cogma: 動的ネットワーク環境における組み込み機器間の連携用ミドルウェア

河口信夫[†,††] 稲垣康善[†††]

情報機器の小型・低価格化により，これまでは考えられなかった多様な機器によるネットワークが実現しつつある．しかし，現状では，ほとんどの機器は特定用途に限られた通信機能のみを持ち，他の機器との連携や，動的環境への対応は考慮されていない．今後，より多様性を増すであろうネットワーク機器に対し，動的環境に適用し，拡張可能なシステムを容易に構築できる枠組が望まれている．本研究では，一時的に会議や展示会に持ち込まれる機器や，家庭やオフィスの情報機器においても，事前の設定なしに，また特定のサーバを必要とせずに利用できるアドホックネットワークを構築し，その上で機器間の連携アプリケーションを容易に実現するミドルウェアを提案する．本ミドルウェアでは，移動ソフトウェアの技術を導入し，実行時のソフトウェアインストールを可能にする．この機能によって，一度設置した組み込み機器でもバージョンアップや設定変更を容易に行うことができる．

cogma: Cooperative Gadgets for Mobile Appliances

Nobuo Kawaguchi[†,††] and Yasuyoshi Inagaki[†††]

Recent advancement of technology enables the network connections for various kinds of appliances. However, most of these appliances only have fixed set of the communication functions. In this paper, we propose a middleware named “cogma” for mobile appliances. It enables the easy development of cooperative applications among various kinds of appliances in mobile environment. By virtue of dynamic code mobility based on mobile agent technology, our middleware enables the on-demand code installation.

1. はじめに

近年の情報機器の普及により，様々な情報機器がいたるところに埋め込まれ，利用されるようになりつつある．例えば車の中にはエアコン，カーナビ，ステレオ等の制御のために，多数のコンピュータが埋め込まれている．また，家庭の中にも，テレビ，冷蔵庫，FAX，エアコンなど，多数のコンピュータが導入されている．しかしながら，これらの埋め込み機器は，基本的に単独での利用しか考えられていない．すなわち，車に埋め込まれたステレオやカーナビを，人が持ち込んだノートブック PC や PDA と連携させて，PC 上の音楽データをステレオから再生したり，PDA の持つ位置情報をカーナビに転送したりすることはほとんど考えられていない．

† 名古屋大学大型計算機センター
Computation Center, Nagoya University
†† 名古屋大学統合音響情報研究拠点 (CIAIR)
Center for Integrated Acoustic Information Research, Nagoya University
††† 名古屋大学大学院工学研究科
Graduate School of Engineering, Nagoya University

このように，すでに多数の埋めこまれた組み込み機器が存在するにもかかわらず，組み込み機器間で連携がとられていない理由としては，機器をできる限り安価に実現するために，最小限度のハードウェア及びソフトウェアしか持たないこと，および，組み込み機器間での連携用のミドルウェアが整備されていないことなどが挙げられる．今後，より多様性を増すであろうネットワーク機器に対し，動的環境に適用し，拡張可能なシステムを容易に構築できる枠組が望まれている．

一方，我々はすでに，ノート PC 等においてアドホックネットワークに対応したモバイルエージェントシステム MAGNET[1] を構築してきた．このシステムでは，無線通信等で接続されたノート PC 間で，モバイルエージェントを用いてアドホックネットワークを手軽に構築できる．さらに我々は様々なアプリケーションを MAGNET 上のモバイルエージェントを用いて実現してきた[2)~4)]．MAGNET のモバイルエージェントはシンプルな機構を持つため，複数のエージェントを用いたプロトコルの実現や，ファイル管理システム，グループ管理システム等の分散協調ソフトウェアを容易に構築す

ることができる．

本研究では，一時的に会議や展示会に持ち込まれる機器や，家庭やオフィスの情報機器，様々なセンサ等に対し，事前の設定なしに，また特定のサーバを必要とせずに利用できるアドホックネットワークを構築し，その上で機器間の連携アプリケーションを容易に実現するミドルウェア cogma を提案する．

cogma (Cooperative Gadgets for Mobile Appliances) は，移動する情報機器 (Mobile Appliances) のための協調ソフトウェア群 (Cooperative Gadgets) である．本研究での「移動」する情報機器は，携帯電話や PDA といった人と共に持ち運ばれるデバイスに加え，展示会や会議室などで動的に構成される情報機器や家庭内での容易なレイアウト変更に対応する情報家電といった半固定型の情報機器をも含んでいる．これら半固定型の情報機器は，これまでは移動しない情報機器とみなされ，様々な設定を初期導入時に行なうことが当然とされてきた．しかし，近年の情報機器の氾濫によって，機器の接続設定は高度な知識を必要とするものになり，また個々の設定は単純であったとしても，多数の機器の設定は繁雑な作業となる．一方，機器の初期導入時は一種の動的環境と考えることも可能である．本研究では，半固定の機器であっても，移動速度や変更の頻度が低い動的環境とみなし，これらの機器に対応した動的なコンフィギュレーションを実現する．これによって，多数の情報機器の導入時にも，機器間の動的な連携が可能になり，人による設定作業を最低限とすることができる．

本ミドルウェアでは，移動ソフトウェアの技術を導入し，実行時のソフトウェアインストールを可能にする．この機能によって，一度設置した組み込み機器でも，バージョンアップや設定変更を容易に行うことができる．本ミドルウェアによって，様々な情報機器を設定なしに持ち寄るだけで機器間の連携が可能なアプリケーションを容易に構築できる．具体的な連携アプリケーションとしては，プロジェクタやスクリーン，ライト，オーディオ設備，複数の PC などを連携させたスマート会議室や，温度センサ，光センサ，エアコン，ライト，テレビ，ステレオなどを連携させるスマートリビングなどが挙げられる．多くの組み込み機器や情報機器が本ミドルウェアに対応することによって異種組み込み機器間での連携が可能になるため，新たなアプリケーションが創造される可能性がある．本ミドルウェアでは，近年急速に立ち上がりつつあるアドホックネットワーク技術を連携用の基礎的なインフラとして用い，さらに動的なソフトウェアインストールの導入によって，様々な状況に，高度に適応可能なシステムを実現している．

以下，2 章において組み込み機器間の連携を容易に行うために必要なソフトウェアシステムについて検討し，3 章以降で，我々が開発しているミドルウェア cogma の設計と実装について述べる．

2. 組み込み機器間の動的連携

本研究で対象とする組み込み機器は，ネットワークを介して通信が可能であるものとする．また，アドホックネットワークを積極的に利用するためには，リンクの断続検知や他ノード検知の機能が必要である．

様々な機器に埋め込まれたシステム間で連携を行うためには，以下の機能を持つことが望ましい．

(A) **自分自身の基本機能を把握**
機器固有の情報は，基本的には機器自身しか知ることができない．そこで，各機器は自分の機能に関する情報を保持し，必要に応じ，他のシステムからの問い合わせに応答することが望ましい．さらに，機器の持つ情報のみで制御が可能であることが望ましい．

(B) **動的に機能の修正・追加が可能**
一般に，組み込み機器を設置後に，システムの再インストールを行うことは繁雑な作業となる．多くの機器はファームウェアアップデートと呼ばれる手法で，アップグレードが可能であるが，専用の PC やソフトウェアの利用が必要であったり，システムの再起動が必要になる．そこで，システムの動作中にも基本的なソフトウェアの修正・追加がネットワークを介して可能なことが望ましい．

(C) **外部からの機能制御が可能**
ネットワーク通信が可能な組み込み機器においては，機器の持つ情報を外部から取得したり，機器の制御が外部から可能であることが望ましい．これによって，インタフェースを持たない組み込み機器でも外づけのデバイスによるインタフェースの実現が可能になる．

(D) **可能な限り少ないリソースで動作**
組み込み機器は，一般に限られたメモリ，CPU しか持たない．このような状況でも動作するためには，最低限の機能とそれ以外の機能を分離して実現する必要がある．

上記の機能により，接続される組み込み機器間では，互いの機能の動的な認識により，連携動作が可能になる．さらに，外部からのモニタリングが可能となる．ただし，組み込み機器のハードウェアの制約を考慮すると，できる限りシンプルな枠組で実現されることが望ましい．

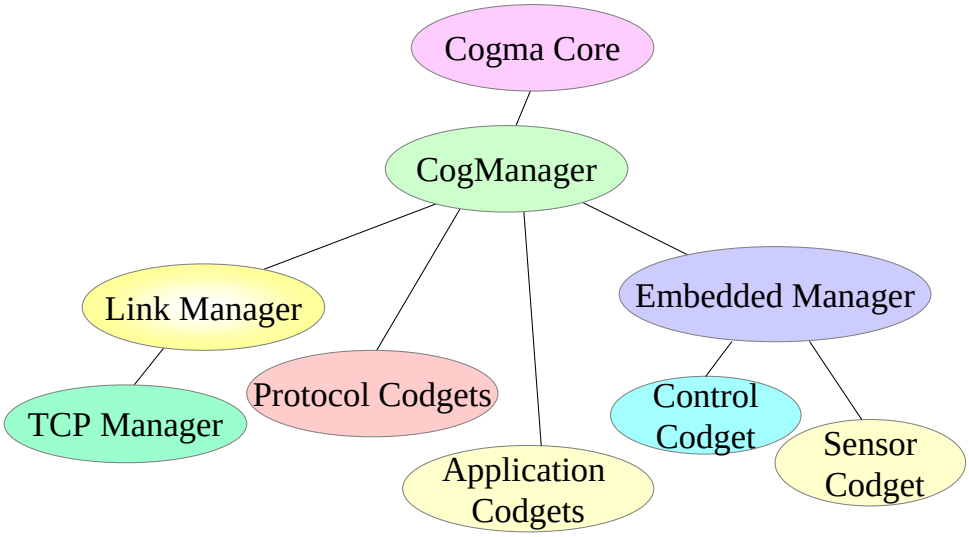

図 1 システムの階層構造

また，本稿では機器の状態変化の粒度を 5 秒に 1 回程度とする．これは，ネットワーク状況等の環境変化をポーリングによって検知していることが主な理由である．

3. 組み込み機器間連携用ミドルウェアの設計

本節では，我々が開発している組み込み機器間連携用ミドルウェア cogma(Cooperative Gadgets for Mobile Appliances) の設計について述べる．前節で挙げた要件を満たすために，cogma では移動ソフトウェアの技術を採用した．ほとんどのコンポーネントを移動可能に設計することによって，(B) の要件を満たすことができる．また，(A) の要件を満たすために，移動ソフトウェアを用いた機器のリソース管理を行う．さらに，リソース管理を行うソフトウェアモジュールは，上位のモジュールマネージャによって管理される．(C) の要件については，外部と内部のインタフェースとなるモジュールとして実現することとした．(D) の要件は，様々な管理機構をユーザインタフェースと分離して実現し，組み込み機器では，インタフェースに関するモジュールを省くこと等によって省メモリ化を実現している．また，組み込み機器での動作を可能とするために，現時点では Java 仕様として Personal Java 1.1.3 を採用している．

以下に cogma を構成するモジュールを示す (図 1)．

- **基幹システム (Cogma Core)**
  ソフトウェアの動的管理を担当
- **Codget 管理 (CogManager)**
  各 Codget を管理
- **通信リンク管理 (Link Manager)**
  各通信リンクを監視，Codget を送受信
- **組み込み機器管理 (Embedded Manager)**
  センサやスイッチ等の機器固有のモジュールを管理
- **プロトコル Codget 群 (Protocol Codgets)**
  Codget 間通信プロトコルを実現
- **アプリケーション Codget 群 (Application Codgets)**
  各種連携アプリケーションを実現

CogManager 以下の各機能は，すべて Codget と呼ばれる基本コンポーネントで実現されているため，実行時の動的な入れ替えが可能である．また，プロトコル Codget の配布によって，様々な組み込み機器に適したプロトコルを動的に実現することが可能になる．また，図 2に示すように，NotePC や組み込み様々な機器上で本システムは動作する．

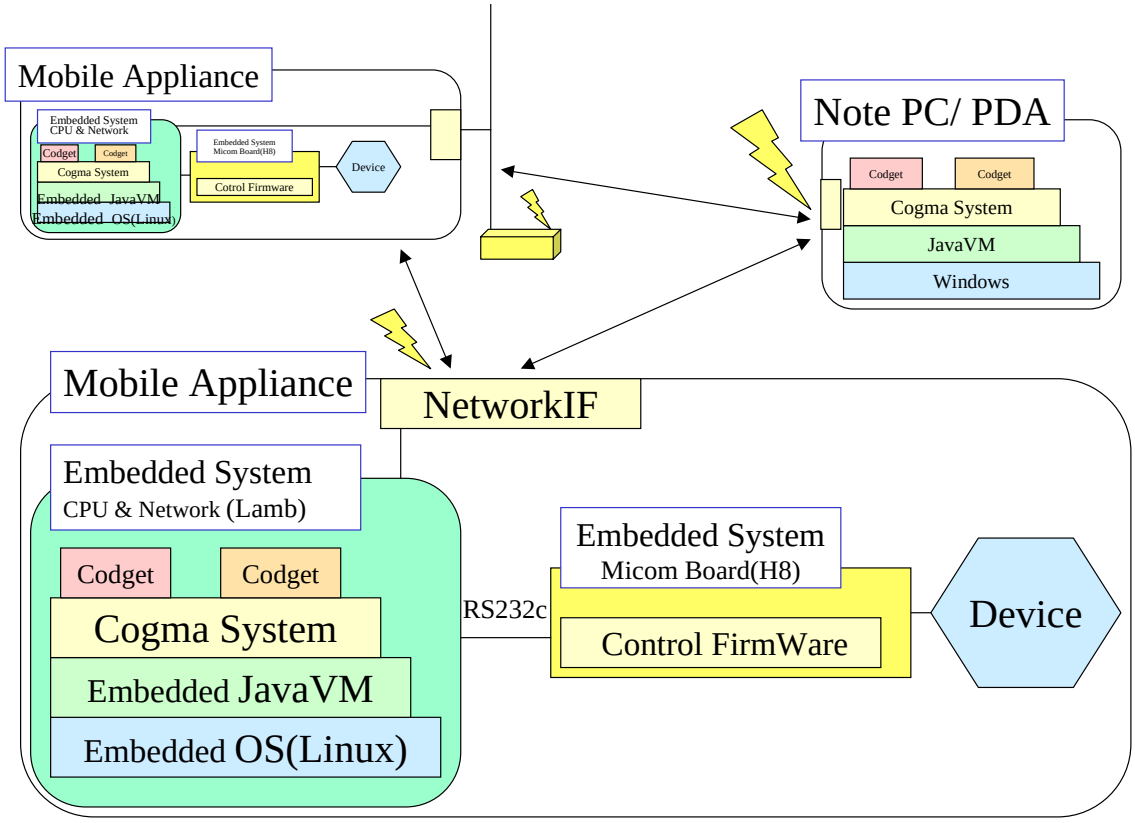

図 2 システム間の関係

3.1 基本モジュール単位： Codget

cogma では， Codget と呼ぶソフトウェアモジュールを主に用いてシステムを構成している． Codget は外部から実行を停止させ，シリアライズによって永続化が可能である． Codget は移動エージェントシステムにおける移動エージェントに対応しているが，リソース管理や，プロトコル，マネージャの実現にも用いられているため，必ずしも移動するわけではない．しかし，動的なシステム更新[5] を行うために，永続化可能なモジュールとして実現されている．また， Codget が移動する場合でもプログラムコードは必ずしも同時に送付されず，必要に応じて送付が行われる．各 Codget は起動されたホストに対応したユニークな ID (CodgetID) を持つ．CodgetID は移動したり，永続化されても変わらないため，次節で説明する Codget 間通信で用いられる．

3.1.1 Codget 間通信

cogma では，簡略化のために，ホスト間通信や Codget 間通信のための機構をシステムとして特別に用意していない．その代わり， Codget 間の連携手法として RMI を単純化した手法を独自に実現し採用している．本手法では，同一 ID を持つ Codget が同じノード (CogManager) に登録されると，既存の Codget の`onSame`という関数が，登録された Codget を引数として呼び出される．これによって，既存の Codget から，登録された Codget を直接操作することを可能としている．また，この機能のため，同一ノード上では同じ ID を持つ Codget は同時に 2 つ以上は存在しない．

例えば，複数のノード間で情報収集を行う Codget を例として挙げる．この Codget は，一度，ローカルで登録された後に，複製を残して他ノードへ移動する．Codget が他ノードで情報収集を行った後に，ローカルノードへ帰ってくると，ローカルに残っていた Codget の`onSame`が呼び出されるため，帰ってきた Codget から必要なデータを取り出すことができる．このように，同一 Codget 間の情報同期が非常に簡単に行える点が本手法の特徴である．また，同一ノード内での通信と，別ノード間での通信がほぼ同一の手続きによって実現できる．

3.1.2 プロトコル Codget

互いに相手を知らない Codget 間の通信はプロトコ

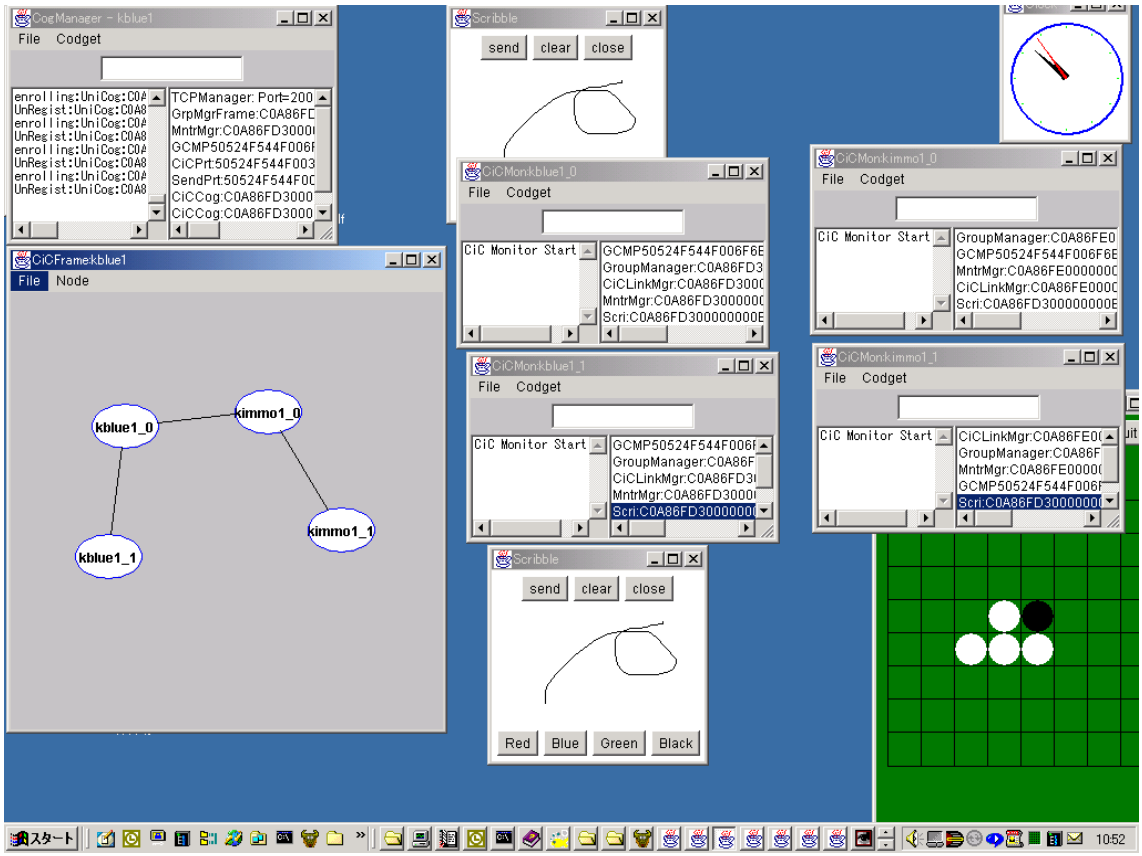

図 3 cogma 動作画面 (CiC の実行中)

ル Codget によって実現される．ユニークな ID を持つ Codget と異なり，プロトコル Codget では TCP のポート番号のようにプロトコル毎に ID を定義する．一般に，プロトコルを実装するためには，プロトコル仕様に従ったプログラムを記述する必要がある．本手法では，プロトコル仕様ではなく，プロトコル Codget をライブラリとして配布することによって，プロトコルの実装がライブラリ呼び出しのみで実現できる．この機能が，通信が必須な連携ソフトウェアの実現を容易にしている．

3.2 Codget の管理

Codget を管理する CogManager は Codget に加え LinkManager や EmbeddedManager 等のマネージャも管理する．さらに，3.5節に記述する CiC(Cogma in Cogma) によって CogManager 自身を階層的に起動することが可能であり，システム構成を安全に外部から変更したり，同一ホスト上の CogManager 間にファイアウォールを導入し，安全なノードの構築が実現できる．

3.3 通信リンクの管理

cogma では，無線リンクや赤外線通信等，複数の通信リンクを用いることが可能である．リンクを管理するために，リンクマネージャが存在する．リンクマネージャはリンクの状態変化の通知，リンクを通じた Codget の移送を行う．ただし，リンクマネージャが管理するのは 1 ホップの通信のみである．マルチホップの通信の実現には，MAGNET[1] で提案されているように，ネットワークプロトコルを実現する Codget が用いられる．

新たな機器の追加や機器の移動によって通信リンクに変化が生じた場合，リンクマネージャは通信リンクの変化の通知を希望している Codget に状態変化を知らせ，各 Codget はリンクの変化に応じた動作を各自で行なう．例えば，リンク状態に基づきネットワークを構成している Codget の場合には，ネットワークの再計算を行なう．

3.4 電源断・電源再投入への対応

組み込み機器では，任意の時点での電源断や電源再投入がおこり得る． cogma はソフトウェアモジュールの多くを Codget で実現しているため，システム状態が変化した時に各 Codget を永続化しておくことによって，不意の電源断に対しても，直前の永続化情報を用いて復

帰することを可能とする．

3.5 論理ネットワークコンフィギュレータ

cogma では，アドホックネットワークの上に，さらに論理的なネットワークを構築することが可能である．この機能を我々は CiC(Cogma in Cogma) と呼んでいる．CiC では，Codget 管理モジュール (CogManager) を Codget として実現している．これによって，単一の cogma システム内に仮想的に複数の cogma システムを階層的に動作させられる．さらに，CiC 用の CogManager 間には，CiC 内での論理的な通信リンク (CiCLink) を導入しており，外部から CiCLink を制御することが可能になる．CiC によって，物理的に接続されている (ネットワーク上で通信可能な) 機器を論理的に接続することが可能になる．CiC の制御は CiCProto と呼ばれるプロトコル Codget によって実現されており，外部機器からのネットワークを介した制御が可能になっている．そのため，ユーザインタフェースを持たない組み込み機器のネットワーク接続設定を外部機器や外部ソフトウェアによって行なえる．

また，論理ネットワークと 3.6節のグループとを組み合わせることによって，様々なアプリケーションに対して専用の論理ネットワークを実現することができる．これは，ある種のセマンティックネットワークの実現とみなすことができる．

3.6 セキュリティとグループ管理

外部からネットワークを用いて制御可能なシステムでは，セキュリティへの対応が問題となる．cogma ではグループ管理[4] に基づく権限管理を行う．各ノードは複数の権限を持つグループに属しており，管理権限を持つグループ内で移送された Codget と，それ以外の Codget では，システム操作に対する権利が異なる．これによって，グループ外のシステムからの攻撃を回避できる．

各 cogma システムには，必ず グループ管理機構が存在しており，各ノードのグループ参加状況を把握している．各グループの定義そのものも，移動ソフトウェアによって実現されているため，リンク状態や位置，時間といった，状況に依存した多様なグループの実現も可能である．

アドホック環境におけるグループの認証には，グループを定義しているソフトウェアコードに基づく手法を検討している．これによってそれぞれのグループそのものの保持が，グループへの参加を証明することになる．また，耐タンパハードウェアの利用[6] も検討している．

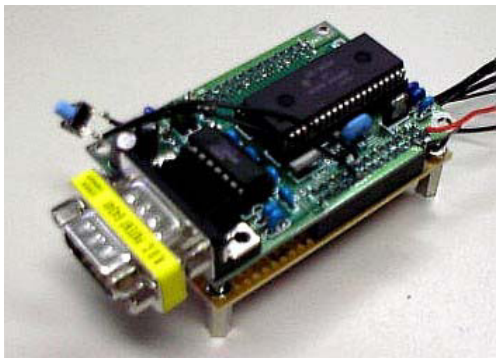

図 4 マイコンモジュール (AKI-H8)

図 5 組み込み PC モジュール (LAMB-EM-01)

4. ハードウェア環境

様々な組み込み機器に直接 CPU を埋め込むことは困難であるため，cogma では複数の IO ポートを持つマイコンシステムを通じて様々な機器を制御することとした．機器制御用に 16bit マイコンの H8/3664 を採用し (図 4)，制御 CPU として PCMCIA 無線カードを装備した LAMB-EM-01 (5x86,133MHz: 図 5) を採用した．この間は RS232C ケーブルで接続される (図 2)．H8 マイコンでは赤外線万能リモコンが実現可能なので，リモコン制御可能な機器が制御できる．LAMB-EM-01 上では，Linux および Personal Java1.1.3 が動作している．cogma は PC 上や PDA(SHARP Zaurus) でも動作するため，これらの間での連携ソフトウェアをプロトタイプとして構築している．

ノート PC 等のふんだんにメモリや CPU が存在する環境とは異なり，組み込み機器上の cogma システムでは，リソースが限られているため，システムの最低限必要な部分のみ実装し，ユーザインタフェース等の画面表示の機能は省かれている．

現在の cogma システムの動作画面を，図 3に示す．この画面では，2 台のノード上で動作している cogma

上にそれぞれ2つづつの CiC が動作している。これらの CiC は CiCFrame と呼ばれる論理ネットワークコンフィギュレータから起動されており，各ノード上で動作する Codget の起動もこの画面から行なうことができる．また，論理ネットワークコンフィギュレータは，実際の論理ネットワークとは独立に起動，終了が可能である．例えば，他のノード上で起動することも可能であり，この画面と同様の接続関係を取得できる．

5. 関連研究

関連する枠組として， Jini[7] や Hive[8] が挙げられる．しかし，これらは共に Java の RMI を利用しているため，アドホックネットワークへの対応やマルチリンクへの対応が行えない．また， Jini は Lookup サーバによって，サービス利用者とサービス提供者が明確に分かれている点で，各ノードへ Codget を動的に移動可能な cogma と異なる．

動的連携を行うシステムとしては， STONE[9] や Ja-Net[10] が挙げられる． Ja-Net は CyberEntity 間のインタラクションによって実現される動的環境であるが，自己組織化の実現に重点が置かれ，プロトタイプ実装向けではない． STONE は動的サービス発見に基づき，サービス合成を行うことが可能な枠組であるが，機能ユニットとサービスリゾルバが独立しており，各機器が対等ではない点で本システムと異なる．

また， Smart Space Lab[12] における SONA[11] では，情報家電間の自律的な連携を目指し，機器発見のためのルックアップサービスに Jini，機器の動的なソフトウエア導入のために Aglets[14] を用いている． Aglets は機器発見の機能を持っていないため， Jini の導入により動的環境への適応が可能であるように思われるが，各機器の機能が対等ではないため， Smart Space Lab のように，組み込みの部屋環境では利用可能であるが，移動先のような真にアドホックな環境での利用ができない．

一般的に，各機器が対等な設計は効率面で問題がある．しかし，アドホックネットワークを考慮した場合，システムの基本機能が分散していると，動的サービスを実現することができなくなってしまう．そこで， cogma では各機器は対等な機能を持つように設計されている．ただし，基幹機能以外のユーザインタフェースや機器固有の機能に関しては，機器によって実現を変えている．

MAGNET[1] は， cogma の基礎となるシステムであり，ここで得られた経験が cogma では生かされている．cogma と MAGNET の最も大きな違いは， MAGNET は NotePC 等の必ず画面やネットワーク，キーボードやマウスが存在することを前提として構築されたのに対し， cogma では，それらが必ずしも存在することを考えていない点である．この点は，システム設計の様々な場面に影響を与えている．また， MAGNET はその時の最新の JDK を採用しているが cogma では，組み込み機器での利用のために，すでに古くなりつつある Personal Java を用いている．また， MAGNET では，試験用にエミュレーション環境を用いており，すべての通信が監視できる神の視点からの制御が可能であった．しかし， cogma では，エミュレーションは用いず，外部からの監視手法として， CiC を導入した．これは，エミュレーション環境では再接続や設定変更のために，再起動が必ず必要になるが，組み込み機器では再起動をできる限り少なくしたいことや，起動パラメータの入力がほとんど不可能であるためである．分散ソフトウエア開発において，通信のモニタリングは必須の機能であり，組み込み機器上でのソフトウェアの動作を監視できる CiC の機能は， cogma を高効率なソフトウェア開発のプラットフォームとして利用できることを示している．

6. まとめ

組み込み機器間で連携アプリケーションを容易に構築するためのミドルウェア cogma を提案し，実装している． cogma は， MAGNET[1] で得られた移動ソフトウェアの技術と経験を用い，組み込み機器に特化した機能を実現したシステムであり，シンプルな構成にも関わらず，動的なアプリケーションを簡便な枠組で実現することができる．

現在は基本的な機能を持つ cogma 1.0 のリリース準備を行っている状況である．また，次リリースとしては，セキュリティ機能の強化，分散シナリオの導入，組み込み機器制御の標準化等を検討している．また，既設家電[13] への対応へも検討したい．

なお， cogma に関する各種ドキュメント，開発状況については， cogma プロジェクトホームページ[15] にて公開しているので参照されたい．

謝辞 本研究は，情報処理振興事業協会より委託を受けた財団法人ソフトウェア工学研究財団が実施した「平成 13 年度高度情報化支援ソフトウェアシーズ育成事業」での支援を受けたものである．

参考文献

1) Nobuo Kawaguchi,Katsuhiko Toyama, and Yasuyoshi Inagaki: MAGNET: Ad-Hoc Network System based on Mobile Agents , Spe-

cial Issue for Mobile Agents for Telecommunication Applications , Computer Communication, Vol.23, pp.761–768(2000).
2) Nobuo Kawaguchi, Yasuyoshi Inagaki: Multi-Link Ad-Hoc Communication System based on Mobile Agents, Proc. of SNPD'01, pp.311–318(2001).
3) 杉浦俊一，河口信夫，外山勝彦，稲垣康善：モバイルエージェントを用いたファイル管理システムの提案, 情報処理学会, DiCoMo2000, pp.145–150(2000).
4) 宮越勇樹，河口信夫，外山勝彦，稲垣康善：アドホックネットワークにおける柔軟なグループ管理手法: モバイルエージェントを用いた実現, 情報処理学会, DiCoMo2000, pp.1–6(2000).
5) 河口信夫，外山勝彦，稲垣康善：モバイルエージェントに基づく動的拡張可能なソフトウェアシステム, 情報処理学会，夏のプロシン報告集, pp.71–78(1999).
6) 春木洋美，河口信夫，稲垣康善：耐タンパハードウェアを用いたモバイルエージェント保護手法, 情報処理学会, DiCoMo2001, pp.43–48(2001).
7) Sun Microsystems, Inc. : "Jini Architecture Specification Version 1.1," http://java.sun.com/jini/specs/(2000).
8) Nelson Minar, Matthew Gray, Oliver Roup, Raffi Krikorian , and Pattie Maes,"Hive: Distributed Agents for Networking Things," ASA/MA'99, (1999).
9) 南正輝，森川博之，青山友紀：ネットワークサービスシンセサイザのデザイン，信学技報, IN2000-192, pp.1–8(2001).
10) Tomoko Itao, Tetsuya Nakamura, Masato Matsuo, Tatsuya Suda, Tomonori Aoyama, "The Model and Design of Cooperative Interaction for Service Composition," 情報処理学会, DiCoMo2001,pp.7–12(2001).
11) 青木崇行，中澤 仁，徳田 英幸：自律的な情報家電制御機構の構築，情報処理学会，情報家電コンピューティング研究グループ第 1 回ワークショップ,(2001).
12) T. Okoshi, S. Wakayama, Y. Sugita, S.Aoki, T. Iwamato, J.Nakazawa, T. Nagata, D. Furukusa, M. Iwai, A. Kusumoto, N. Harashima, J.Yura, N. Nishio , Y. Tobe, Y. Ikeda and H. Tokuda, "Smart Space Laboratory Project: Toward the Next Generation Computing Environment", in Proceeding of IEEE Third Workshop on Networked Appliances(IWNA2001), (2001).
13) 丹 康雄：既設家電機器を活用するホームネットワーク，情報処理学会，情報家電コンピューティング研究グループ第 1 回ワークショップ,(2001).
14) Danny B. Lange, Mitsuru Ohshima, Programming and Deploying Java Mobile Agents with Aglets, Addison Wesley(1998).
15) cogma プロジェクト ホームページ: http://www.cogma.org/