的に呼び出すことによって，その同じグループに属する述語集合によって構成される一連の処理が継続される．同じ述語グループに属するものは一つの状態を表現し，再帰的に自己を呼び出すことによって，同じ状態での動作を保持する．また，他のノードからの通信や各種のイベントにしたがって，自己を他の状態へ遷移させるときには，条件節に特定のイベントに適合するルールを記述した述語を定義し，実行節の最後に他のグループに属する述語の呼び出しを行えばよい．状態の動作の終了は，実行節内部での述語呼出を行わずに述語定義を完了することによって実現できる．このとき，どの述語も実行待ち状態にならないため，述語の実行の終了と同時に一連の状態遷移動作は完全に終了する．また，TinyMRL 処理系は，分散サービス解決におけるファシリテータを介した様々なエージェント動作をシステムコールとして実装しているため，自律的

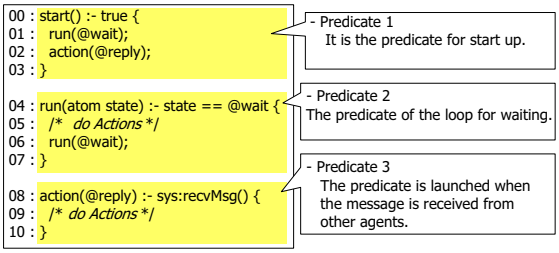

図 1: TinyMRL のコード記述例

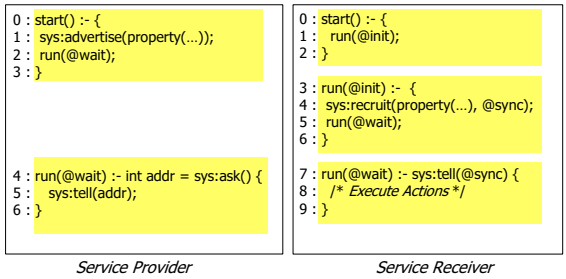

図 2: ファシリテータを介した分散サービス解決の動作記述例

な協調動作を簡便に記述することが可能である．図 2 は，ネットワーク中に存在するファシリテータを介したサービスの「徴発」動作を行う記述例である．

2.2 セキュリティシステム

センサネットワークにおけるセキュリティは，個人情報や，その生体情報を扱う場合に際して，特に必要とされる機能である．しかしながら，限られた計算機資源の中で実装するためには，通常の計算機とは異なる手法を用いる必要があることが指摘されている [1]．TinyMRL 処理系が提供するセキュリティ機能は，データの暗号化による傍受の防止，セマンティックセキュリティの実現，データ認証であり，本研究におけるエージェント言語の処理系と融合される．ここで，開発者自身が設置した全てのセンサノードは信頼することが可能であり，悪意のある攻撃者によるクラッキングの試みは，他のノードによるアドホックネットワークへの参加を通じて行われるものと仮定する．データの暗号化および復号化，データ認証に使用される全ての鍵は，信頼できる全てのノードが共通して所持する秘密鍵から生成される．TinyMRL によって記述されたエージェント $Agent_a$ が，他の特定のエージェント $Agent_b$ への通信を行うための以下のシステムコールを実行すると，エージェントの動作は TinyMRL 処理系へと移行する．

`syscall:sendMsg(`$Agent_b$`, <Message>)` (1)

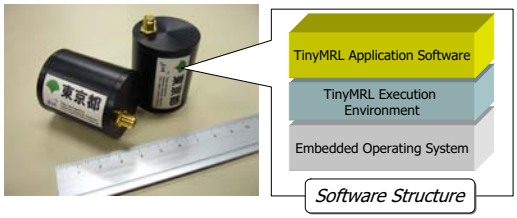

図 3: 使用するハードウェアとシステム構成

このとき，TinyMRL 処理系によって実行される $Agent_a$ および $Agent_b$ における相互通信の内容は以下の様に表現することが可能である．

$$Agent_a \rightarrow Agent_b : N_a, \texttt{<Message>} \quad (2)$$

$$Agent_b \rightarrow Agent_a : \{\texttt{<Message>}\}_{<K_{ba}>}, \texttt{MAC}(K_{ba}, N_a||E_b) \quad (3)$$

$$E_b = \{\texttt{<Message>}\}_{<K_{ba}>} \quad (4)$$

2 式および 3 式における N_a は，$Agent_a$ として定義されたセンサノードで生成されたノンスを意味する．本研究におけるシステムは，エージェント間の通信を実行する毎にノンスを生成する．ノンスとは，無作為のビット列であり，メッセージの順序性やデータの新規性を保障するために用いられる．データ通信システムコールの呼び出しと共にノンスが生成されると，TinyMRL 処理系は，内部に存在するデータ領域に使用されたノンスを格納する．この後，他のエージェントから受信した返答メッセージおよび MAC 内に，自身が生成したノンスが含まれていることを確認することによってメッセージの順序性を保障し，エージェント間協調プロセスを起動させる．

3 実装

3.1 使用するデバイス

次に本研究によって開発するマルチエージェント言語および言語処理系の実装について述べる．本研究において，我々は CrossBow 社によって開発されたセンサーネットワークデバイス MOTE をセンサノードとして使用する．図 3 左は，本研究で使用するために，様々なセンサによって MOTE を拡張したデバイスである．このデバイスは ATMEL 社製の 8bit の CPU と 4kbyte の RAM を持ち，315MHz 帯での無線通信を行う．本研究による我々のシステムは，図 3 右に示される構成であり，最小セットの実装イメージにおいて，必要 ROM サイズは 12Kbyte，必要 RAM サイズは 2.8Kbyte となっている．

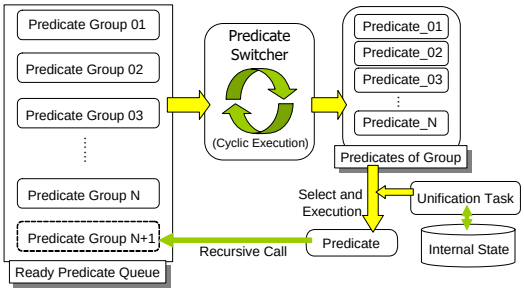

図 4: 述語の実行におけるスケジューリング処理の流れ

3.2 述語の実行スケジューラ

エージェントの動作は，複数の述語の定義と実行によって決定されるが，ノードの内部状態および実環境によって複数の述語を並列に動作させることも必要となる．このとき，動作する述語は，エージェントの内部状態と外部からの入力によって動的に決定されることとなる．各ノードの動作を決定する述語のスケジューラ動作を図 4 に示す．これは，各タスク毎にスタック領域を確保しないシングルスタックのシステムにおける実装であり，非常に小さなリソースにおける計算機システムに特化した手法である．述語は，述語名および引数の数によってグループ化される．他の述語からの呼び出しによって，実行可能状態に遷移した述語グループは，実行待ちキューに投入される．このとき，周期的に動作するタスク切替器が，実行待ちキューの先頭に位置する述語グループを取り出す．次に，取り出された述語グループ内部に格納された述語群と，エージェントの内部状態および外部からの入力による状態変数を参照して，各述語に定義されている実行条件に適合する述語が選択され，実行に移る．このとき，グループ内の全ての述語が実行条件に適合しない場合には，再び述語グループが実行待ちキューの最後に投入されることとなる．また，実行する述語の最後に，自身の述語を再帰的に呼び出すことによって，自身の属する述語グループを待ちキューの最後に接続することが可能となる．

4 評価実験および考察

次に，本研究によって開発されたシステムの有効性を実験によって確認する．評価項目として，我々のマルチエージェントシステムにおける通信処理性能と，エージェント言語の記述性について測定し，考察を行う．図 5 は，本システムにおけるアドホックネットワークによるルーティングを介した通信速度

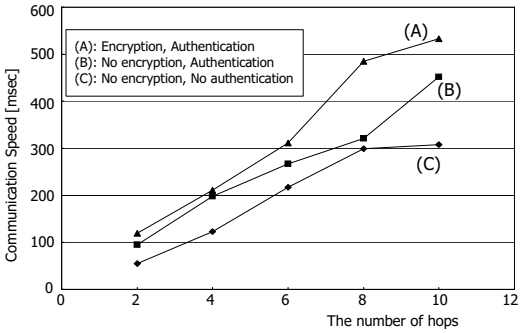

図 5: TinyMRL を使用したノード間における通信速度

の処理結果である．実験は，次の 3 つの設定，(A) パケットの暗号化と認証処理，(B) 平文通信と認証処理，(C) 平文通信および認証処理無し，についてデータを取得した．グラフに示されるように，セキュリティを付加した遅延は 0.2〜0.1 秒であり，ハードリアルタイムを要求される箇所での使用は不可能であるが，ユビキタスコンピューティングなどへの応用には十分な能力であると考えられる．

図 6 は，本研究におけるエージェント言語の記述性を示している．実験は 4 人の被験者によって行われており，熟練したプログラミングスキルを持つ被験者 2 人 (Subject1 および Subject2）と手続き型言語の初歩のみを知識として持つ被験者 2 人（Subject3 および Subject4）によって構成される．記述性の実験は，複数のプログラミング課題を被験者に提示し，それらを記述するまでに要した時間を計測した．課題は (1) デバイス制御，(2) イベント処理，(3) 通信，(4) ファシリテータを介したエージェント協調動作の 4 つである．なお，グラフでは，本研究によるエージェント言語による記述時間 ($T_{<agent>}$) とネイティブ言語による時間（$T_{<native>}$) の比 ($T_{<agent>}/T_{<native>}$) としてプロットされている．グラフより，全ての被験者および課題について，1 を下回っており，本研究によるエージェント言語の動作記述の簡便性が確認できる．特に，その傾向は，課題 3 および 4 について顕著であり，通信及びエージェント協調の動作記述性に優れていることが確認できる．

5 まとめ

本研究において，我々は，センサネットワークのノード間によって必要とされる動的な協調問題を実現することを目的としたエージェント言語 “TinyMRL” および処理系を開発した．これは，センサノードの動作をルール形式による単位動作の列挙として記述

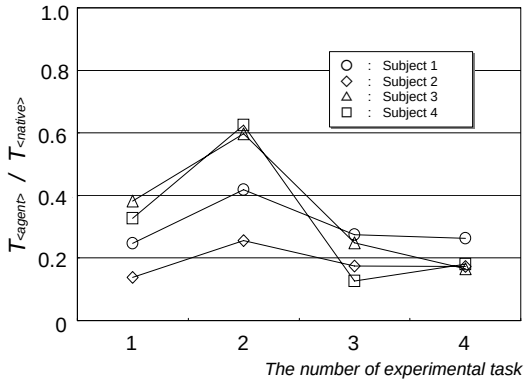

図 6: 本研究におけるマルチエージェント言語の記述性評価

することが可能であるだけでなく，マルチエージェントの機能である分散サービスの動的な発見・提供を行う動作をシステムコールとして実装している．これにより，自律動作するセンサノード間の協調動作を簡潔に記述することを可能とするものである．また，限られたリソース上で動作するセキュリティシステムを構築し，エージェント処理系と融合させることによって，無線アドホックネットワークにおける安全な通信を実現した．そして，我々の開発したシステムの有効性を実証するために，その実行速度についての性能と，動作定義の簡易性を計測し，その有効性を示した．

参考文献

[1] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, J.D.Tygar: “SPINS: security protocols for sensor netowrks,” Proceedings of the 7th annual international conference on Mobile computing and networking, pp.189-199, July 2001, Rome, Italy.

[2] B.Bauer, J.Muller and J.Odell: “Agent UML: A Formalism for Specifying Multiagent Software System,” in Proc. Agent-Oriented Software Engineering, LNCS 1957, Springer-Verlag, 91-103, 2000.

[3] B.Warneke, M.Last, B.Liebowitz, and K.Pister: “Smart dust: Communicating with a cubic-millimeter computer,” IEEE Computer, pages 44-51, January 2001

[4] 西山裕之，大林真人，大和田勇人，溝口文雄: “ロボット間協調を容易に実現する並列論理プログラミング言語の設計”，ロボット学会誌，vol.19，No.5，pp.620-631，2001.

[5] J.Lind: “Issues in Agent-Oriented Software Engineering,” Proc. Agent-Oriented Software Engineering, LNCS 1957, Springer-Verlag, 45-58, 2000.

[6] J.Odell, V.M.Parunak and B.Bauer: “Representing Agent Interaction Protocols in UML,” in Proc. Agent Oriented Software Engineering, LNCS 1957, Springer-Verlag, 121-140, 2000.