OUKA
Osaka University Knowledge Archive

Title	イベント駆動型データベースの高度応用に関する研究
Author(s)	寺田, 努
Citation	大阪大学, 2003, 博士論文
Version Type	VoR
URL	https://hdl.handle.net/11094/583
rights	
Note	

Osaka University Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

Osaka University

工38746

イベント駆動型データベースの高度応用に関する研究

2003年7月

寺田　努

イベント駆動型データベースの
高度応用に関する研究

２００３年７月

寺　田　　努

内容梗概

近年，マイクロエレクトロニクス技術や無線通信技術の発達により，日常生活のあらゆる部分にコンピュータが入り込み，あらゆるものが電子化されるようになった．PDAやノートパソコンを持ち歩くモバイルコンピューティング，家電にインターネット接続などさまざまな機能をもたせた情報家電といった言葉は一般的になり，近年ではコンピュータを常に身につけて生活をおくるウェアラブルコンピューティングや，環境内のあらゆるものにコンピュータを埋め込んでどこでもコンピュータを使えるようにするユビキタスコンピューティングといった概念も登場している．

常に自分用の端末を持ち歩き，あらゆる場所にコンピュータが埋め込まれるようになると，従来のように目的に応じて端末を取り替えるのではなく，端末の機能が状況に応じて変わるという枠組みが有効であるといえる．このように，あらゆるものが自律的に動作し，連携しながら状況に応じて動作を変更するためには従来と異なる新しいコンピューティングの枠組みが必要となる．

そこで本研究では，さまざまなサービスを提供するための新たな枠組みとして，イベント駆動型データベース(アクティブデータベース)の概念を用いたサービス実現手法を提案する．イベント駆動型データベースでは，ECAルールとよぶ処理ルールの集合でシステムの動作を記述するため，ユーザは要求を因果的に記述でき，ルールの追加/削除によりシステムのカスタマイズや機能追加も容易になる．本研究ではさまざまな環境を想定し，それらの環境にアクティブデータベースの枠組みを適用することで多様なサービスが提供できることを示す．また，ECAルールの無限連鎖などアクティブデータベースの枠組みを用いることで起こる問題点を明らかにし，その解決方法を提案する．

本論文は全6章から構成され，その内容は以下のとおりである．まず第1章において序論を述べ，第2章においてモバイルコンピューティング環境のためのアクティブデータベースであるAMDS(Active Mobile Database System)について述べる．AMDSは移動体の接続・切断といった事象をECAルールで取り扱えるようにしたアクティブデータベースシステムであり，モバイル環境においてルールベースのサービスを行う際の基盤となるシステムである．本研究では，AMDSの設計および実装に加え，分散配置されたECAルール群が異常動作を起こすかどうかを検出する手法を提案する．さらにAMDSのためのアプリケーション構築環境およびシミュレータについて述べ，AMDSを用いることでモバイル環境におけ

るアプリケーション構築が容易になることを示す.

第3章では地理情報システムのためのアクティブデータベースであるActiveGISについて述べる. ActiveGISはGPSを用いて取得する位置情報と無線で配信される地理データを基に位置依存サービスを提供するためのアクティブデータベースであり, 現実世界における建物との位置関係に応じてデータを受信したり, 受信した地理情報を加工して道案内を行う機能をもつ. 地理情報は現在標準化が行われているG-XML形式で配信されることを想定しており, ECAルールをXML形式で表現したECA-MLと共に配信することで, 地理情報を受信時にルールを用いたカスタマイズを行うといったサービスが提供できる.

第4章では放送環境のためのアクティブデータベースであるSADB(Super Active DataBase system)について述べる. SADBは放送データの受信およびフィルタリングを行うためのアクティブデータベースであり, 大量の受信データを高速に処理できるようにデータ受信に関する処理の高速化を行っている. また, 放送環境では大量に配信される不要なデータから必要なデータのみを抽出する情報フィルタリング技術が重要になるため, フィルタリング要求記述から自動的にフィルタリング処理のためのECAルール群を生成し, さらに環境の変化に応じてコストの低い処理を自動的に選択する情報フィルタリングアルゴリズムを提案する. さらに, 受動的にフィルタリングを行うだけでなく, 放送型/オンデマンド型通信が統合された放送型データベース環境において, ECAルールを用いて能動的に問合せを行う方式を提案する.

第5章ではリアルタイムプレゼンテーション作成のためのアクティブデータベースであるアクティブカラオケについて述べる. リアルタイムプレゼンテーションとは, 与えられたシナリオや周囲の環境の変化に応じてデータベース中のマルチメディア素材をリアルタイムで検索し, 提示するシステムである. プロトタイプシステムとして製作したアクティブカラオケは, カラオケの背景を動的に作成するシステムであり, 周囲の状況の変化をイベントとして検出し, 画像の提示やエフェクトの適用をルールにより制御する.

最後に, 第6章でこれまで述べた研究を統合することで実現する統合型ルールベース環境について述べ, 統合のための課題, 問題の解決方法について考察し, 本論文のまとめを行う.

研究業績

1．学会論文誌発表論文

1. 寺田 努，塚本昌彦，西尾章治郎: アクティブデータベースを用いた地理情報システム，情報処理学会論文誌，Vol. 41，No. 11，pp. 3103–3113 (Nov. 2000).

2. 寺田 努，塚本昌彦，西尾章治郎: 放送型データ受信のためのアクティブデータベースの設計と実装，電子情報通信学会論文誌，Vol. J83-D-I，No. 12，pp. 1272–1283 (Dec. 2000).

3. 澤井里枝，塚本昌彦，寺田 努，Loh Yin Huei，西尾章治郎: 情報フィルタリングの関数的性質について，電子情報通信学会論文誌，Vol. J85-D-I, No. 10, pp. 939–950 (Oct. 2002).

4. 寺田 努，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースの動的トリガグラフ構築機構の設計と実装，情報処理学会論文誌:データベース，Vol.43, No. SIG12(TOD16), pp. 52–63 (Dec. 2002).

5. 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: フィルタリング関数におけるセレクションとランキングについて，情報処理学会論文誌:データベース，Vol.43, No. SIG12(TOD16), pp. 80–91 (Dec. 2002).

6. 寺田 努，塚本昌彦，西尾章治郎: アクティブデータベースを用いたカラオケの背景作成システム，情報処理学会論文誌，Vol. 44, No. 2, pp. 235–244 (Feb. 2003).

7. 寺田 努，塚本昌彦，西尾章治郎: 2つのPDAを用いた携帯型エレキベースの設計と実装，情報処理学会論文誌，Vol. 44, No. 2, pp. 266–275 (Feb. 2003).

8. 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: 合成フィルタリング関数の性質について，情報処理学会論文誌：データベース，Vol. 44，No. SIG3(TOD17)，pp. 43–53 (Mar. 2003).

9. 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: 情報フィルタリングの実行順序に関する関数的性質について，情報処理学会論文誌：データベース，Vol. 44，No. SIG3(TOD17), pp. 54–64 (Mar. 2003).

10. 加下雅一，寺田 努，原 隆浩，塚本昌彦，西尾章治郎: データベース放送システムのためのサーバと移動型クライアントによる協調型問合せ処理方式，情報処理学会論文誌：データベース，Vol. 44，No. SIG8(TOD18)，pp. 92–104 (June 2003).

2．研究会等発表論文（査読付）

1. S. Sanguantrakul, T. Terada, M. Tsukamoto, S. Nishio, K. Miura, S. Matsuura, and T. Imanaka: User Customized Classification and Selection for Broadcast Data, *in Proc. of Semantic Issues in Multimedia Systems, IFIP TC-2 Working Conference*, pp. 596–601 (Sep. 1999).

2. 寺田 努, 村瀬 亨, 塚本昌彦, 西尾章治郎: Active GIS: アクティブモーバイルデータベースを用いた地理情報システム，電気情報通信学会第11回データ工学ワークショップ(DEWS2000)論文集(CD-ROM) (Mar. 2000).

3. 澤井里枝, 寺田 努, 塚本昌彦, 西尾章治郎: フィルタリングSQL: フィルタリングのためのユーザ要求記述言語，電気情報通信学会第11回データ工学ワークショップ(DEWS2000)論文集(CD-ROM) (Mar. 2000).

4. 澤井里枝，Loh Yin Huei，寺田 努，塚本昌彦，西尾章治郎: フィルタリングの関数的性質とその関係について，電気情報通信学会第12回データ工学ワークショップ(DEWS2001)論文集(CD-ROM) (Mar. 2001).

5. T. Terada, M. Tsukamoto, and S. Nishio: Active GIS: A Geographic Information System Using Active Database Systems, *in Proc. of Symposium on Asia GIS 2001*(CD-ROM) (June 2001).

6. R. Sawai, M. Tsukamoto, Y. H. Loh, T. Terada, and S. Nishio: Functional Properties of Information Filtering, *in Proc. of the 27th International Conference on Very Large Data Bases (VLDB2001)*, pp. 511–520 (Sep. 2001).

7. 寺田 努，塚本昌彦，西尾章治郎: 2つのPDAを用いた携帯型エレキベースのためのインタフェース，日本ソフトウェア科学会第9回インタラクティブシステムとソフトウェアに関するワークショップ(WISS2001)，pp. 185–190 (Dec. 2001).

8. 澤井里枝, 塚本昌彦, 寺田 努, Loh Yin Huei, 西尾章治郎: フィルタリング関数におけるセレクションとランキングの性質について, 情報処理学会データベースとWeb情報システムに関するシンポジウム (DBWeb2001), pp. 277–284 (Dec. 2001).

9. T. Terada, M. Tsukamoto, and S. Nishio: A Portable Electric Bass Using Two PDAs, *in Proc. of First International Workshop on Entertainment Computing (IWEC2002)*, pp. 286–293 (May 2002).

10. T. Terada, M. Tsukamoto, and S. Nishio: An Active Database System for Receiving Broadcast Data, *in Proc. of IASTED International Conference on Information Systems and Databases (ISDB 2002)*, pp. 122–128 (Sep. 2002).

11. M. Kashita, T. Terada, T. Hara, M. Tsukamoto, and S. Nishio: A Collaborative Query Processing Method for a Database Broadcasting System, *in Proc. of IASTED International Conference on Communications, Internet, and Information Technology (CIIT 2002)*, pp. 60–66 (Nov. 2002).

12. 早川敬介, 塚本昌彦, 寺田 努, 義久智樹, 岸野泰恵, 柏谷 篤, 坂根 裕, 西尾章治郎: ユビキタスコンピューティングのための入出力制御デバイス, 日本ソフトウェア科学会第10回インタラクティブシステムとソフトウェアに関するワークショップ (WISS2002), pp. 127–132 (Dec. 2002).

13. 澤井里枝, 塚本昌彦, 寺田 努, 西尾章治郎: フィルタリング関数の和積とその性質, 電気情報通信学会第14回データ工学ワークショップ (DEWS2003) 論文集 (Mar. 2003).

14. T. Terada, M. Tsukamoto, and S. Nishio: A System for Presenting Background Scenes of Karaoke Using an Active Database System, *in Proc. of ISCA 18th International Conference on Computers and Their Applications (CATA 2003)*, pp. 160–165 (Mar. 2003).

15. M. Kashita, T. Terada, T. Hara, M. Tsukamoto, and S. Nishio: An Adaptive Query Processing Method according to System Environments in Database Broadcasting Systems, *in Proc. of ISCA 18th International Conference on Computers and Their Applications (CATA 2003)*, pp. 174–179 (Mar. 2003).

16. R. Sawai, M. Tsukamoto, T. Terada, and S. Nishio: Composition of Filtering Functions, *in Proc. of the 9th International Conference on Databese Systems for Advanced Applications (DASFAA 2003)*, pp. 293–300 (Mar. 2003).

17. 寺田 努, 塚本昌彦, 西尾章治郎: ウェアラブル生活を豊かにするルールベース環境音楽システム, 情報処理学会シンポジウムシリーズ マルチメディア, 分散, 協調とモバイルシンポジウム (DICOMO2003) 論文集, pp. 389–392 (June 2003).

18. T. Terada, M. Tsukamoto, and S. Nishio: A Dynamic Construction Mechanism of a Trigger Graph on Active Databases in Mobile Computing Environments, *in Proc. of 6th International Workshop on Mobility in Databases and Distributed Systems (MDDS'03)* (Sep. 2003, to appear).

3. その他の研究会等発表論文

1. 寺田 努, 塚本昌彦, 西尾章治郎: モーバイルコンピューティング環境におけるECAルール配送機構を用いたデータベースビューの実現, 情報処理学会 第55回全国大会講演論文集 (3), pp. 351–352 (Sept. 1997).

2. 寺田 努, ソムヌック サグアントラクーン, 塚本昌彦, 西尾章治郎, 三浦康史, 松浦 聰, 今中 武: アクティブデータベースを用いた放送型データ格納方式, 情報処理学会研究報告 (分散処理とマルチメディア研究会 97-DPS-85), Vol. 97, No. 104, pp. 243–248 (Nov. 1997).

3. ソムヌック サグアントラクーン, 寺田 努, 塚本昌彦, 西尾章治郎, 三浦康史, 松浦 聰, 今中 武: 放送型データのユーザ適応型分類・選択手法, 情報処理学会研究報告 (分散処理とマルチメディア研究会 97-DPS-85), Vol. 97, No. 104, pp. 249–254 (Nov. 1997).

4. 寺田 努, ソムヌック サグアントラクーン, 塚本昌彦, 西尾章治郎, 三浦康史, 松浦 聰, 今中 武: 放送型データ受信のためのアクティブデータベースについて, 情報処理学会研究報告 (データベースシステム研究会98-DBS-116(1)), Vol. 98, No. 58, pp. 119–126 (July 1998).

5. S. Sanguantrakul, T. Terada, M. Tsukamoto, S. Nishio, K. Miura, S. Matsuura, and T. Imanaka: User Customized Classification and Selection for Broadcast Data（木構造を用いる放送型データのフィルタリング・分類手法），情報処理学会研究報告 (データベースシステム研究会 99-DBS-117)，Vol. 99，No. 6，pp. 25–30 (Jan. 1999).

6. 寺田 努，莫 君，村瀬 亨，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースのECAルール実行監視機構の設計と実装，情報処理学会研究報告 (データベースシステム研究会 99-DBS-119)，Vol. 99，No. 7，pp. 369–374 (July 1999).

7. 寺田 努，塚本昌彦，西尾章治郎: Active Karaoke: アクティブデータベースを用いたカラオケの背景作成システム，情報処理学会研究報告 (音楽情報科学研究会 2000-MUS-34)，Vol. 2000，No. 19，pp. 73–78 (Feb. 2000).

8. 寺田 努，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースの動的トリガグラフ構築手法，情報処理学会研究報告 (データベースシステム研究会 2000-DBS-122)，Vol. 2000，No. 69，pp. 191–198 (July 2000).

9. 寺田 努，塚本昌彦，西尾章治郎: G-XMLをサポートするアクティブデータベースシステム，情報処理学会研究報告 (モバイルコンピューティングとワイヤレス通信研究会 2000-MBL-14)，Vol. 2000，No. 87，pp. 123–130 (Sep. 2000).

10. 寺田 努，塚本昌彦，西尾章治郎: カラオケの背景を動的に作成するアクティブデータベースシステム，第61回情報処理学会全国大会 (Nov. 2000).

11. 寺田 努，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースの動的トリガグラフ構築機構の実現，情報処理学会研究報告 (モバイルコンピューティングとワイヤレス通信研究会 2001-MBL-17)，Vol. 2001，No. 46，pp. 39–46 (May 2001).

12. 加下雅一，寺田 努，塚本昌彦，原 隆浩，西尾章治郎: データベース放送システムにおける移動型クライアントのための問合せ処理方式，情報処理学会研究報告 (モバイルコンピューティングとワイヤレス通信研究会 2001-MBL-17)，Vol. 2001，No. 46，pp. 47–54 (May 2001).

13. 寺田 努，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースのシミュレーション環境について，情報処理学会研究報告 (データベースシステム研究会 2001-DBS-125(1))，Vol. 2001，No. 70，pp. 351–358 (July 2001).

14. 澤井里枝，塚本昌彦，寺田 努，Loh Yin Huei，西尾章治郎: フィルタリング関数の合成とその性質について，情報処理学会研究報告 (データベースシステム研究会 2001-DBS-125(2))，Vol. 2001，No. 71，pp. 61–68 (July 2001).

15. 寺田 努，塚本昌彦，西尾章治郎: モバイル環境におけるアクティブデータベースを用いた地理情報システムについて，地理情報システム学会 空間IT分科会 第1回空間ITワークショップ，Vol. 2001，No. 1，pp. 26–33 (July 2001).

16. 寺田 努，塚本昌彦，西尾章治郎: DoublePad/Bass:2つのPDAを用いた携帯楽器，情報処理学会研究報告 (音楽情報科学研究会 2001-MUS-41)，Vol. 2001，No. 82，pp. 77–82 (Aug. 2001).

17. 宮前雅一，中村聡史，寺田 努，塚本昌彦，西尾章治郎: ウェアラブルコンピューティングのための拡張可能なルール処理システム，情報処理学会研究報告 (情報家電コンピューティング研究グループ 2002-IAC-3), Vol. 2002, pp. 41–46 (June 2002).

18. 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: フィルタリング関数の合成順序について，情報処理学会研究報告 (データベースシステム研究会 2002-DBS-128)，Vol. 2002，No. 67，pp. 335–342 (July 2002).

19. 寺田 努，塚本昌彦，坂根 裕，義久智樹，岸野泰恵，早川敬介，柏谷 篤，西尾章治郎: ユビキタスコンピューティングのための入出力制御デバイスの動作記述方式，ヒューマンインタフェースシンポジウム2002論文集，pp. 331–334 (Sep. 2002).

20. 塚本昌彦，寺田 努，早川敬介，柏谷 篤: ルールに基づく入出力制御によるユビキタスコンピューティング，ヒューマンインタフェースシンポジウム2002論文集，pp. 327–330 (Sep. 2002).

21. 早川敬介，柏谷 篤，塚本昌彦，寺田 努，義久智樹，岸野泰恵，坂根 裕，西尾章治郎: ユビキタスコンピューティングのための入出力制御デバイスの設計と実装，ヒューマンインタフェースシンポジウム2002論文集，pp. 335–338 (Sep. 2002).

22. 柳瀬康宏，森田和延，青木功介，釣 裕美，高木越子，山本光穂，堀 雅和，黒田 卓，山西潤一，寺田 努，塚本昌彦: ウェアラブルコンピュータを使用した野外学習支援システムの開発，平成14年度電気関係学会北陸支部連合大会予稿集，pp. 229 (Sep. 2002).

23. 寺田 努，塚本昌彦，坂根 裕，義久智樹，岸野泰恵，早川敬介，柏谷 篤，西尾章治郎: ユビキタスコンピューティングのための入出力制御デバイスの動作記述言語，第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 197–198 (Sep. 2002).

24. 塚本昌彦，寺田 努，早川敬介，柏谷 篤: ユビキタスコンピューティングを実現するためのルールに基づく入出力制御デバイス，第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 195–196 (Sep. 2002).

25. 早川敬介，柏谷 篤，塚本昌彦，寺田 努，義久智樹，岸野泰恵，西尾章治郎: ユビキタスコンピューティングのための入出力制御デバイスのハードウェアアーキテクチャ，第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 199–200 (Sep. 2002).

26. 義久智樹，塚本昌彦，坂根 裕，寺田 努，岸野泰恵，早川敬介，柏谷 篤，西尾章治郎: ユビキタスコンピューティングのための入出力制御デバイスのソフトウェアアーキテクチャ，第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 201–202 (Sep. 2002).

27. 岸野泰恵，義久智樹，寺田 努，塚本昌彦，坂根 裕，早川敬介，柏谷 篤，西尾章治郎: ユビキタスコンピューティングのための入出力制御デバイスのPC統合環境，第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 203–204 (Sep. 2002).

28. 塚本昌彦，寺田 努，堀 雅和: ウェアラブルコンピューティングのためのシステム基盤，第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 213–214 (Sep. 2002).

29. 宮前雅一，中村聡史，寺田 努，塚本昌彦，橋本隆之，青木功介，堀 雅和，西尾章治郎: ウェアラブルコンピューティングのためのルール処理システムの設計と実装，第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 215–216 (Sep. 2002).

30. 中村聡史，宮前雅一，寺田 努，塚本昌彦，柳瀬康宏，釣 裕美，堀 雅和，西尾章治郎: ウェアラブルコンピューティングのためのルール処理システムを用いたサービス，第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 217–218 (Sep. 2002).

31. 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: フィルタリング関数の合成と実システムへの適用について，情報処理学会研究報告 (データベースシステム研究会 2002-DBS-129)，Vol. 2003, No. 5, pp. 75–82 (Jan. 2003).

32. 加下雅一，寺田 努，原 隆浩，塚本昌彦，西尾章治郎: 放送型データベースシステムにおける適応的問合せ処理方式，情報処理学会研究報告 (データベースシステム研究会 2002-DBS-129)，Vol. 2003, No. 5, pp. 9–16 (Jan. 2003).

33. 中尾太郎，寺田 努，塚本昌彦，宮前雅一，庄司 武，岸野泰恵，義久智樹，西尾章治郎: ウェアラブル型ルールベースシステムを用いた農作業支援システム，第65回情報処理学会全国大会講演論文集 (5)，pp. 211–214 (Mar. 2003).

34. 三浦直樹，宮前雅一，寺田 努，塚本昌彦，西尾章治郎: Aware-Mail: ウェアラブルコンピューティング環境のためのイベント駆動型メールシステム，第65回情報処理学会全国大会講演論文集 (5)，pp. 207–210 (Mar. 2003).

35. 小寺拓也，澤井里枝，寺田 努，塚本昌彦，西尾章治郎: 数学的性質を利用した処理方法最適化機構をもつ情報フィルタリングシステム，情報処理学会研究報告 (July 2003，発表予定).

36. 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: 情報フィルタリングのためのユーザ要求記述言語 FilteringSQL について，情報処理学会研究報告 (July 2003，発表予定).

目 次

第1章

序章

1.1 研究の背景

近年，マイクロエレクトロニクス技術や無線通信技術の発達により，日常生活のあらゆる部分にコンピュータが入り込み，あらゆるものが電子化されるようになった．PDAやノートパソコンを持ち歩くモバイルコンピューティング，家電にインターネット接続などさまざまな機能をもたせた情報家電といった言葉は一般的になり，近年ではコンピュータを常に身につけて生活をおくるウェアラブルコンピューティングや，環境内のあらゆるものにコンピュータを埋め込んでどこでもコンピュータを使えるようにするユビキタスコンピューティングといった概念も登場している[35][71][72]．近い将来には，図1.1に示すように，多くの人々が自分専用の端末を常に持ち歩き，あらゆる場所に設置されたコンピュータや他のユーザとデータをやり取りしながらさまざまなサービスを受ける時代がやってくると期待されている[67]．

あらゆる物が電子化されるようになると，その機器に要求される機能もますます多様になり，これまでのように単一の機能だけを提供するコンピュータではユーザの要求に対応できなくなる．例えば，博物館において入館者に携帯端末を配り，今いる場所に応じて展示物情報を端末に表示するようなサービスを考える．すべての人が自分専用のコンピュータを持ち歩くようになると，施設ごとにわざわざ端末を借りるのではなく，自分がいつも持ち歩いている端末上で博物館のサービスが提供される方が便利である．このように自分用の端末を常に持ち歩くようになると，目的に応じて端末を取り替えるのではなく，自分の端末の機能が状況に応じて変わるという枠組みのほうが有効であるといえる．

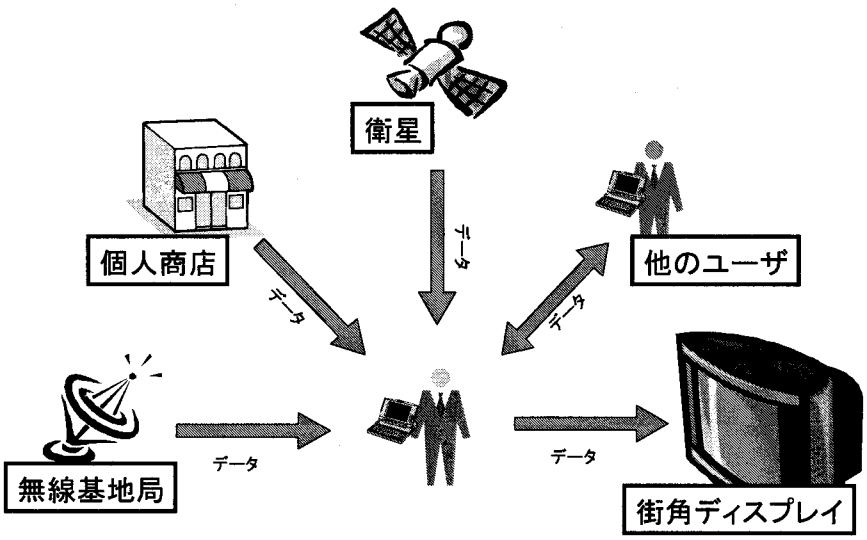


図 1.1: モバイル環境におけるサービス

あらゆるものが自律的に動作し，状況に応じて動作を変更するためには新しいコンピュータの枠組みが必要となる．そこで本研究では，このようなサービスを提供するための新たな枠組みとして，イベント駆動型データベース (アクティブデータベース) の概念を用いたサービス実現手法を提案する．

1.2 アクティブデータベースの利用

一般に人間は現実世界の事象を因果的に捉えるという見方がある [27]．したがって，サービス記述においても事象の因果関係を直接的に記述する枠組みが有効であるといえる．ここで，本研究では因果関係を記述するために次の3つの要素からなる**ＥＣＡルール**を利用する．

イベント（E）：発生する事象

コンディション（C）：ルールを実行するための条件

アクション（A）：行う動作

ECA ルールでは，何か事象 (E: Event) が発生したときに，特定の条件 (C: Condition) を満たせば，ある動作 (A: Action) を行う．ECA ルールを用いて動作を記述する方式は，古

```
define rule [ルール名] is
    on [イベント] to [対象データベース]
    where [コンディション]
    do (instead) [アクション]
```

図 1.2: POSTGRESのECAルール記述構文

くからデータベースの分野で，自律的に動作するデータベースシステム(アクティブデータベース)の動作を記述するのに用いられており，ECAルールを用いることで，ユーザの要求を因果的に，直感的に記述できるようになる[23][74]．また，ECAルールはアクションの実行によって新たなイベントを発生させられるため，1つのイベントの発生によって複数のECAルールを連鎖的に実行できる．この枠組みはエキスパートシステムなどで人間の知識を効率よく表現するためのif-then形式とほぼ同等のものであるが，ECAルールではif節で記述する内容を事象と条件に分割することでより直感的なルール記述を可能にしている．

一般にアクティブデータベースで記述できるイベントやアクションはデータベース操作(検索，挿入，削除，更新)であり，コンディションにはデータベースのレコード同士やレコードと定数による比較が記述できる．ECAルールの記述構文は明確には規定されていないが，例えばPOSTGRES[55]のECAルール記述構文は図1.2のようになっている．[イベント]にはルールを発火させるイベント名を記述する．このとき，対象となるデータベース名を指定することでそのデータベース内に起こった事象のみを取り扱える．[コンディション][アクション]にはそれぞれコンディション内容とアクション内容を記述する．アクション記述の際には*instead*スイッチを付加することで，イベント発生の元となった操作は行わずに，アクションだけを実行できる．POSTGRESにおけるルール例を図1.3に示す．このルールは社員管理データベースにおいて，*employee*テーブルに新しいレコード(新入社員)が挿入されたとき，自動的にその*salary*を職員全員の平均値から10を引いた値にするルールである．このように，アクティブデータベースにおけるECAルールは，発注処理の自動化や，データベース内容の改変が制約を満たしているかのチェックに用いられることが多い．

アクティブデータベースは，リレーショナルデータベース(Relational Database: RDB)やオブジェクト指向データベース(Object-Oriented Database: OODB)のように，それ自身

```
define rule SetSalary
on insert to employee
do begin
 newSalary := (select avg(salary) from employee) - 10
 update employee (salary=newSalary) where employee.id = new.id
 end
```

図 1.3: ECA ルール例

が新たなデータモデルを提案するデータベースではないため，既存のデータベースを拡張することで実現するのが一般的である．RDB を拡張して実現したアクティブデータベースには，POSTGRES[55]，Starburst[73]，Ariel[14]，HiPAC[8] などが知られている．OODB を拡張して実現したアクティブデータベースには，ODE[11]，Jasmine/A[22] などが知られている．本研究で構築するアクティブデータベースシステムは，RDB を拡張して実現したアクティブデータベースである．

モバイル環境や放送環境において，ECA ルールを用いてさまざまなサービスを実現するためには，データパケットの受信やユーザの移動などデータベースに関連するもの以外のイベントも取り扱えるようにルールの仕様を拡張する必要がある．アクションに関してもデータベース操作以外に，情報の表示やタイマの設定，音楽再生など多様な機能が必要となる．そこで，本研究ではさまざまな環境を想定し，それらの環境にアクティブデータベースの枠組みを適用することで，多様なサービスを提供できるシステムの構築を目的とする．具体的には，想定環境において必要とされる機能を抽出し，ECA ルール言語仕様の拡張およびシステムの設計・実装を行う．また，ECA ルールの無限連鎖などアクティブデータベースの枠組みを用いることで起こる問題点を明らかにし，その解決方法を提案する．

1.3 本論文の構成

本論文は全6章で構成され，その内容は以下のとおりである．第2章ではモバイルコンピューティング環境におけるサービスを提供するためのアクティブデータベースシステムである AMDS(Active Mobile Database System) について述べる．AMDS は，移動体の接続・

切断といった事象をイベントとして取り扱えるようにしたアクティブデータベースシステムであり，モバイル環境においてルールベースサービスを行う際の基盤となるシステムである．この章ではAMDSの設計および実装に加え，分散配置されたECAルール群が無限ループ等の異常動作を起こすかどうかを検出する手法を提案する．また，AMDSのためのアプリケーション構築環境およびシミュレータについて述べ，AMDSを用いることでモバイル環境におけるアプリケーション構築が容易になることを示す．

次に，第3章では地理情報システムのためのアクティブデータベースシステムであるActiveGISについて述べる．ActiveGISはGPSから取得する位置情報と無線で配信される地理データをもとに位置依存サービスを提供するためのアクティブデータベースであり，現実世界における建物との位置関係に応じてデータの送受信を行ったり，受信した地理情報を加工してカスタマイズされた地図を作成する機能をもつ．地理情報は現在標準化が行われているG-XML形式で配信されることを想定しており，ECAルールをXML形式で表現したECA-MLを共に配信することで，状況の変化に応じて受信した地理情報をカスタマイズするといった機能を提供している．

第4章では放送環境のためのアクティブデータベースシステムであるSADB(Super Active DataBase system)について述べる．SADBは放送データの受信およびフィルタリングを行うためのアクティブデータベースであり，大量の受信データを高速に処理できるようにデータ受信に関するイベント処理の高速化を行っている．また，放送環境では大量に配信される不要なデータから必要なデータのみを抽出する情報フィルタリング技術が重要になるため，フィルタリングのためのユーザ要求記述言語から自動的にフィルタリング処理のためのECAルール群を生成し，環境の変化に応じてコストの低い処理を自動的に選択する情報フィルタリングアルゴリズムを提案する．さらに，受動的にフィルタリングを行うだけでなく，放送型/オンデマンド型通信が統合された放送型データベース環境において，ECAルールを用いて能動的に問合せを行う方式を提案する．

第5章では，リアルタイムプレゼンテーション生成のためのアクティブデータベースについて述べる．リアルタイムプレゼンテーションとは，与えられたシナリオや周囲の環境の変化に応じてデータベース中のマルチメディア素材をリアルタイムで検索し，提示するシステムである．プロトタイプシステムとして構築したアクティブカラオケは，カラオケの背景を動的に作成するシステムであり，周囲の状況の変化をイベントとして検出し，画像の提示やエフェクトの適用をルールにより制御する．

最後に，第6章ではこれまで述べた研究を統合することで実現する統合型ルールベース環境について述べ，統合のための課題，問題の解決方法について考察し，本論文のまとめを行う．

なお，第2章は文献[59][64][65]で公表した結果に基づき論述し，第3章は文献[60][62]で公表した結果に基づき論述する．第4章は文献[29][30][47][48][49][50][51][52][63]で公表した結果に基づき論述し，第5章は文献[61][66]で公表した結果に基づき論述する．

第2章

モバイル環境におけるデータ統合のためのアクティブデータベース

2.1 まえがき

近年，無線通信技術や計算機ハードウェア技術の急速な発展により，無線通信機能をもつ携帯端末を用いることで場所を固定せずにネットワーク上のさまざまな資源を利用することが可能になった．この新しい計算環境を移動体計算環境と呼ぶ．移動体計算環境のモデルは図2.1に示すように，固定ネットワークに無線通信可能な移動端末を含んだ形態である．移動体計算環境においては，以下のような新しいサービスを提供できるようになる．

- 会社において，各社員は一台ずつ各自のスケジュールが入力された携帯端末を持ち歩き，本社でスケジュールデータを統合して全社員のスケジュールを管理して効率的に仕事を割り当てる．
- 遊園地などのアミューズメント施設で入場者に一台ずつ携帯端末を持たせ，各アトラクションの待ち時間等の情報を提供する．
- 美術館や博物館において，入館者は一台ずつ携帯端末を持ち，展示物に近づくと自動的にその展示物の詳細情報が表示される．

このようなサービスを実現するためには，移動体から，または移動体上でデータを収集する必要がある．しかし，従来の分散データ管理技術ではホストの移動を考慮していないため，移動体のネットワークへの接続・切断やデータの収集を自動的に行う共通の基盤が

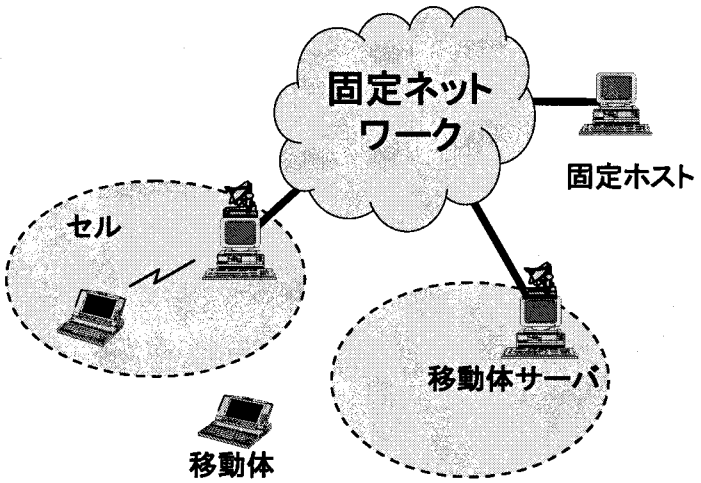


図 2.1: 移動体計算環境

望まれている．このような要求に対し，筆者らはアクティブデータベースを拡張することで，移動体計算環境における各種のイベントを容易に扱うことができるアクティブモーバイルデータベース (Active Mobile Database System: AMDS) の研究を行ってきた [40][41]. AMDS では，従来のアクティブデータベースのイベント（更新・挿入・削除など）に加え，移動体の接続・切断などを扱うイベントも提供しており，AMDS を用いることで移動体計算環境におけるアプリケーション開発が容易になる．

AMDS の動作記述言語である ECA ルールはアクションの実行によって新たなイベントを引き起こすことができる．ECA ルールを連鎖的に実行させることで複雑な動作が実現できる一方，無限ループなど，予期せぬ動作に陥る可能性がある．そのため，ECA ルールの実行を監視してシステムの異常動作を回避する必要がある．そこで本章では，移動体計算環境において ECA ルールの実行を監視して ECA ルールの無限連鎖による異常動作を検出するためのいくつかの手法を提案する．また，異常動作検出機構を利用しながら安全なアプリケーション開発及びシミュレーションを可能にする AMDS シミュレータを提案する．以下，2.2 節では AMDS の概要について述べ，2.3 節で異常動作検出手法について述べる．次に 2.4 節で提案手法の評価およびアプリケーション開発を行う環境である AMDS シミュレータについて説明し，最後に 2.5 節で本章のまとめを行う．

2.2 AMDS

AMDSはアクティブデータベースを拡張して移動体計算環境に適合させたものである．AMDSでは，従来のアクティブデータベースで提供されている，データベースに関するイベント（挿入，削除，更新，検索）に加えて，移動体の動作に関するイベント（移動体のセルへの接続・切断），AMDS間の通信に関するイベント（データの受信）を用意している．アクションとしてはAMDS間のデータ送信関数，データベース対する問合せ関数などが提供されている．以下，AMDSの動作記述言語であるECAルールと，AMDSにおけるECAルールの処理モデルについて述べる．

2.2.1 AMDSのECAルール

AMDSは一般のアクティブデータベースと同様ECAルールで動作する．AMDSにおけるECAルールの記述構文を図2.2に示す.「イベント名」には，ルールが対象とするイベントの名前を記述し,「対象テーブル」には，イベントが対象とするテーブルを指定する．イベントは1つのルールに対して1つのみ記述でき，複数イベントの同時発生といった記述はできない.「変数型宣言」では，ECAルール中で使用するローカル変数を定義する．宣言する変数は，文字列変数，タップル変数またはタップルの組を表す変数のいずれかである.「コンディション」にはルールの発火条件を記述する．記述は，‘< 左辺 > < オペレータ > < 右辺 >’の形の羅列で行う．両辺には，データベース属性や変数を指定する.「アクション」にはルールが発火したときに行う動作を1つ以上記述する．

AMDSで提供されているイベント，アクションを表2.1，2.2に示す．イベントのうち，CONNECTおよびDISCONNECTに関しては，移動体サーバにおいてのみ発火するイベントである．また，到着したデータの内容を用いて処理を行うルールなどでは，イベント対象となったタップル情報が必要になる場合があるため，NEWデータ，OLDデータと呼ぶシステム変数を用意する．イベント発生時にこれらの変数に必要な情報が自動的に格納され，ルール中で自由に使用できる．各イベントに対するNEWデータ，OLDデータの内容を表2.3に示す．

ECAルールの記述例として，近づいてきた移動体のスケジュールを収集するルール例を図2.3に示す．接続ルールは，移動体が接続してきたときにその移動体に対してデータ要求を行う移動体サーバ用のルールである．また，返信ルールは移動体サーバからの要求パケッ

```
CREATE RULE ルール名 ON イベント名
    [ TO 対象テーブル ]
    [ 変数型宣言 ]
    [ WHERE コンディション ]
    THEN DO アクション
```

図 2.2: ECAルールの記述構文

表 2.1: AMDSのイベント

名称	内容
CONNECT	移動体のセルへの接続
DISCONNECT	移動体のセルからの切断
SELECT	データ参照
INSERT	タップルの挿入
DELETE	タップル削除
UPDATE	タップル更新
RECEIVE	データパケットの受信
TIMER	設定したタイマの発火

表 2.2: AMDSのアクション

名称	内容
QUERY([クエリー内容])	データベース操作
SEND([宛先], [送信内容])	データの送信
INSERT_ECA([ルール内容])	ECAルール格納
DELETE_ECA([ルール識別子])	ECAルール削除
ENABLE_ECA([抽出条件])	ECAルール有効化
DISABLE_ECA([抽出条件])	ECAルール無効化
SET_TIMER([タイマ条件])	新たなタイマの設定
KILL_TIMER([タイマ識別子])	タイマの削除

表 2.3: NEWデータとOLDデータの内容

イベント	NEW	OLD
CONNECT	接続移動体情報	–
DISCONNECT	–	切断移動体情報
SELECT	参照タップル	–
INSERT	挿入タップル	–
DELETE	–	削除タップル
UPDATE	更新後タップル	更新前タップル
RECEIVE	到着パケット内容	–
TIMER	タイマ識別子	–

```
CREATE RULE 接続 on CONNECT
  THEN DO
   SEND( new.from, "Request_" );

CREATE RULE 返信 on RECEIVE
  WHERE new.header = 'Request_'
  THEN DO data = QUERY("select s.*
                        from Schedule s");
   SEND( new.from, "result_", data );
```

図 2.3: ECAルール例

トを受信したとき，スケジュールデータを送り返す移動体用ルールである．

移動体サーバでは，移動体が接続してきたときに接続ルールが起動し，接続ルールのアクションにより，移動体の返信ルールが起動する．このように，ECAルールでは1つのイ

ベントの発生によって複数のECAルールを連鎖的に実行させて複雑な動作を実現でき，そのようなECAルール群を用いることで，街角情報配信サービスや博物館における展示物情報表示サービス，すれ違う人と趣味が一緒ならば通知するコミュニケーション支援サービスなどさまざまなサービスが提供できる．

2.2.2 実行モデル

前節で示した記述構文に沿って記述されたECAルールは，AMDS上で次のように処理される．

1. イベントキューからイベントをひとつ取り出す．
2. イベントに適合するルールを検索．
3. 適合するルールがあればコンディションを評価．
4. コンディションが真ならアクション実行．

システムはイベントの発生を検出すると，まず発生イベントに関するNEWデータ，OLDデータを作成し，それらをまとめてイベントキューと呼ぶイベント格納用キューに格納する．システムは常にイベントキューを監視しており，キューにイベントがあれば取り出して発生したイベントに適合するルールを検索する．検索されたルール群はそれぞれコンディション評価され，コンディションが成り立つならアクションが実行される．ルールは実行プライオリティをもたないため，ルール群に属するルールが評価される順番は規定されない．また，アクション実行によって発生する新たなイベントはイベントキューに格納されるため，ルール群の処理がすべて終わるまでは違うイベントに関する処理が行われることはない．コンディション記述においてタップルの参照などイベントとなり得る操作が行われる場合も，その操作はイベントとして認識されない．ルールの実行は集合指向で行われる．たとえば，1つのSQL文を用いてデータベースに3つのタップルが挿入された場合，3つのINSERTイベントが起こるのではなく，1つのINSERTイベントしか起こらない．

2.2.3 システムの構成

AMDSのシステム構成を図2.4に示す．イベント検出部では，保持しているデータベースや外部からのパケット受信など，システム内外で起こったイベントを検出する．その際，

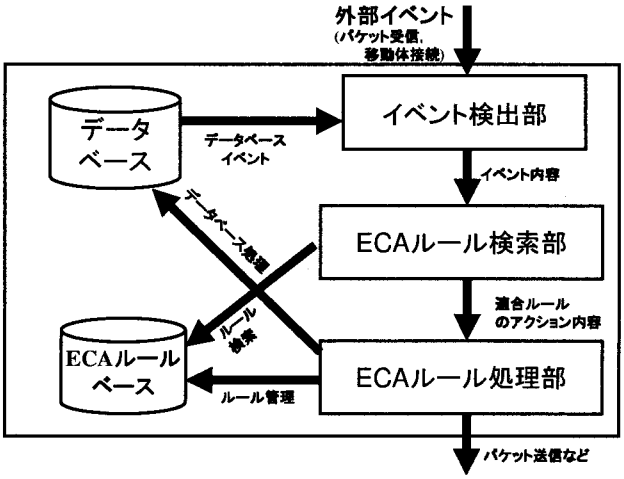


図 2.4: AMDSのシステム構成

NEW/OLDデータの作成処理を行う．ECAルール検索部では，イベント検出部からイベントを受け取り，ECAルールベースから対応するECAルールを検索する．存在すれば，それらのルールすべてを取り出し，コンディションの評価を行い，実行されるべきルールをECAルール処理部に渡す．ECAルール処理部では，ECAルール検索部から受け取ったルールのアクションを実行する．

2.3 異常動作の検出

ECAルールは，アクションの実行によって新たなイベントを発生させられるため，複数のECAルールを連鎖的に実行させて複雑なサービスを提供できる．例えば図2.5に示した例は，本屋に近づいたときに自動的にユーザが欲しい本の存在と値段を提示するサービスを表すECAルール群である．図におけるサーバルール1，2は本屋の移動体サーバに格納されているルール，クライアントルール1，2はユーザの端末に格納されているルールである．ユーザが本屋に近づくと，まず接続を検出したサーバルール1が起動され，ユーザに対して欲しい本の情報を問い合わせるメッセージを送る．メッセージを受信したクライアント側では自動的にクライアントルール1が起動し，ローカル端末中の個人データベースから欲しい本の情報を問い合わせてサーバに送信する．サーバがその情報を受けると今度はサーバルール2が起動し，本屋のデータベースからその本があるかどうか，値段はいく

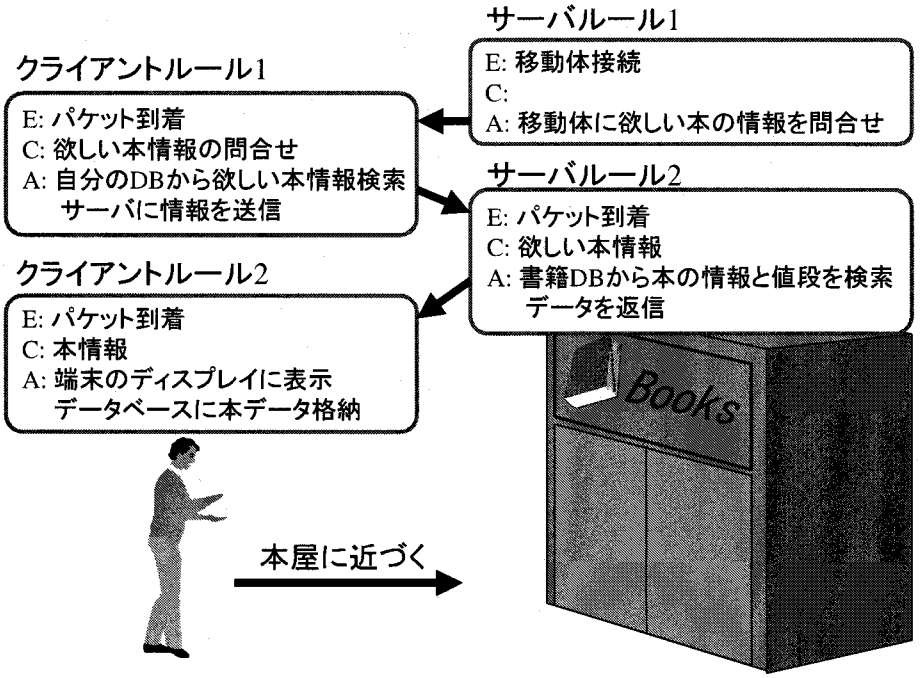


図 2.5: ECA ルールを連鎖させたサービスの例

らかといった情報を検索してユーザに送り返し，そのメッセージにより起動されたクライアントルール2がユーザに結果を提示する．

このように，ECA ルールは連鎖的に実行させることで複雑な動作を実現できる．一方，無限ループなど予期しない連鎖を起こす可能性があり，このような異常動作を検出することが必要となる．例えば，先ほどの本屋のサービスに加えて，ユーザがデータベースに本情報を格納した際にも本屋サーバに本情報を問い合わせるサービスを追加するため，図2.6に示すようにクライアントルール3を追加した場合，クライアントルール2，クライアントルール3，サーバルール2の間で無限ループが発生することになる．移動体がもつルールと移動体サーバがもつルールを個別に見てもループを起こすかは判断し難いが，移動体の移動や保持しているルールの変化などにより，異常動作が発生する．

このような異常動作を解析的に検出するのは困難であるが，ECA ルールを用いたアプリケーションを構築する場合，ECA ルールが異常動作を起こさないことを保証する必要がある．したがって，異常動作が起こるかどうかをあらかじめ調べたり，異常が実際に起こったときにそれを検出するような仕組みが必要となる．

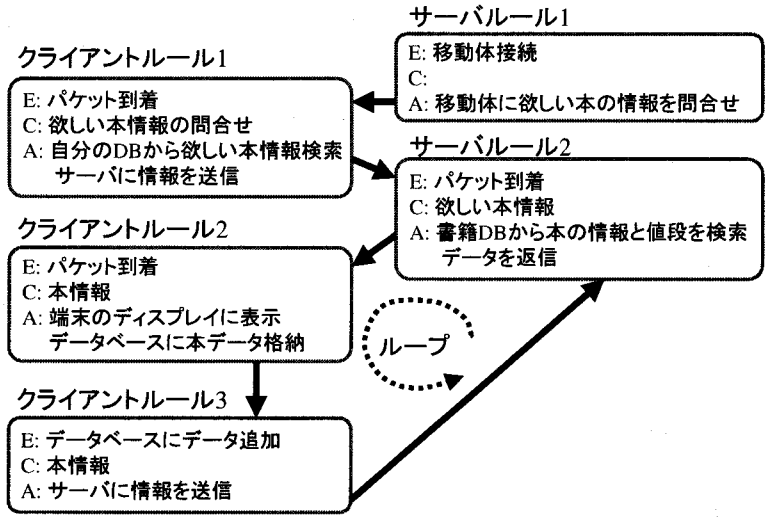


図 2.6: 無限ループの発生例

2.3.1 異常動作の検出手法

アクティブデータベースにおける異常動作検出には，次に示す2つの手法が考えられる．

- 事前検出: データベース（ECAルール）が動作していない状態で，トリガグラフと呼ばれる有向グラフを用いて論理的に異常動作の有無を調べる．
- 実行時検出: 実際にデータベースが動作している状態で，リアルタイムに異常動作の有無を検出する．

事前検出で用いるトリガグラフとは図2.7に示すように，あるECAルールから次に引き起こすルールに対して有向のパスをはるという作業をすべてのルールに対して行ったもので，出来上がったグラフ中にループが存在していなければ，そのルール群は安全であるとするものである [2][7]．トリガグラフは，その信頼性を高めるためのさまざまな拡張がされており [32][33]，アクティブデータベースの無限ループ検出においては信頼性が高い手法として広く用いられている．

しかし，移動体計算環境においては移動体の移動によりネットワーク構成が動的に変化する．また，AMDSではECAルールのホスト間でのやりとりも頻繁に起こるため，複数ホスト間にまたがるECAルールの相互関係を考慮する必要がある．筆者らの研究グループでは，AMDSにおいて静的にトリガグラフを構築できるマージトリガグラフを用いた異常

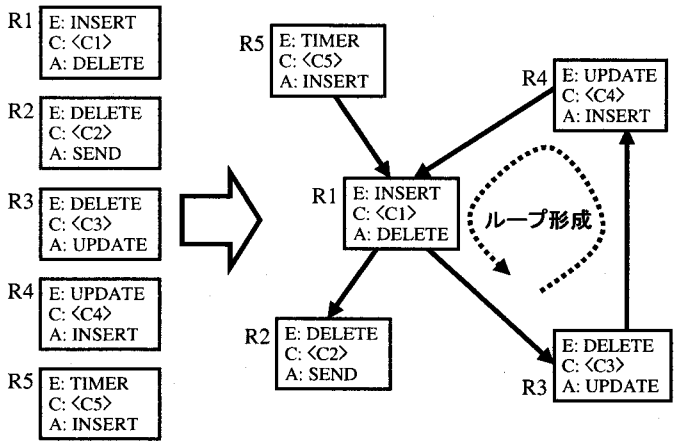


図 2.7: トリガグラフの構築例

動作検出に関する研究を行ってきた[41]. しかし，マージトリガグラフの構築には非常に高いコストが必要であり，また，マージトリガグラフを作成するためには，あらかじめすべての移動体や固定ホスト内にあるECAルールを知っておかなければならない．さらに，ネットワーク構成の変化に応じてトリガグラフの構成も変化するため，各移動体があらゆる移動体サーバに接続した場合についてトリガグラフを構築する必要がある．実環境では，どのようなルールをもった移動体が接続してくるかをあらかじめ知っているという仮定は現実的でなく，マージトリガグラフを実環境で利用することは困難である．

2.3.2 AMDSにおける異常動作の実行時検出手法

前節で述べたように，移動体計算環境上では移動体の移動により，ネットワーク構成は多様であり，あらゆる場合を考えて静的検出を行うことは現実的でない．そこで，本節では実際にシステムを実行させながら，ECAルールの連鎖実行を監視することで，システムの異常動作を検出する手法を提案する．

異常検出のパラメータ

提案する実行監視機構では，ルールカウンタ，タイムスタンプという2つのパラメータを用いて異常動作を検出する．

- ルールカウンタ: 現在のECAルールの連鎖的な実行回数

- タイムスタンプ: ECAルールの連鎖実行が開始された時刻

ECAルール実行監視機構では，新たな連鎖実行が開始されるたびに，ルールカウンタとタイムスタンプを作成する．ルールが連鎖的に実行されている間，カウンタは増加し続け，値が一定以上になったり，タイムスタンプに刻まれた時刻から一定以上経っても連鎖が終了しなかった場合，異常動作が起こったと判断する．

ルールカウンタの処理

ルールカウンタは単に連鎖数をカウントするだけでなく，移動体の移動を考慮したいくつかの処理を行っている．具体的には，移動体計算環境ではネットワーク構成の変化によりそれまで記録していたカウンタの内容が無効になる場合があるため，次のような処理を行って異常動作検出の信頼性を高めている．

- 1ループ中に異なる2つの場所で，同じ移動体による連鎖の中継が起こった場合は，1回目の移動体中継以前のカウントは無効にする．
- ループを中継していた移動体が切断したら，切断以前のカウントは無効にする．

また，さまざまなホスト間を移動しながら仕事を行うタイプのアプリケーションでは，多数のホスト間を移動しながらデータを収集するようなルールが長期にわたって連鎖し続ける可能性がある．そのような場合，カウンタのリミットを大きくすることで対応できるが，あまりカウンタのリミットを大きくしてしまうと，一般の異常動作が検出できなくなる．そこで，ルールカウンタを次の2段階に分割する．

- グローバルなカウンタリミット: この値を超えたら絶対に異常動作とする値．大きな値を設定する．
- ホスト内連鎖のカウンタリミット: ホスト間にまたがる連鎖が起こった場合，それまでのカウンタの値はリセットされる．

このように，カウンタのリミットを2つ設けることによって，ホスト間を移動しながらデータを集めるタイプのアプリケーションに関してはある程度連鎖数が大きくなっても異常動作とならず，かつ，実際にホスト内で異常動作が起こった場合の検出は行える．

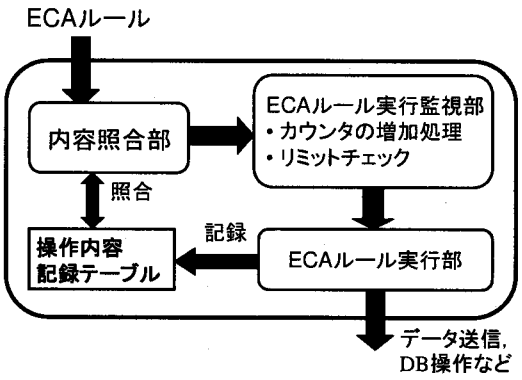


図 2.8: ECA ルール処理機構の構成

実行監視機構の設計

AMDS では，イベント検出機構において検出したイベントが，新たに起こったイベントであるのか，あるいは他のルールのアクションから引き起こされた連鎖中のイベントなのかが判断できない．そこで，図2.4における ECA ルール処理機構を図2.8に示すように拡張し，イベントの連鎖性を判断する機構を組み込む．

ECA ルール実行時に，そのアクション内容およびカウンタ，タイムスタンプの値を操作内容記録テーブルに記録しておく．新たに ECA ルールが実行される場合は，内容照合部において，操作内容記録テーブルに記録されたアクション内容と，現在のイベント内容を比較することで，どのルールのアクションから連鎖したのかを判断する．連鎖でないと判断した場合は，新たにカウンタを設置する．ECA ルール実行監視部では，カウンタの増加処理や，カウンタ，タイムスタンプのリミットチェックを行う．

アクションがデータ送信であった場合は，現在のカウンタとタイムスタンプの値を送信パケットに付加する．受信側は，データ受信イベントを処理する際，カウンタとタイムスタンプを復元して用いる．

また，ループの中継移動体が切断したらそれ以前のカウントを無効にする拡張を行うためには，操作内容記録テーブルに，ループを中継した移動体情報と，そこまでのカウントの値も記録する必要がある．そして，移動体の切断イベントを処理する際に操作内容記録テーブルをチェックし，該当移動体が含まれるデータを書き換える．また，中継移動体情報を用いれば，移動体の中継がループしているかどうかがわかるため，ホストのループの有無によって処理を切り替える拡張も可能となる．

リミット値の設定

AMDSで構築するアプリケーションはいくつかのパターンに分類でき，その際のタイムスタンプ・カウンタリミットは次のように設定する．

- リアルタイム性が必要なアプリケーション

  セル内に存在する人を1秒ごとに記録するなど，処理に時間制約をもつアプリケーションでは，タイムスタンプのリミットを短く設定することによって，制限時間内に達成できなかった処理を検出できる．

- 一つの処理に長時間かかるアプリケーション

  全国の社員のデータを収集するなど，処理に非常に時間がかかるアプリケーションでは，タイムスタンプは利用せず，カウンタの値を大きくするか，あるいはホスト内とホスト外で別のカウンタリミットを用いることで，信頼性の高い異常検出を行える．

- その他のアプリケーション

  処理時間が短く，リアルタイム性を考慮しなくてもよいアプリケーションでは，タイムスタンプとルールカウンタを統合利用することで，異常動作の検出を行える．

このように，ルールカウンタとタイムスタンプのリミットを適切に設定することで，さまざまなタイプのアプリケーションに対応できる．異常動作検出後の処理に関しては2.3.5節で述べる．

2.3.3 AMDSにおける異常動作の直前検出手法

2.3.2節で述べたように，トリガグラフを用いた静的検出をAMDS上で実現するのは現実的でない．そこで，前節ではAMDS上で実行時検出を行う手法を提案した．実行時検出は，アプリケーションを実際に動作させる際の安全性向上に対して大きな効果が期待できる．しかし，実行時検出は実際に異常が発生してからその検出を行うので，システムを停止したくない場合や，システムの安全性をチェックしておきたい場合には用いることができない．そのため，やはり異常が実際に発生する前にその可能性を調べる手法が必要となる．そこで，本節では直前検出と呼ぶ新たな異常動作検出手法を提案する．直前検出手法では，ネットワーク構成の変化に応じてトリガ情報をホスト間で送受信してトリガグラフを再構築する．本手法は，トリガグラフを用いて異常動作を検出するため，実際に異常が

発生する前にその可能性を調べられる．また，ネットワーク構成の変化に応じて必要な情報だけをやり取りするため，新たな移動体の接続にも対処でき，トリガグラフの計算量も削減できる．

直前検出手法におけるトリガグラフの構築は以下のステップで行う．

1. 自ホストのトリガグラフ生成と無限ループ検出．
2. RSパスの検出．
3. RSパスの縮退．
4. RSパスの送信．
5. 他ホストでのRSパスの受信処理．

まず，各ホストがそれぞれのトリガグラフを作成し，ローカルな無限ループの検出を行う．ここで異常が検出されなかった場合，ホスト間にまたがった連鎖を引き起こす可能性のある部分トリガグラフを他ホストに送信する．この部分グラフを**RS(Receive-Send)パス**と呼ぶ．他のホストではこのRSパスを受け取ったら，その情報を含めて再び無限ループの検出処理を行う．

以下，それぞれのステップについて説明する．

トリガグラフ生成のタイミング

自ホストのトリガグラフ生成が必要となったとき，アルゴリズムが開始される．トリガグラフ生成が必要となるタイミングは以下のうちいずれかとなる．

- システム開始時．
- ネットワーク構成が変化したとき．

  ネットワーク構成の変化によりトリガグラフの再構成が必要になる．具体的には移動体サーバがCONNECTイベントまたはDISCONNECTイベントを検出した場合となる．

- ECAルールの構成が変化したとき．

ECAルール構成の変化によりトリガグラフの再構成が必要になる．具体的にはINSERT_ECA，DELETE_ECAアクションの実行によるECAルールの追加・削除，ENABLE_ECAアクション，DISABLE_ECAアクションを実行によるECAルールの動作停止・解除が行われた場合となる．

- 他のホストからRSパス情報を受け取ったとき．

  他のホストのトリガグラフ情報の更新により，RSパス情報が送信される．それを受け取った場合，自ホストのトリガグラフも再構築する必要がある．

自ホストのトリガグラフ作成

自ホストのトリガグラフを構築する作業は次のような手順で行う．

1. パス集合の作成．

   自ホストのルール集合に属するルールすべてについて，そのアクションが発火させる可能性のあるルールがあればそのパスをパス集合に加える．

2. パス集合から連鎖パスを生成．

   パス集合のすべての要素について，パス集合の要素を用いてパスを連結する操作を繰り返し，ループを検出する．本手法では，あるルールから始まった連鎖が再び同じルールを発火させた場合ループであると判断する．もしループとなるパスが存在しなければトリガパス生成は正常終了する．

3. コンディションのチェック．

   ループが存在した場合，それが無限ループになるかどうかを判断するために以下の手順でコンディションのチェックを行う．

   (a) NEW，OLD変数の置換．

   コンディションに，NEW変数やOLD変数が用いられていた場合，そのルールを引き起こしたルールのアクションを参照し，NEWやOLDを具体的なテーブル名や値に変換する．この操作を行うことで，複数のコンディションが同じテーブルにアクセスしているかどうかが判断できる．

   (b) コンディションのマージ．

連鎖パスに沿ってコンディションをANDで連結する．コンディションを連結する際にはそのルールのアクションもチェックし，そのアクションがアクセスしているテーブルに関するコンディションは，アクション実行により満たされなくなる可能性があるのでそれまでの連結コンディションから取り除く．

(c) コンディションのチェック．

マージされたコンディションに明らかな矛盾がなければ，無限ループと判断．矛盾があればその連鎖パスは無効となる．

このような手順でトリガグラフを作成し，ホスト内での無限ループを検出する．図2.7を用いてトリガグラフの構築例を示す．図ではシステム内にR1からR5までの5つのルールが存在する．ルールxからyへのパスを(x, y)，パスを組み合わせてできる連鎖を連鎖パスと呼び，ルールx, y, zがその順で発火するとき(x, y, z)と表現すると，図2.7におけるパス集合は$\{(R1, R2), (R1, R3), (R3, R4), (R4, R1), (R5, R1)\}$となる．このパス集合から連鎖パス$(R1, R3, R4, R1)$が検出され，もしマージされたコンディションが成り立つ可能性があるならループを検出したことになる．

RSパスの作成

AMDSにおいて，図2.9に示すような，ホスト間にまたがった連鎖を引き起こすのは，SENDアクションとRECEIVEイベントの組み合わせだけである．また，ホスト間にまたがった連鎖により無限ループが発生するためには，無限ループを構成するホストにRECEIVEイベントで始まってSENDアクションで終わる連鎖パスが存在する必要がある．そこで，RECEIVEイベント→SENDアクションのパスを**RS(Receive-Send)パス**と呼ぶ．自分のホストにRSパスが存在した場合，それがホスト間にまたがった無限ループを構成する可能性があるため，関連するホストにRSパスの情報を送信することでホスト間にまたがった無限ループを検出する．RSパスを用いることでホスト間でのトリガ情報のやり取りを最小限に抑えられる．

RSパスの検出方法は次のようになる．まず，自ホストのパス集合の中からRECEIVEイベントで始まっているものを選び，前節で自ホストのトリガグラフを構築した際と同様の手法を用いて連鎖パスを検出する．もし，連鎖パスの中にSENDアクションで終わっているものがあれば，そこまでをRSパスとして検出する．

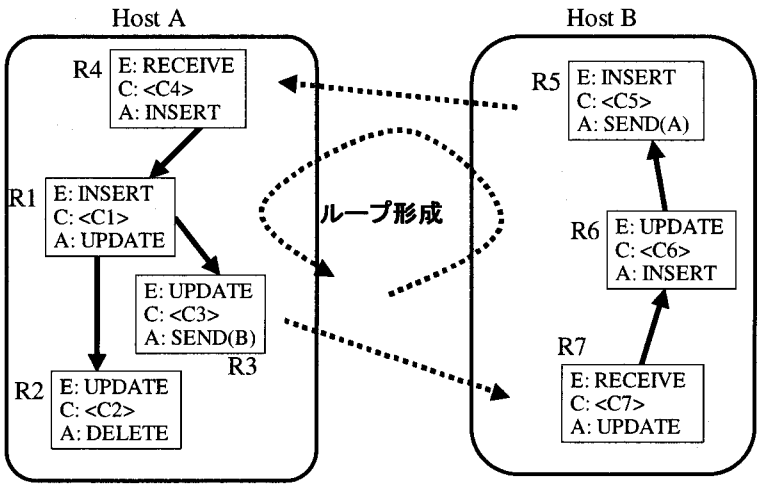


図 2.9: ホスト間にまたがった連鎖

RSパスの縮退

RSパスは送信先ホストでそのままルールの連鎖パスとして使われ，さらに他のホストに伝播することもあるため，出来上がったRSパスをそのまま送信すると通信量が多くなる．そこで，以下の手順でRSパスから不要な情報を削除し，データ量を削減する．

1. 連鎖パス内でのコンディションの縮退．

   RSパスは，連鎖を構成するすべてのルールの情報を持っているが，最終的に利用するのは先頭のルールのRECEIVEイベントの部分と最後のルールのSENDアクションの部分のみとなる．そこで，RSパスを1つのルールの形に縮退する．まず，先頭ルールのイベントと最終ルールのアクションをそれぞれ縮退後のルールのイベント，アクションとし，コンディションをマージした結果を縮退後のコンディションとする．ここで，ローカルホスト内のテーブルに関するコンディションは，RSパス送信先のホストには関係しないのですべて取り除いておく．

2. 複数の縮退RSパスのマージ．

   縮退したRSパスのうち，SENDアクションの宛先が同じものがあれば，コンディションをOR条件で統合することで，1つのルールにマージする．

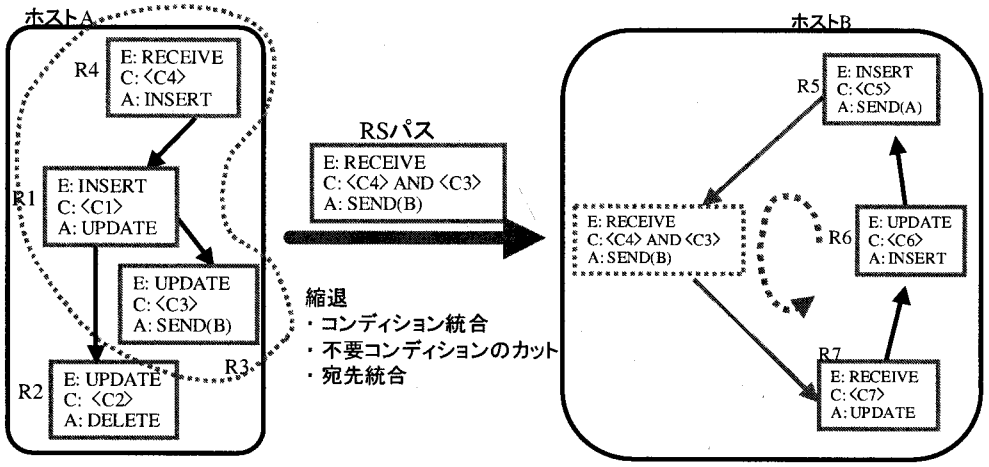


図 2.10: RSパスを用いたループ検出例

RSパスの送信

生成したRSパスを送信する．送信先ホストは以下のどちらかとなる．

- 対象のRSパスのSENDアクションに宛先が指定されている場合．

  宛先が固定されている場合は，そのホストにしかルールの連鎖が起こらないため，対象ホストのみにRSパスを送信する．

- SENDアクションに宛先がない場合．

  宛先を指定しない場合は，無線通信可能な範囲にいるすべてのホストが対象になるため，RSパスも無線通信可能なすべてのホストに送信する．

図2.10にRSパスによる無限ループ検出例を示す．ホストAにおいてアルゴリズムが開始され，R4，R1，R3がRSパスとして検出される．上に述べたアルゴリズムをもとに，RSパスはひとつのルールの形に縮退されてホストBに送られる．ホストBではこの受信ルールを用いて無限ループを検出する．

トラフィック評価

ここでは，提案手法を用いることで増加するシステムのトラフィック量の評価を行う．評価は遊園地における待ち時間取得アプリケーションをシミュレータ上に構築して行った．こ

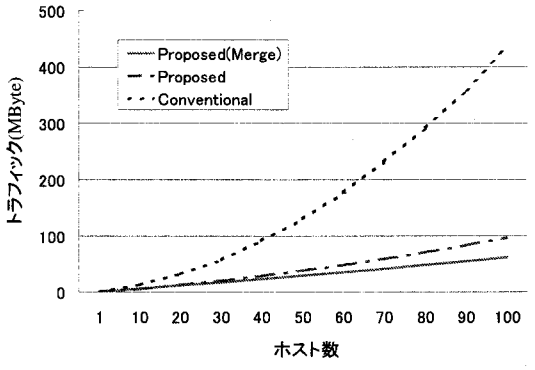


図 2.11: ホスト数とトラフィックの関係

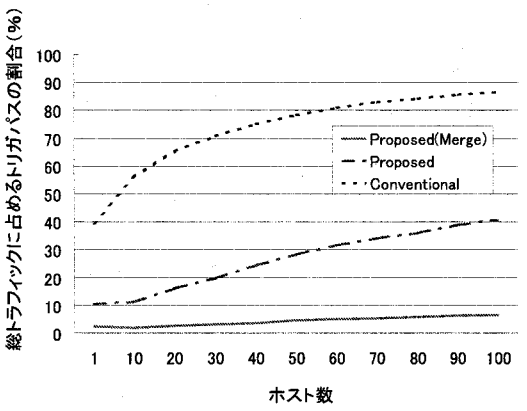


図 2.12: トラフィックとパス情報の関係

のアプリケーションは，ユーザが各施設に近づいたときに，各施設の移動体サーバが自動的に待ち時間情報およびお勧め巡回経路情報を施設情報管理サーバから取得し，結果をユーザに配信するものである．また，ユーザが自発的に施設に関する情報を問い合わせることも可能である．アプリケーションを構成するECAルール数は，移動体サーバで12個，移動体で8個であった．

各施設を表す移動体サーバを環境内に6台設置し，同時に存在する移動体の数を1〜100まで変化させたときに，提案手法がシステムのトラフィックに与える影響を調べた．評価の比較対象として，RSパスを用いるがRSパスのマージは行わない手法，およびRSパスを用いずに，すべてのホストに更新されたトリガ情報を伝播する手法のトラフィックを調べた．また，各ホスト上では，一定の確率で保持するデータベースに対する問合せや更新が発生するとした．

移動体は，500×500マスのフィールド上に配置され，1ステップにつき1マス移動できるとした．本シミュレーションでは，移動体はランダムに選んだ移動体サーバに向かって進み，たどり着いたら一定時間休む，という動作を繰り返すようにした．上記の条件において10万ステップのシミュレーションを行いその結果を調べた．結果のグラフを図2.11，2.12に示す．

図2.11は，環境内の移動体数が変化したときの総トラフィック量の変化を示している．総トラフィック量とは，実際にアプリケーションが送受信するデータ量に，異常動作検出に用いるパス情報の通信量を加えたものである．したがって，この総トラフィック量は，実環境において実際に異常動作検出機構を備えたアプリケーションを動作させるときにシステムに発生するトラフィック量を表している．

グラフより，すべてのホストでトリガグラフをもつ従来手法では，ホスト数の増加に対してトラフィック量が指数関数的に増加していることがわかる．これは，従来手法ではトリガの更新情報を接続しているすべてのホストに対して送信しており，ホスト数の増加が，トリガ情報の更新の発生頻度および送信相手の増加の両方に影響するためであると考えられる．提案手法では，RSパスを用いることによるパス情報量の減少だけでなく，関連するホストにしかトリガ情報が送信されないため，トラフィックの増加はほぼ一次関数的になっている．ホスト数100の場合で見ると，従来手法のトラフィック量は提案手法の7倍以上になっており，提案手法の有効性は明らかである．

図2.12は，移動体数を変化させたときに総トラフィックに占めるトリガ情報の割合を示している．従来手法では，かなりホスト数の少ない段階からトラフィック量の大部分をトリガ情報が占めており，異常動作検出のために大量のトラフィックを発生させていることがわかる．RSパスのマージを用いない手法では，従来手法に比べて大幅に効率が改善されているが，ホスト数100の場合でトラフィックの4割程度をトリガ情報が占めており，さらにホスト数が増大したときに対応できなくなる恐れがある．RSパスのマージを用いる手法では，常に総トラフィックの1割以下となっており，本手法を用いることで発生するトラフィック量がシステム全体にそれほど影響を与えていないことがわかる．

異常動作の検出能力について

本節では，提案手法の無限ループ検出能力について述べる．提案手法はルールの停止性を判定するアルゴリズムであるため，提案手法が正しく停止性を判定できることを証明する必要がある．まず，以下の定理を示す．本定理の証明は文献[41]に示されているものと同様であるため省略する．

[定理1] 任意のECAルール集合に対して，なんらかのイベントから到達可能なループがないとき，システムは安全である．ここで，システムが安全であるとは，無限ループによるECAルールの無限連鎖が起こらないことを意味する．

定理1より，環境内に存在するすべてのECAルールに対してトリガグラフを構築した場合，そのトリガグラフにループが存在していなければシステムは安全であることがわかる．次に以下の補題が成立する．

[補題1] ホスト間にまたがるループは必ず2つ以上のRSパスの組合せである．

[証明] AMDSのルール言語仕様より，他のホストにイベントを発火させることが可能な

アクションはSENDアクションのみであり，他のホストによって発火させられるイベントはRECEIVEイベントのみである．連鎖パスがループを形成していることから，各ホストの連鎖パスは必ずRECEIVEイベントをもつルールで始まり，SENDアクションをもつルールで終わることが示せる．□

[補題2] あるRSパスを構成するルールのいずれかを基点として，提案する検出アルゴリズムを実行した場合，そのRSパスを含むループは必ず検出される．

[証明] RSパスの終端はSENDアクションである．RSパス情報はSENDアクションの宛先に送信されるため，ホスト間の連鎖パスが存在する限りRSパス情報も送信される．補題1より，ループはRSパスとホスト間の連鎖パスにより構成されているため，あるRSパスを含むループは存在すれば必ず検出される．□

[補題3] すべてのRSパスは，そのRSパスが組織可能になったときに検出アルゴリズムによって評価される．

[証明] 連鎖パスが変化するのは，端末内でルール構成が変化するかあるいは端末間の接続関係が変化したときである．4.1.1節で述べたとおり，どちらの場合においてもそのタイミングを基点として検出アルゴリズムが実行され，すべてのRSパスが評価される．□

[補題4] RSパスの縮退が，ループ検出間違いを発生させることはない．

[証明] ループ検出アルゴリズムは，ループがあるかないかを検出するためのもので，途中のパスには関連しない．RSパスの縮退はループの接続状態を変化させないので，RSパスの縮退がループ検出間違いを引き起こすことはない．□

これらの定理・補題より，次の定理が示される．

[定理2] 提案する検出アルゴリズムによってループ検出されなければシステムは安全である．

[証明] 定理1および補題1〜4より，各端末内で完結するループは端末内のトリガグラフにより検出され，端末間にまたがるループはRSパスによって検出されることがわかる．□

定理2より，提案アルゴリズムにおいて安全であると判断された場合，システムの安全性が保障されることが分かる．一方，提案アルゴリズムにおいてループが検出された場合でも，実際にはシステムが安全である場合がある．そのような例として次のような場合が考えられる．

- 検出されたループが無限ではなく，一定回数ループすれば必ず停止するとき．
- ルールの無効化や削除アクションによって，ループが実際には成立しないとき．

前者は，例えばルールr_1→ルールr_2→ルールr_1というループが存在し，r_1のコンディションが「変数xの値が一定値以上」とされていて，r_2のアクションがxの値の減少，かつxを増加させるアクションが存在しない場合が該当する．この場合，xの値はループが実行される度に減少し，いつかはr_1のコンディションを満たさなくなる．そのため，このルールセットは無限ループではなく安全だと判断できる．後者は，例えばルールr_3→ルールr_4→ルールr_3というループが存在し，r_4のアクションがr_3を無効化する場合が該当する．

前者のパターンの検出に関しては，Leeらによって複数回ループを展開する手法が提案されている[33]．また後者のパターンの検出に関しては，Vaduvaらによってアクション-ルール間関係を用いた手法が提案されている[69]．これらの手法を採り入れることで本研究におけるループ検出の精度を高められると考えられるが，これらの手法はECAルールの言語仕様に大きな制限を加えており，そのまま本研究に適用できない．例えばVaduvaらの手法を適用できるのは，有効化・無効化されるルールがアクションの中で明示的に指定されている場合に限られる．そのため，ルールを明示的に指定するだけでなく，データベースから得た内容やNEWデータ，OLDデータの内容によるルール指定など，静的にルール指定が決まらない本システムには適用できない．

このように，本システムにおいては検出されたループが本当に無限ループであるかどうかは保障できず，さらなる判定精度の向上は今後の課題であるが，上記の手法を部分的に採用して無限ループの判断精度を向上させるといった方法が考えられる．

停止性判定中のホスト移動について

これまで，簡単化のために本アルゴリズムにおける停止性判定中にネットワーク構成の変化やルールの追加・削除といった環境の変化が起こった場合については言及していなかった．環境が変化した場合にはこれまでに述べたように，ルールの追加やホストの接続を検出したホストから新たにアルゴリズムが開始されるため，検出能力が落ちることはない．このとき，それまで実行されていたアルゴリズムのインスタンスは，すでに他のホストへ伝播している場合など停止させることが困難であるため，そのまま並行して実行される．継続して実行されるアルゴリズムが環境の変化により影響を受けるのは下記の場合である．

- 変化する前の環境ではループが存在していなかったが，環境が変化した結果ループを検出する場合．

- 変化する前の環境ではループが存在していたが，ホストが退出するなどの結果ループが成立しなくなる場合.

前者の場合，実際にループが存在するため，そのループをアルゴリズムが検出することに問題はない．一方，後者の場合はすでにRSパス中に組み込まれているホストが退出してしまった場合など，実際にはループが成立しなくなった環境においてもループがあると検出する可能性がある．しかし，退出直前の時点ではループが存在していたことから，その状況に対してループ検出をユーザに通知する本アルゴリズムの処理に問題はないと言える．したがって，停止性判定中に環境の変化が起こった場合も提案アルゴリズムは有効に動作する.

無限ループ以外の原因による異常動作について

本研究で検出する異常動作はECAルールの連鎖実行による無限ループであるが，ECAルールによるその他の異常動作として，次のような例が挙げられる.

- ECAルール生成アクションによるルールの無限生成.
- 継続的な外部からのイベント発生によるルールの無限実行.

本システムではルール生成アクションを用いて，データベースに格納されたルールをシステムに追加するといった記述ができる．この機能により，システムが無限にルールを生成し続ける可能性がある．また，頻繁かつ継続的に外部からデータベース操作等が行われることで，ルールの処理速度より発火されるべきルールの数が常に多い状況となり，ルールの無限実行状態に陥る場合がある.

これらの異常動作はECAルールセットが原因ではなく，システムによるルール数の上限設定や，端末の能力を高めることで回避する問題であるため，本研究では取り扱わない．また，これらの異常動作が起こる状況であっても，提案アルゴリズムはルールの連鎖実行による無限ループを検出できる.

関連研究

近年のトリガグラフの研究としては，トリガグラフの検出能力向上に関する研究がある[4][5][28][33][70]．従来のトリガグラフでは，イベントとアクションのみによってトリガグ

ラフを構築していたが，このような方法では実際は安全であるルール群も異常動作として判定してしまう．そこで，これらの研究ではルール中のコンディション部を考慮することによって，異常動作検出の精度を高めている．文献[70]ではコンディションとアクションの記述に強く制限を加えることで，コンディションを考慮したトリガグラフを実現し，文献[5]では代数的アプローチによりコンディションを加えたトリガグラフを構築している．文献[4]，文献[28]では一度トリガグラフを構築してから，トリガグラフの各エッジについてのコンディションを評価して，連鎖的に実行されないエッジを取り除いている．文献[33]では，トリガグラフ上でループとなった部分グラフを展開することで，無限ループなのか複数回ループして止まるのかを判定している．また，文献[32]では，トリガグラフの各エッジごとに評価するのではなく，エッジをいくつか連結した意味のあるパスごとにコンディション評価を行うことで検出精度を高めている．

また，従来のトリガグラフ構築アルゴリズムでは取り扱えなかった高度なECAルール言語仕様に対応したアルゴリズムとしては，Vaduvaらの手法がある[69]．この文献では，複数事象の同時あるいは連続発火をイベントとして記述できる混成イベントや，他のルールを有効化・無効化するアクション，イベントを発火できるコンディションなど，より複雑なECAルールが記述できるアクティブデータベースにおけるトリガグラフ構築手法を提案している．

これらの研究は，システムに存在するすべてのECAルール内容が明らかになった状態でのトリガグラフ構築と無限ループ検出に関する研究であり，ホスト間の連鎖を効率的に検出するための手法を提案している本研究とは共存できる．本研究の提案手法におけるローカルホストでのトリガグラフ構築部分にこれらの関連研究で用いられている手法を採用することで誤検出が減少すると考えられる．

2.3.4 AMDSにおける異常動作検出のまとめ

2.3節では，AMDSの異常動作検出手法について述べた．ECAルールを用いたアプリケーションではルール群が連鎖的に実行されるため，無限ループなどの異常動作が起こる可能性がある．したがって，AMDSを用いたアプリケーションを提供するためには異常動作を予防および検出する機構が必要であり，その解決方法としてAMDSにおける実行時検出と直前検出の2つの異常動作検出機構を提案した．2つの手法は選択的に用いるのではなく，相補的に用いればより高い信頼性でAMDSアプリケーションの異常動作を検出できる．

前節で述べたように，本章で提案する手法を用いて異常を検出した場合でも，それは必ずそのルール群が異常動作を起こすことを保証するものではない．したがって，異常が発生した場合も単純にシステムを停止させるだけではなく，柔軟な処理が行えることが望ましい．

そこで，AMDSにおける異常動作検出手法では，異常動作を検出した場合以下の3種類の処理を選択的に，または組み合わせて利用できるようにしている．

- ユーザへの通知．

  異常動作を検出した場合ユーザにアプリケーションの継続か停止かを選択させる．継続する場合，ループを形成している連鎖パスを危険パスとし，危険パスが実際に発火した場合にはそれ以降のイベント発火とルール連鎖をトレースしてログに記録する．異常が発生した場合，ログを利用することで原因の解決を図ることは一般的であるが，本手法では危険性がある場合のログだけが記録されるため，ログ記録処理のオーバーヘッドは少なくてすむ．

- 異常発生原因の除去．

  異常動作の原因となったルールやホストを切り離すことで無限ループを回避する．例えば移動体が接続してきたことにより，トリガグラフがループを検出した場合，その移動体との通信を拒否するといった方法で原因の除去を行う．

- エラーイベントを発生させる．

  異常動作を検出したときの状況をNEWデータにもつERRORイベントを発生させる．したがって，ERRORイベントを処理するECAルールを記述しておくことで柔軟なエラー処理が可能となる．

2.4 AMDSシミュレータ

AMDSを用いて移動体アプリケーションを構築する場合，アプリケーションの機能をECAルールの組で表現することになるが，アプリケーションを実際に動作させるためには，まず環境内のあらゆるホストにそのホスト用に作成したECAルールを配置し，さらに実際に移動体を動かすことで移動処理のチェックを行う必要がある．このようにAMDSで移動体アプリケーションを構築する際には次のような問題点がある．

- AMDSはあらゆる端末上に存在するため，アプリケーションをテストするたびにそれぞれの端末にその端末用のルールを設定する必要がある．
- アプリケーションの動作は，移動体のセルへの接続や切断動作を起点とするものが多いため，アプリケーションをテストするためには，ECAルールを設定した端末を実際に動かして接続・切断動作を行う必要がある．
- ECAルールは連鎖的に動作させることで複雑な動作を実現可能であるが，連鎖的なルール実行は無限ループなどの予期せぬ異常動作を引き起こす可能性がある．このような異常動作の発生をチェックしたり，アプリケーションのトラフィック量を予想するためには，多数のホストをさまざまなパターンで動作させる必要がある．

したがってこれらの問題点を解決するためには，多数のホストを仮想的に設置し，各ホストにECAルールを設定してさまざまなパターンで動作させ，デバッグやトラフィック測定を容易に行えるようなアプリケーション開発・シミュレーション環境が必要になる．

2.4.1 システムの設計

前節に述べた要求に対し，本研究ではAMDSのためのアプリケーション開発・シミュレーション環境であるAMDSシミュレータを構築する．AMDSシミュレータは，仮想的なフィールド上に移動体や移動体サーバを実際に配置して動作させることで移動体アプリケーションのシミュレーションを行うシステムである．配置したホストには自由にECAルールをもたせ，また移動モデルを設定できるため，トラフィック解析などのシミュレーションを行うだけでなく，アプリケーション開発にも利用できる．AMDSシミュレータは以下の特徴をもつ．

1. 多数の移動体を設置し，自由に移動させることができるので，大規模なアプリケーションのシミュレーションが可能．
2. 異常動作検出機能の使用/不使用など，AMDSがもつ機能の詳細について使用するかどうかを選択できるため，各機能が性能に与える影響を測定できる．
3. グラフィカルなインタフェースで直接ホスト情報を閲覧/修正できるため，アプリケーションを少しずつ修正しながらテストが行える．

4. 移動体の移動パターンは，ランダムに動くものや特定の移動体サーバに向かうものの他に，自由にシナリオを書いて動作させられるため，さまざまな移動モデルをシミュレートできる．
5. 移動体が通れない場所を表現する障害物を自由に設置できるため，想定環境を忠実に再現できる．
6. シミュレーション中は実際に仮想フィールド上で移動体の動作状況を確認できるため，エラーが起こったホストや特定のイベントが起こったホストの色を変化させて視覚的にアプリケーション実行状況を把握したり，地理的なホストの分布を確認しながらアプリケーションのテストを行える．

AMDSシミュレータの概観を図2.13に示す．AMDSシミュレータはその機能ごとに，メイン画面，ホスト画面，ルール画面，移動モデル設定画面，オプション画面，解析画面の6つの画面に分かれている．それぞれについて以下に述べる．

メイン画面

図2.14に示すメイン画面では，ホストの配置や障害物の設置などを行い，シミュレーション環境を構築する．設置したホストをクリックすることでホスト詳細画面，画面右に並んだボタンを押すことで解析画面やパラメータ設定画面を表示する．また，シミュレーションの実行/停止や，セーブ/ロードもこの画面から行う．シミュレーション実行中にはウォッチウインドウに現在注目しているホストの詳細情報が表示される．

ホスト画面

図2.15に示すホスト画面では各ホストの詳細情報を表示する．各分割ウインドウにはそれぞれホストがもつECAルールのトリガグラフ，無限ループ情報，RSパス情報が表示されている．この画面では，トリガグラフによりホストがもつルール群の関係を視覚的に表示し，無限ループを構成する可能性がある場合はその部分を抜き出して別ウインドウに表示している．また，2.3節で述べた異常動作検出で用いるRSパスの情報も別ウインドウに表示している．トリガグラフ中の各円はそれぞれ1つのルールを表しており，円をつなぐ線がルール間の連鎖状況を示している．ホスト画面では以下の操作が可能である．

ホスト画面では，ホスト位置や問合せ発生率などのパラメータを修正できる．特に，移動モデルを変えるときは，次に述べる移動モデル設定画面が新たに立ち上がる．また，各

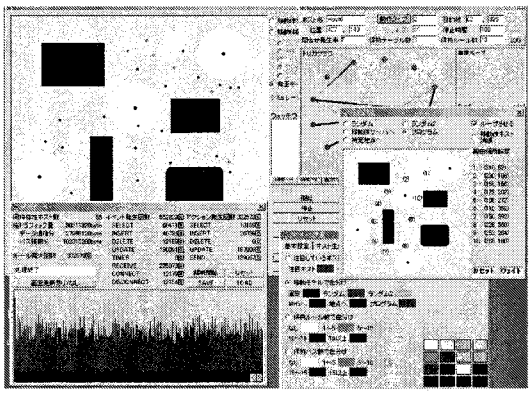


図 2.13: AMDSシミュレータの概観

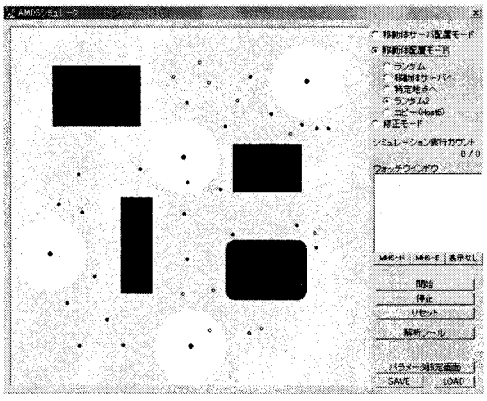


図 2.14: メイン画面

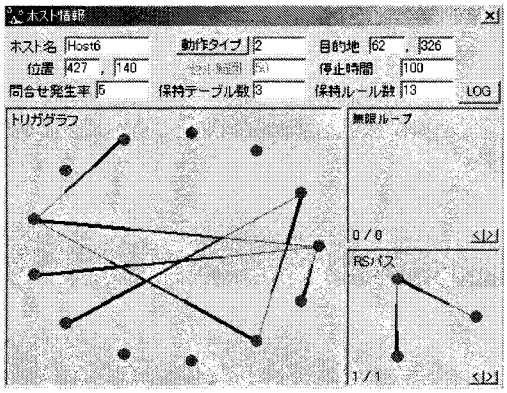


図 2.15: ホスト画面

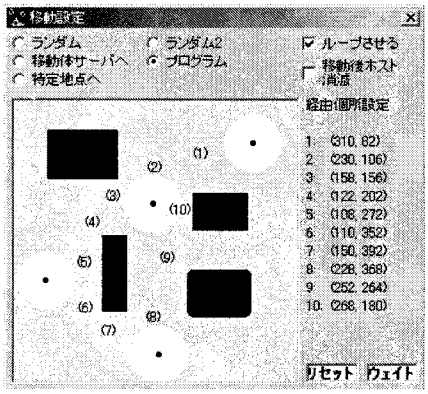


図 2.16: 移動モデル設定画面

ルールを右クリックすることで，ルール設定画面が現れ，各ルールの内容を変更できる．さらに，LOGボタンを押すことで，自分のホストに関するシミュレーションログを抽出して表示できる．

移動モデル設定画面

図2.16に示す移動モデル設定画面では，ホストの移動モデルの変更と経路設定を行う．選択できる移動モデルとしては，ランダムに移動するもの，画面上のMHSのうちどれかに向かっていく動きをするもの，プログラムされた動きをするものなどを用意している．動作をプログラムする場合は，メイン画面を縮小表示した経路設定画面を用いてホストの移動

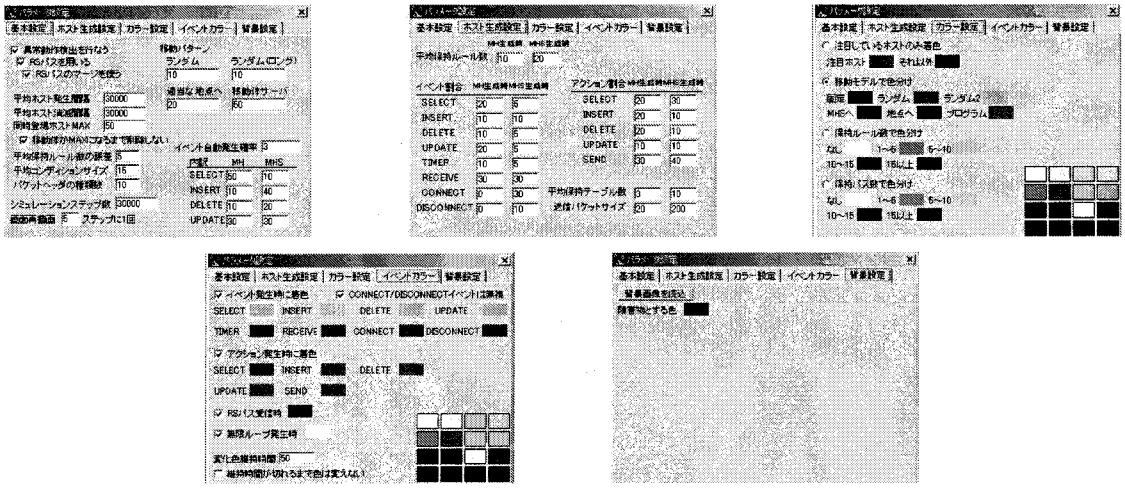


図 2.17: オプション画面

経路を設定する.

オプション画面

図2.17に示すオプション画面では，AMDSシミュレータのさまざまなパラメータを設定する．設定項目を以下に示す.

- 基本設定: 異常動作検出機構を使用するか，RSパスを用いるかといったAMDSの運用形態の設定を行う．また，同時登場ホスト数やシミュレーションステップ数など，シミュレーション自体の設定を行う.
- ホスト生成設定: ホストを自動生成する際の，移動パターンやルールの自動作成パラメータを設定する.
- カラー設定: ホストの色分け方針と使用する色を設定する．また，背景画像および障害物の設定を行う.

解析画面

解析画面を図2.18に示す．解析画面ではシミュレーションを行った結果を集計し，グラフ化して表示する．集計項目は各イベントの発生数やアクション実行数のほか，総トラフィックやその内訳など多岐に渡る．シミュレーション結果はファイルに読み書きできる.

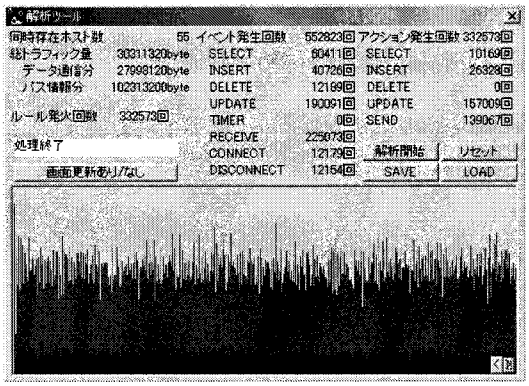


図 2.18: 解析画面

2.4.2 AMDSシミュレータの用途

AMDSシミュレータを用いたアプリケーション開発

一般に，分散環境におけるクライアント/サーバアプリケーションを構築する場合，サーバ/クライアントのアプリケーションをそれぞれ実際に構築してテストするか，またはスタブやプロキシなどを用いてテストを行うのが一般的である．しかし，AMDSは，移動体の移動がアプリケーションの実行に大きな影響を与えるため，移動体の移動をなんらかの形で実現する必要がある．これまでは，実際にノートパソコン等の携帯端末上にクライアント用プログラムを配置し，実際に携帯端末を移動体サーバのセルに出し入れしてアプリケーションのテストを行っていた．しかし，この方法ではアプリケーションを修正するたびに携帯端末にルールを配置する必要があり，また多数のクライアントを想定したアプリケーションのテストが行えないという問題がある．

AMDSシミュレータを用いてアプリケーションを開発する場合，画面内に多数のホストを配置してそれぞれのホストを柔軟に動作させることができるため，多数のユーザを想定したアプリケーションのテストが可能である．また，画面内には自由に障害物を配置できるため，実際の環境における施設レイアウトに応じたシミュレーションが可能である．さらに，各ホストのECAルールを設計する際には，トリガグラフ情報や無限ループ情報を提供して，ECAルール設計を補助している．デバッグ時には，ホストの動作状況を実際に画面上で見ることができ，特定のイベント発生時など条件に応じてホストの色を変化させられるため，ホストの地理的な偏りや全体的なイベント発生状況を把握しながらデバッグを行

える．このように，AMDSシミュレータを用いてアプリケーションを開発することで，多数の移動体を対象とするアプリケーションを容易に構築できる．

AMDSシミュレータを用いたトラフィック解析

AMDSシミュレータでは，構築した移動体アプリケーションを実際に動作させ，そのトラフィックを測定できる．ここでは，トラフィック解析の一例としていくつかのアプリケーションを実際に構築し，アプリケーション特性に対するトラフィック量の変化を調べる．本研究では以下の3つのアプリケーションを実際に構築した．

- 博物館アプリケーション

  博物館内に5つの移動体サーバを設置し，それぞれが展示物を表している．ユーザが展示物に近づくと，自動的に移動体サーバから展示物の詳細情報が送られ，移動体に表示される．もしユーザが特定の展示物に対する情報要求を出した場合は，他の移動体サーバに問合せて返送する．ホストがもつ平均ルール数は7個．このタイプのアプリケーションの特徴は，移動体サーバが通信をするのはほとんど自分のセル内の移動体に対してであり，移動体サーバ同士の通信があまり起こらないことである．

- 遊園地アプリケーション

  遊園地内に5つの移動体サーバを設置し，それぞれがアトラクションを表している．ユーザがアトラクションに近づくと，自動的に移動体サーバは待ち時間管理サーバにアトラクションの待ち時間情報を問合せ，その結果を元にユーザに待ち時間情報とお勧め経路情報を提供する．ユーザが特定のアトラクションの情報を要求した場合は，接続している移動体サーバが待ち時間管理サーバに問合せて結果を返送する．ホストがもつ平均ルール数は13個．このタイプのアプリケーションの特徴は，移動体サーバ同士が頻繁に通信することである．

- 掲示板アプリケーション

  環境内に5つの移動体サーバを設置し，それぞれが電子掲示板を表している．ユーザが掲示板に近づくと，移動体サーバは掲示されている情報を接続移動体に対して送信する．移動体が新たな掲示情報を移動体サーバに送信した場合，移動体サーバはその時点で自分のセル内に存在するすべての移動体に対して更新情報を送信する．ホストがもつ平均ルール数は7個．このタイプのアプリケーションの特徴は，移動体サーバ

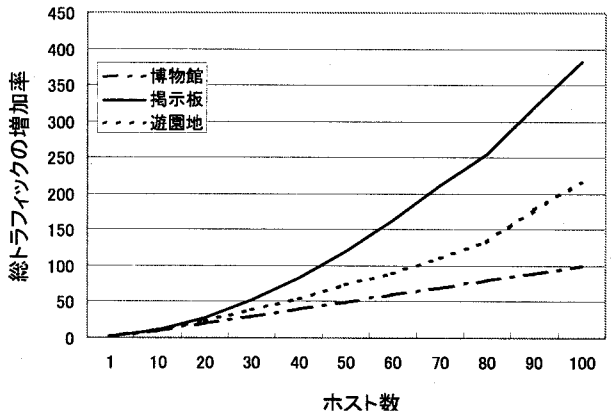


図 2.19: ホスト数に対する総トラフィックの変化

の情報が更新されたときに，その情報がセル内のすべてのホストに対して送信されることである．

これらの3つのアプリケーションをAMDSシミュレータ上に実際に構築し，アプリケーションの特性の違いによるトラフィック量の変化を調べた．移動体は500×500マスのフィールド上に配置し，1ステップにつき1マス移動できるとした．同時に存在する移動体数を1～100まで変化させて10万ステップのシミュレーションを行った．

ホスト数の増加に対するトラフィック量の増加割合を図2.19に示す．博物館アプリケーションでは，ホスト数とトラフィック増加量がほぼ比例関係になっており，ホスト数が100倍になればトラフィック量も100倍になっている．これは，移動体サーバが移動体と1対1でサービスを行う関係にあり，移動体サーバへの同時接続数がトラフィックに影響を与えないためであると考えられる．一方，遊園地アプリケーションおよび掲示板アプリケーションは2次関数的な増加傾向を示している．これは，移動体サーバ同士が通信しており，トリガグラフを作成する場合などに通信先の移動体サーバに接続している移動体の数がトラフィックに影響するためであるといえる．特に掲示板アプリケーションでは，移動体上でデータが更新された際に自分のセル内にいるホストにデータを送信するため，セル内に同時に存在するホスト数がトラフィックに大きな影響を与えている．

2.5 むすび

本章ではモバイル環境におけるアクティブデータベースシステムであるAMDSにおいて安全にアプリケーションを開発・運用するためのいくつかの機構について述べた．ECAルールの無限連鎖による異常動作の発生を回避するために，ECAルールの実行監視機構および直前検出機構を構築し，アプリケーションの開発・テストの困難さを解消するために，AMDSシミュレータを構築した．これらの機構を用いることで，安全なアプリケーションを効率的に開発・運用できるようになる．今後の課題としては，各異常動作検出手法の実機での評価が挙げられる．また，移動体計算環境においてECAルールで動作するアプリケーション群を実際に構築して運用することで，言語仕様の見直しやシステムの最適化手法の提案を行う予定である．

第3章

位置依存サービスのための地理情報変換機能をもつアクティブデータベース

3.1 まえがき

モバイルコンピューティングが普及してユーザがPDAやノートPCなどの携帯端末をいつでも持ち歩くようになると，携帯端末上で近辺地図の表示や道案内を行うといった位置依存のサービスを提供する地理情報システムを利用したいという要求が生じる．地理情報システム（GIS: Geographic Information System）とは，デジタル化された地図をベースに，様々な情報を付加して加工・分析し，人間にわかりやすい形にビジュアル化するシステムである．

一般にGISを利用する目的としては，地図上の距離や，体積などを調べる地理的計量処理，地図要素間の空間関係を分析する近接分析，点，線，面などの図形から等しい距離にある領域を確定し，小売店の商圏などを調査するバッファリング，属性の異なる複数のレイヤを重ね合わせて新しい主題図を作るオーバーレイ，最短経路の探索や巡回問題などを処理するネットワーク分析などが挙げられる[54].

実際の商品では，エプソン社のロカティオやソニー社のVAIO PCG-C2GPS，エンペックス社のポケナビのように，GPSを搭載した携帯端末がすでに販売されており，位置情報を利用したサービスが行われている．また，OGC（Open GIS Consortium）[45]の活動や，G-XML[12]，POIX（Point Of Interest eXchange Language）[46]，RWML（Road Web Markup Language）[10]など，地理情報記述の標準化活動がさかんに行われている．

このように，屋外で携帯端末を用いて地理情報システムを利用する環境が整ってきたといえるが，従来のシステムには次のような問題点がある．

- 情報の統合利用が難しい

  従来のシステムやアプリケーションは互換性がないため，あるシステムで利用できる情報が他のシステムでは利用できないという問題がある．モバイル環境における地理情報システムでは，各種のアプリケーションを連携動作させた柔軟なサービスが要求されるため，アプリケーション間の連携が必要となる．

- 多様な位置依存サービスへの要求に対応できない

  屋外での地理情報システムを利用する場合，以下のような位置依存サービスへの要求がでてくる．

  - 遊園地などのアミューズメント施設で，各アトラクションの待ち時間等の情報を端末上に表示する．また，訪れたいアトラクション情報を端末に入力しておくことで，効率的な順番にナビゲートしてくれる．
  - 買いたい本の情報を端末に入力しておくことで，自分が本屋の近くに来たときに，自動的にその本屋のデータベースを検索して本の在庫や価格などの情報を得てユーザに提示する．
  - 観光地などで，名所に近づくと，その詳細を説明してくれたり，順路を教えてくれる．

  ここで示した例はいずれも，その時間，その場所にいる人にとって役に立つ情報を提供する例である．特定の場所に行ってオブジェクトを発見する実世界におけるアドベンチャーゲームを提供したり，特定の日時に特定の場所に行かなければ見られないバーチャル花火のイベントを行うといったサービスは，位置依存だから行えるサービスであり，位置依存であるから情報の価値が高まる[56]．いくつかの位置依存サービスは既存の商品で実現されているが，サーバとインタラクションを行なう複雑なサービスは提供できていない．

- 柔軟なデータ管理機構をもっていない

  地図上のオブジェクトデータは膨大な量であるが，携帯端末の記憶容量は限られているため，自分が今いる場所に近いオブジェクト情報だけを得ることが必要となる．た

だし，携帯端末の移動により近接するオブジェクトは変化するため，ユーザの動きに合わせて近接するオブジェクトのデータを，随時受け取り，更新する必要がある．

また，端末上で近辺の情報を取得したり，必要なデータを記録するといったサービスへの要求が出てくるが，そのようなサービスを実現するためには柔軟なデータ管理機構が必要となる．

したがって，本章ではこれらの問題点を解決する地理情報システムの構築を目的とする．多様な位置依存サービスを実現するためには，位置依存の情報や，周囲のオブジェクトデータを提供する移動体サーバを各地に設置した上で，移動体から，または移動体上でデータを収集する必要がある．また，提案するシステムは，ユーザのオブジェクトへの接近やデータの到着といった突発的な事象を取り扱う必要がある．そこで，本章ではアクティブデータベースを用いてこれらの要求を満たす地理情報システムActiveGISを構築する．ActiveGISでは柔軟な位置依存サービスを提供するため，ECAルールを用いた地理情報の動的変換機能および地図上のオブジェクトとの位置関係に応じたイベント発生機能を提供する．また，本章では地理情報変換のためのECAルールであるECA-MLを提案し，地理情報にECA-MLを組み込んで配信することで期限付き情報配信や個人化した地図の作成など柔軟なサービスが提供できることを示す．以下，3.2節ではシステムの設計について述べ，システムの動作記述言語であるECA-MLの言語仕様およびサービス例について詳細に解説する．3.3節ではシステムの実装について述べ，3.4節では提案システムの考察を行う．最後に3.5節で本章のまとめを行う．

3.2 システムの設計

本節ではActiveGISの設計について述べる．ActiveGISは，GPS機器と無線通信機能および電子地図を備えた携帯端末にアクティブデータベースを組み込んでいくつかの拡張を行ったシステムである．ActiveGISでは，一般のアクティブデータベースで提供されているイベント・アクションに加えて，地図オブジェクトに対するイベント（オブジェクトへの近接・離脱）やアクション（地図の表示やオブジェクトの形状変更など）を提供している．以下，まずActiveGISで用いる地理情報のデータフォーマットであるG-XMLについて説明し，次にActiveGISの動作記述言語であるECA-MLについて説明する．

3.2.1 地理情報フォーマットG-XML

3.1節で述べたように，近年地理情報の標準化がさかんに行われており，本システムにおいてもそれらの標準化された地理情報フォーマットのひとつであるG-XML(第1版)を用いる．G-XMLはXML形式の記述仕様に従った空間コンテンツ相互流通のためのプロトコルであり，G-XML標準化検討委員会によりプロトコルの検討及びプロトタイプの開発が行なわれている．G-XMLはReal World G-XML，Point&Direction Based G-XML，Graphic Based G-XML，Semantic G-XMLの4種類の文書型定義(DTD)に基づく実装仕様と，DTDだけでは既定できない制約条件等を示した運用仕様から構成されている．以下にそれぞれのG-XML定義の概要を示す．

- Real World G-XML: 実世界の事象を抽象化した地物を記述する．
- Point&Direction Based G-XML: ある時点のある地点に関する情報を記述する．
- Graphic Based G-XML: 地図におけるグラフィックス表現を記述する．W3C(The World Wide Web Consortium)が仕様化を進めているSVG(Scalable Vector Graphics)をもとに，地図固有の付加情報を追加したもの．
- Semantic G-XML: 略地図表現のための構造化情報を記述する．

このように，G-XMLは目的に応じた4種類の仕様を提供しており，さまざまなアプリケーション要求に対応できる．

図3.1にG-XMLの記述例を示す．図左上はReal World G-XMLの記述例で，本屋や道路といった具体的な地物を図形表現で直接表現できる．図左下はSemantic G-XMLの記述例で，地理上のノードやパス，関連情報などを用いて略地図を表現している．G-XMLは地理オブジェクト間のリンク機能を提供しているため，図右に示すようにReal World G-XMLで表現した地図上にSemantic G-XMLで表現した略地図を重ねるといったように，組合せて利用することが想定されている．

3.2.2 ECA-ML言語仕様

ActiveGISの動作記述言語ECA-MLは，ECAルール記述をXML形式で表したものである．ECA-MLを用いることで，XML形式でルールを送受信でき，また，G-XML記述に

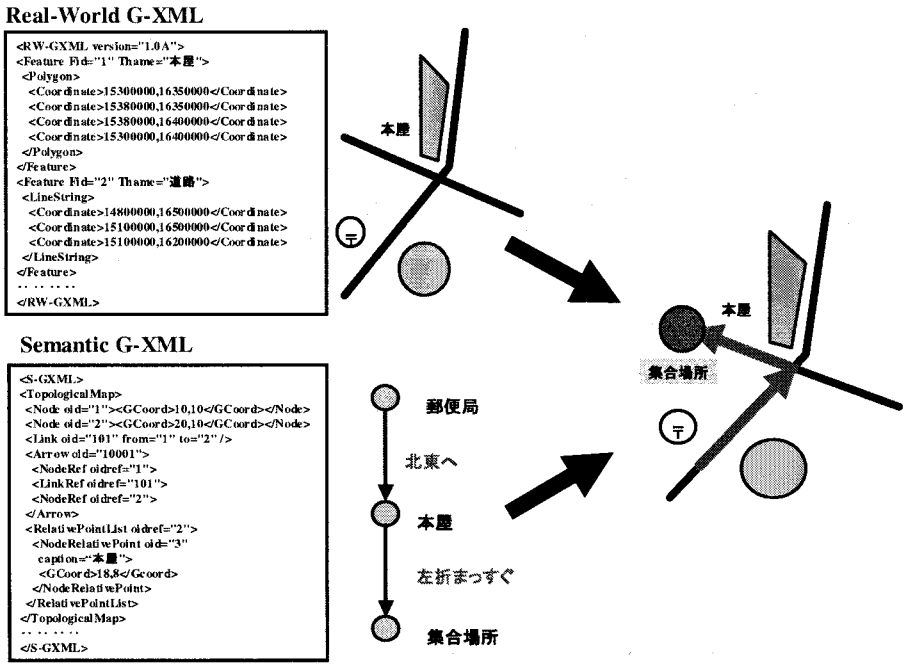


図 3.1: G-XML 記述例

ECA-ML 記述を含めることも可能になる．また，G-XML の要素を直接処理することで目的別に地図を作り変えたり，動的にオブジェクトの色を変更したいという要求に応えるため，ECA-ML ではアクション部に XML の文書変換言語である XSLT をサポートしている．XSLT を用いることで，G-XML 文書を柔軟に処理できる．

ECA-ML の記述構文を図 3.2に示す．ECA-ML は`ECARULE`タグで囲んで記述する．`ECARULE`タグには属性として [ルール ID] と [ルール適用範囲] がある．[ルール ID] はルールを一意に識別できる識別子を記述し，[ルール適用範囲] にはルールが適用される範囲を記述する．適用範囲に 'internal' を指定した場合，ルールが適用される範囲はその文書だけとなる．したがって，G-XML 記述内に 'internal' を指定した ECA-ML 文書を含めておくことで，その文書における G-XML 記述だけに対するルールを記述できる．

`EVENT`タグにはイベントを記述する．イベントの種類を [イベントタイプ] に記述し，イベントの対象を [イベント対象] に記述する．ECA-ML で指定できるイベントタイプの一覧を表 3.1に示す．[イベント対象] は，データベース関連のイベントでは対象データベースやテーブルを記述し，オブジェクトへの近接などのイベントでは，G-XML 中のオブジェクト ID を指定することで，特定のオブジェクトに対するイベントを処理できる．

```
<ECARULE id="[ルール ID]" scope="[ルール適用範囲]">
  <EVENT type="[イベントタイプ]"
         target="[イベント対象]" />
  <CONDITION>
   (<AND> や <OR> などコンディション間制約)
      <ITEM type="[演算子]">
        [コンディション記述]
      </ITEM>
      <ITEM>...
  </CONDITION>
  <ACTION>
    <ITEM type="[アクションタイプ]"
          ret="[返り値格納変数]">
      [アクション記述]
      <VARIABLE>
        [代入変数記述]
      </VARIABLE>
    </ITEM>
    <ITEM>...
  </ACTION>
</ECARULE>
```

図 3.2: ECA-ML の構文

CONDITION タグにはコンディションを記述する．コンディションの記述はITEM タグで行なう．制約条件を複数記述できるようにするため，ITEM タグはCONDITION タグ中に複数記述でき，また，それらをAND タグやOR タグで囲むことで複数コンディション間の関係を記述できる．コンディションはITEM タグの要素として‘[左辺], [右辺]’の形式で記述する．両辺には，データベース属性，受信パケット内容や定数を記述し，比較演算子は [演算子] 属性に記述する．

ACTION にはルールのアクションを記述する．各アクションの記述はITEM タグで行なう．アクションとしてはデータベース操作アクションやパケット送信アクションの他に，XML 記述の変換言語であるXSLT 記述を利用できるようになっている．実際のアクション内容はITEM タグの要素として記述する．type 属性にはアクションの種類を記述し，ret 属性にはアクションの実行結果が代入される変数名を記述する．ECA-ML で記述できるアクションを表3.2に示す．アクション内で変数を利用したい場合は，VARIABLE タグを利用する．要素に‘[置換文字名] : [変数名]’と記述しておくことで，実際のアクションが実行されるときに変数内容に置換される．

表 3.1: イベントタイプ一覧

名称	内容
SELECT	テーブルに対するデータ参照
INSERT	テーブルに対するタップルの挿入
DELETE	テーブルのタップル削除
UPDATE	テーブルのタップル更新
RECEIVE	データパケットの受信
CONNECT	移動体のセルへの接続
DISCONNECT	移動体のセルからの退出
TIMER	設定したタイマの発火
CLOSE	オブジェクトへの接近
AWAY	オブジェクトから離れた

表 3.2: アクションタイプ一覧

名称	内容
QUERY	データベース操作
SEND	データの送信
INSERT_ECA	ECA ルール格納
DELETE_ECA	ECA ルール削除
ENABLE_ECA	ECA ルール有効化
DISABLE_ECA	ECA ルール無効化
SET_TIMER	新たなタイマの設定
KILL_TIMER	タイマの削除
DISPLAY	画面表示
XML_PROC	G-XML を格納して表示
XSLT	XML 変換処理

表 3.3: NEW データと OLD データの内容

イベント	NEW	OLD
SELECT	参照タップル	–
INSERT	挿入タップル	–
DELETE	–	削除タップル
UPDATE	更新後タップル	更新前タップル
RECEIVE	到着パケット内容	–
CONNECT	接続移動体情報	–
DISCONNECT	–	切断移動体情報
TIMER	タイマ識別子	–
CLOSE	近接した地物情報	–
AWAY	–	離脱した地物情報

また，ECA-ML では，2 章で述べた AMDS と同様に NEW データおよび OLD データと呼ぶシステム変数が利用できる．イベントが発生したときにはこれらの変数に必要な情報が格納され，コンディション部やアクション部においてこれらの変数を自由に利用できる．各イベントが発生したときの NEW データ，OLD データの内容を表 3.3に示す．

3.2.3 サービス例

これまでに述べたように，G-XML を用いることでアプリケーション間で互換性を保った地理情報サービスが実現でき，ECA-ML を用いることで地理情報を利用したさまざまなサービスが提供できる．典型的な ActiveGIS の利用例は次のようになる．

1. 目的別地図作成

建物のデータを配信するサーバからデータを受信したとき，建物の種類などの条件によってローカルに蓄積するかどうかを決めるルールを用いることで，レストラン地図やショッピング地図など，ユーザが要求する地物だけからなる地図を表示できる．

2. ローカル情報の取得

建物に近づいたとき，特定種類の建物であったらサーバから情報を得るルールを用いることで，例えば観光地において名所に近づいたときにその詳細を説明したり，観光のための順路を地図上に表示するサービスが実現できる．

3. ローカルサービスの利用

ローカル情報の取得だけでなく，サーバとやり取りを行うことで，サービスを受けることができる．例えば，本屋に近づいたときに，ルールにより自動的に個人情報から買いたい本の情報を検索して本屋のサーバに送信し，サーバ側のルールにより，検索結果を返してもらう，といった複雑なサービスが実現できる．このようなルールを用いると，商店街を歩くだけで自動的に自分が欲しい商品の特価情報が得られ，安く売っている店の前に来たときに教えてくれるようなサービスが実現できる．

4. 行動履歴の蓄積

特定の場所に来たときに，自分のデータベースの内容を更新するようなルールを用いることで，スタンプラリーやオリエンテーリングなど，さまざまな場所をまわってアイテムを集めてまわるようなアミューズメントサービスが実現できる．また，通行証のようなデータを特定の場所を通るたびに蓄積していけば，自分のたどった道筋をあとから知ることができる．

5. 情報の共有

ローカルデータベースが更新されたとき，サーバに更新内容を送信するようなルールを用いることで，情報の共有が可能になる．例えば，ローカルデータベースには自分だけが利用する建物情報や経路情報を自由に追加し．ローカルデータベースの更新に対して，更新データを移動体サーバに送信するようなルールを用いることで，更新を移動体サーバに伝播し，複数の人で経路などの情報が共有できる．

6. 動的なシステム拡張

ユーザはあらかじめECAルールを格納しておかなくても，移動体サーバから送られるECAルールを格納することで，即座にアプリケーションを更新できる．このよう

な機構を用いて，施設の入口で必要なECAルールを受け取ることで，遊園地では遊園地の，水族館では水族館の提供する独自のアプリケーションが利用できる．

実際のECA-MLの例として，次のような機能をActiveGISにもたせることを考える．

1. 本屋専用の地図を作成する．
2. 自分から一定範囲内に指定した種類のオブジェクトを見つけたとき，そのオブジェクトの色を変化させて目立たせ，さらにユーザに通知する．
3. 自分の一定範囲内に本屋があったとき，自分の探している本があるかどうかを本屋のデータベースから検索して結果を表示する．

これらの機能を実現するためのルール例は図3.3，3.4，3.5に示すとおりである．記述例1は，G-XMLデータを受信したときに発火するルールで，アクション部で実行されるXSLTにより，本屋に関するオブジェクトのみを抽出して処理するECA-MLである．記述例2は，オブジェクトが接近したときに発火するルールで，接近が30メートル以内で，さらにそのオブジェクトが本屋であった場合，XSLTにより対象オブジェクトの前景色を黄色に変更し，DISPLAYアクションによってユーザに通知する．ルール例3は，本屋の建物データを受け取ったときに，自分のデータベースからほしい本の情報を検索して本屋に送信し(`rule3-1`)，本屋においてデータベースを検索して本の値段を返し(`rule3-2`)，受け取った値段情報をユーザに通知する(`rule3-3`)ルール群である．

このように，ECA-MLを用いることでさまざまなサービスを提供する地理情報システムが構築できる．

3.3 システムの実装

ActiveGISのシステム構成を図3.6に示す．AMDSモジュールは，2章で述べたAMDSのコア部分を汎用的に利用できるようにモジュール化したものであり，ルール処理やデータの送受信などの処理を行う．地図表示部では，GPSから現在地の情報を受け取り，自分の位置を中心とした地図を表示する．情報表示部では，地図上のオブジェクトがポイントされたときにその情報を表示するとともに，ルールのDISPLAYアクションによって指定された内容を画面に表示する役割をもつ．ユーザインタフェース部では，各種パラメータを入力させたり，地図の拡大縮小など，ユーザからの入力を受け取って処理する．

```
<ECARULE id="rule1" scope="entire">
 <EVENT type="RECEIVE"/>
 <ACTION>
  <ITEM type="XSLT" ret="tempGXML">
   <xsl:template match="/RW-GXML">
    <RW-GXML>
     <xsl:for-each
      select="Feature[@theme='本屋']">
      <xsl:apply-templates
       select="*|@*|comment()|pi()|text()"/>
     </xsl:for-each>
    </RW-GXML>
   </xsl:template>
  </ITEM>
  <ITEM type="XMLProc">
  %s<VARIABLE>%s=tempGXML</VARIABLE>
  </ITEM>
 </ACTION>
</ECARULE>
```

図 3.3: ECA ルール例 1

```
<ECARULE id="rule2" scope="entire">
 <EVENT type="CLOSE"/>
 <CONDITION>
  <AND>
   <ITEM type="LESS" >NEW.DISTANCE, 30</ITEM>
   <ITEM type="EQUAL" >NEW.TYPE, 本屋</ITEM>
  </AND>
 </CONDITION>
 <ACTION>
  <ITEM type="XSLT">
   <xsl:template match="/">
    <xsl:apply-templates
     select="*|@*|comment()|pi()|text()"/>
   </xsl:template>
   <xsl:template match="PolygonStyle[@pffgcolor]">
    <PolygonStyle plcolor="{@plcolor}"
     pffgcolor="YELLOW" plwidth="{@plwidth}" />
   </xsl:template>
  </ITEM>
  <ITEM type="DISPLAY">
   Normal, "近くに%s があります. "
   <VARIABLE>%s=NEW.TYPE</VARIABLE>
  </ITEM>
 </ACTION>
</ECARULE>
```

図 3.4: ECA ルール例 2

```
<ECARULE id="rule3-1" scope="entire">
  <EVENT type="CLOSE"/>
  <CONDITION>
    <AND>
      <ITEM type="<">NEW.DISTANCE, 30</ITEM>
      <ITEM type="=">NEW.TYPE, "本屋"</ITEM>
    </AND>
  </CONDITION>
  <ACTION>
    <ITEM type="QUERY" ret="QDATA">
      "SELECT BookName FROM RequestBookTBL"
    </ITEM>
    <ITEM type="SEND">
      %s1, "BookRequest", "BookName:%s2"
      <VARIABLE>%s1=NEW.FROM, %s2=QDATA</VARIABLE>
    </ITEM>
  </ACTION>
</ECARULE>

<ECARULE id="rule3-2" scope="entire">
  <EVENT type="RECEIVE"/>
  <CONDITION>
    <ITEM type="=">NEW.HEADER, BookRequest</ITEM>
  </CONDITION>
  <ACTION>
    <ITEM type="QUERY" ret="QDATA">
      "SELECT Price FROM BookTBL
       WHERE BookName=%s"
      <VARIABLE>%s=NEW.BookName</VARIABLE>
    </ITEM>
    <ITEM type="SEND">
      %s1, "Result", "PRICE:%s2"
      <VARIABLE>%s1=NEW.FROM, %s2=QDATA</VARIABLE>
    </ITEM>
  </ACTION>
</ECARULE>

<ECARULE id="rule3-3" scope="entire">
  <EVENT type="RECEIVE"/>
  <CONDITION>
    <ITEM type="=">NEW.HEADER, Result</ITEM>
  </CONDITION>
  <ACTION>
    <ITEM type="DISPLAY">
      MsgBox, "値段:%s"
      <VARIABLE>%s=NEW.PRICE</VARIABLE>
    </ITEM>
  </ACTION>
</ECARULE>
```

図 3.5: ECA ルール例 3

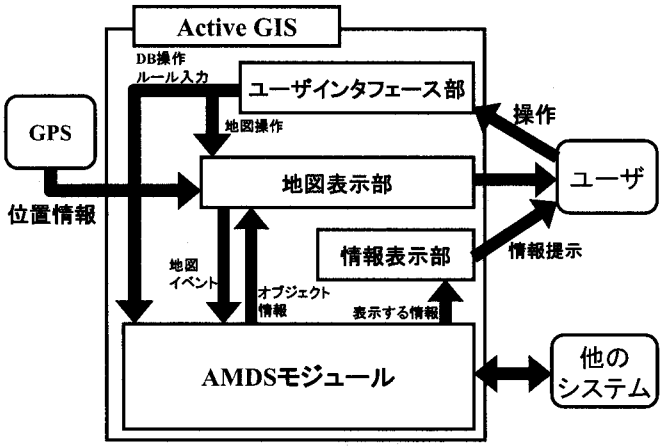


図 3.6: ActiveGISのシステム構成

実装したActiveGISのプロトタイプシステムの稼動図を図3.7に示す．実装はWindowsPC上で，Visual Basic6.0およびVisual C++6.0を用いて行い，無線通信には赤外線通信および無線LANを用いた．図左のメイン画面では，自分の位置を基点とした地図が表示されており，無線通信により受信したG-XMLのオブジェクトが重ねあわされて表示されている．図は前節のルール例3を実行し，本屋のそばを通りかかったときにユーザが欲しいと思っている本の値段情報を自動的に提示している例である．

3.4 考察

G-XMLに対するECA-MLの有効性について

G-XMLを移動体環境上のアプリケーションで用いる場合，以下のような問題点が起こる場合がある．

1. G-XMLでは地理情報を柔軟に表現するために多くの要素および属性が定義されているが，G-XML記述者は自分にとって必要なタグしか記述しない．一方，受信者側のアプリケーションでは，システムの処理の都合上記述されていないタグも必要となる．

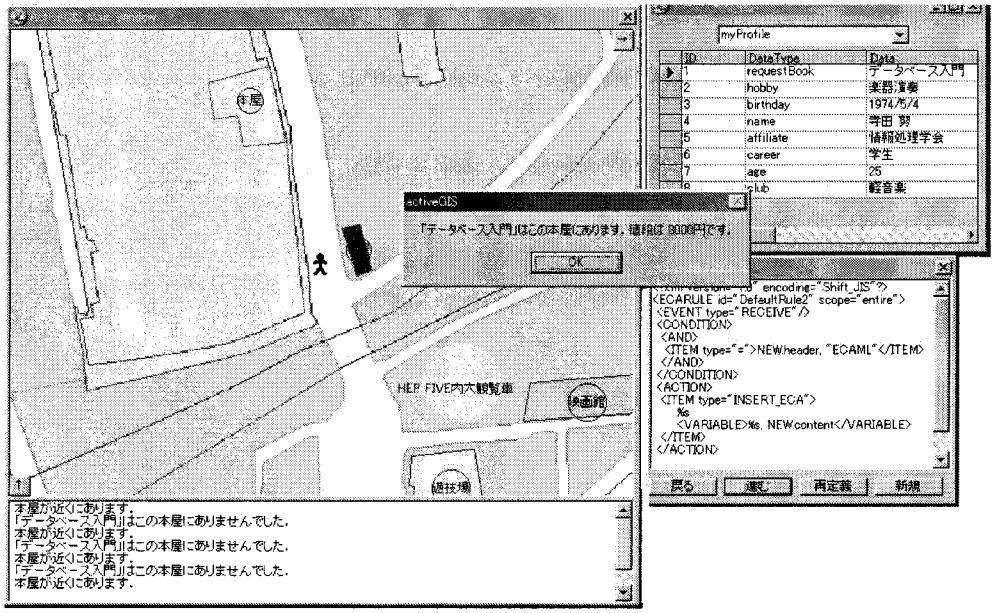


図 3.7: プロトタイプ稼動図

2. G-XML 記述者は，ローカル情報や期限付きの情報をやり取りしたい場合がある．しかし，期限付きの情報は期限が来たら削除したいといったように，データを送信する側がそのデータの処理を記述したい場合がある．

1番目の問題点を解決するためには，適切なタイミングで，受信したG-XMLを操作してタグの補完を行なう必要がある．ECA-MLでは，アクションにXSLを利用できるため，柔軟なタグ操作が可能となる．また，多数のイベント記述を提供しているため，さまざまなタイミングで各種操作を実行できる．

2番目の問題点に関しては，G-XML文書に処理を規定するルールを含めることで解決できる．ECA-MLはXML形式で記述されるため，G-XMLインスタンス中に直接ECA-MLを記述し，さらにECA-MLのSCOPE属性をローカル文書内のみに指定しておくことで，自分が書いた文書のみに影響するルールを記述できる．G-XML文書内にECA-MLを含めて配信する例を図3.8に示す．図に示すG-XML文書は，工事現場情報を表すG-XML文書であるが，2003年7月30日に発火するタイマをセットするルール(SetDeleteTimer)と，タイマが発火したときに工事現場のG-XMLオブジェクトを削除するルール(DeleteObject)を文書中に含めているため，工事の期限が来ると自動的に削除する機能を含めた地理オブ

```
<RW-GXML version="1.0A">
<Feature Fid="3" Thame="工事現場">
  <Polygon>
    <Coordinate>15300000,16350000</Coordinate>
    <Coordinate>15380000,16350000</Coordinate>
    <Coordinate>15380000,16400000</Coordinate>
    <Coordinate>15300000,16400000</Coordinate>
  </Polygon>
</Feature>
<ECARULE id="SetDeleteTimer">
 <EVENT type="INSERT" target="GXML" />
 <ACTION>
  <ITEM type="SET_TIMER">DelTimer, ABS(200307301700) </ITEM>
 </ACTION>
</ECARULE>
<ECARULE id="DeleteObject" scope="internal">
 <EVENT type="TIMER">
 <CONDITION>
  <ITEM type="EQUAL">NEW.ID, DelTimer</ITEM>
 </CONDITION>
 <ACTION>
  <ITEM type="QUERY">
   "DELETE X FROM GXML WHERE X.Thame='工事現場'
  </ITEM>
 </ACTION>
</ECARULE>
</RW-GXML>
```

図 3.8: G-XML 文書に ECA-ML を含めた例

ジェクト配信が実現できている.

以上のように，ECA-ML は G-XML を運用する上で起こる問題点に対処できるため，G-XML の処理言語として有効であるといえる.

ECA ルールの言語仕様について

ActiveGIS で用いる ECA ルールにおいて記述できるイベントおよびアクションは表 3.1, 3.2に示したが，地図上のオブジェクトに対してのイベントは接近と離脱しか用意していない．現時点においても，3.2 節で述べたようなさまざまなサービスを提供できるが，地理情報システムのアプリケーションには現状の ECA ルールでは実現できないサービスもある．たとえば，ネットワーク分析やオーバレイの制御を ECA ルールで行なうことができず，定

期的な処理や災害調査などにおいては，柔軟なタイマ処理を記述できる必要がある．したがって，新たなイベントおよびアクションの追加を含め，言語仕様を再考することが今後の課題となる．

実現可能性について

本システムを実環境で動作させるためには，街中に移動体サーバを設置する必要がある．また，すべての移動端末と移動体サーバにActiveGISを搭載する必要がある．移動体サーバの設置に関しては，既存のPHSや携帯電話の基地局にActiveGISのシステムを組み込むことで実現できる可能性がある．また，移動体サーバはネットワークにつながっていなくても位置依存サービスを提供できるため，各店舗が無線通信機能を備えた移動体サーバを独自に設置することで，すぐにサービスが開始できる．移動体サーバは一般のパソコンにActiveGISを組み込んだものでよいので，コスト面でも充分実現可能である．移動体に関しては，携帯端末にActiveGISを組み込んでおけばよい．PDAや携帯電話などディスク容量やCPUパワーに制限がある場合は，ECAルールの手入力による追加機能といった互換性に問題のない部分を削除するなど，機能を絞って搭載することになる．

位置の特定方法について

現在のシステムでは位置を特定する方法としてGPS機器を用いている．近年，GPS精度の向上により誤差は数メートル程度に抑えられているため，位置の特定に関してはGPS機器で充分であると考えられるが，GPSと他のデバイスとを組み合わせることで，より柔軟な位置管理を行える．たとえば，移動体サーバから移動体に対して，今どのセルに入っているかという情報を送信することで，どの属している移動体サーバ情報を用いたサービスを実現できる．

移動体サーバの設置間隔について

本システムでは移動体サーバの規模や設置間隔については規定していない．すべてのサーバが対等に存在し，一定間隔ごとに移動体サーバを設置する方法の他に，セル範囲が広く，基本的なオブジェクトすべてを管理する移動体サーバを広い間隔で設置し，詳細情報や位置依存サービスを提供するサーバを特定の位置ごとに設置するような階層型の設置方法も考えられる．階層型にすることで，大まかな情報は遠くからでも取得でき，位置依存の情

報は特定サーバの近くに行かないと得られないようにできる．

関連研究

本研究と同様に，GISにアクティブデータベースの機構を取り入れたものに，GISユーザインタフェースの動的カスタマイズに関する研究[44]がある．この研究は，ユーザインタフェースをECAルールを用いて動的に変化させる研究[9][43]をGISに適用したものである．Oliveiraら[44]は，ActiveGISと同様，GISにアクティブルールのメカニズムを盛り込んでいるが，ECAルールはユーザインタフェースをカスタマイズするために用いている．データ内容によってインタフェースを変化させることで，ユーザにとってわかりやすいデータ処理が行えるが，追加されているイベントやアクションも，ユーザインタフェース構築用のものであり，アプリケーションの機能を制御することはできない．一方，ActiveGISでは，ユーザインタフェースをルールのみで制御することはできないが，アプリケーションの機能そのものがECAルールで表現されるように拡張を行なっているため，ECAルールを追加したりカスタマイズすることで，アプリケーション自体をカスタマイズできる．

モバイル環境でGISを利用するための研究としては伊藤らのシステム[25]が挙げられる．文献で述べられているモバイルGISに求められる機能としては，図形情報の入力や更新，データの整理や分析，出力などが挙げられているが，その目的は現地調査における各種のデータ入力をその場で行なうことである．一方，ActiveGISは，ECAルールによるサーバとのインタラクションを用いて位置依存サービスを受けることを主な目的としている．

位置依存サービスの研究例としては，SpaceTag[56]や，モーバイルインフォサーチ[58]が挙げられる．SpaceTagは，時空間上にオブジェクトを配置し，その場，その時間にいないとオブジェクトにアクセスできないように制限を加えることで位置依存サービスを実現する機構である．SpaceTagは，それ自体が時空間上の場所にあるのではなく，サーバにまとめて蓄積されている位置情報をもったオブジェクトである．したがって，端末はデータにアクセスする際には，ネットワーク上に存在するサーバにアクセスするための機能をもつ必要がある．また，悪意をもったユーザによって，離れた場所からネットワーク越しにデータを見られてしまう可能性がある．一方，ActiveGISにおける移動体サーバは，自分の周りにある情報を保持し，それを無線によって送信しているため，近くにいなければ受信できない．したがって，より強固な位置依存サービスを提供できるが，移動体サーバが各所に存在しているという前提が必要となる．モーバイルインフォサーチでは，モバイルユー

ザの現在位置に対して，時刻表や最寄の店などの情報を提供することで，実世界とネットワークの両方から情報を得ることができるようにしたシステムである．移動体上でデータ収集を行なう点でActiveGISとの共通点があるが，モーバイルインフォサーチでは，データ収集をエージェントによって実現しようとしているのに対し，ActiveGISではECAルールによってデータ収集を行なっている．

3.5 むすび

本章では地理情報システムのためのアクティブデータベースシステムであるActiveGISについて述べた．ECAルールをXML形式で表現したECA-MLを用いることで，地理情報の受信時にオブジェクト情報を加工してユーザに提示したり，地理情報中にルールを埋め込んで送信者側から地理情報の取り扱いを制御するといった機能を実現している．ActiveGISを用いることで，さまざまな位置依存サービスを提供することが可能となる．今後の課題としては，考察でも挙げたように言語仕様の見直しが挙げられる．地理情報システムで用いられるオーバーレイ等の機能をECAルールを用いて実現するため，いくつかのイベントやアクションを追加する予定である．また，G-XML以外の地理情報記述言語を取り扱えるように拡張することで，複数の言語に対応した汎用的な地理情報システムを構築する予定である．

第4章

放送環境における高速データ処理を実現するアクティブデータベース

4.1 まえがき

近年，放送衛星や通信衛星の新たな打ち上げ，デジタル放送の開始などにより，衛星を用いた放送型システムに対する注目が高まっている．広帯域を利用することで放送されるコンテンツの種類，量ともに大幅な増加が期待されている．また，デジタル放送を利用することでコンピュータで用いるデータなどのデジタル情報を放送する新たなサービスも多数提供されるようになる．そのため，受信側にはさまざまな種類のデータが大量に放送されることになり，ユーザがこの大量の受信データの中から必要な情報を見つけ出し，効率的に利用することは困難となる．

これまで，放送型システムに関する研究はサーバ側のデータ放送戦略に関して多く行われている[1][3][13][19][75]．しかし，大量のデータが放送される環境においては，データ放送戦略と共に，いったんデータを蓄積し，必要なデータを再利用することの重要性が非常に高い．また，データ放送においては，コンピュータの実行プログラムの様に最初から蓄積を前提としたデータが放送される場合が多い．

このように，放送型データ受信システムを構築する場合，受信データの種類に応じて柔軟にフィルタリング・蓄積・管理を行う機能が必要であるが，従来の放送型システムでは単一の機能に特化されたものが多く，新しいタイプのデータへ対応したり異なるフィルタリングアルゴリズムを適用するといった柔軟な処理が行えなかった．また，衛星放送を用

いたデータ放送などの広帯域を利用した放送型システムでは，短時間に大量のデータが到着するため，フィルタリング処理を高速に行う必要がある．

そこで，本章ではアクティブデータベースを放送型システムに適応させることで，大量の放送型データを効率よく受信・蓄積し，ユーザが蓄積された情報に手軽にアクセスできるシステムを提案する．提案するアクティブデータベースであるSADB(Super Active DataBase system)は，ECAルールを用いてデータのフィルタリング，蓄積・管理，問合せ処理等を実現しており，多種のデータを受信可能な放送型データ受信基盤として利用できる．また，受信データを高速に処理するため，受信動作記述や受信データ処理部分をルール処理部から独立に設計して高速化を行っている．以下，4.2節ではシステムの設計について述べ，4.3節では高速化手法の評価を行う．4.4節ではいくつかの応用システムについて述べ，最後に4.5節で本章のまとめを行う．

4.2 システムの設計

4.2.1 想定環境

SADBへの要求仕様を明らかにするため，次のような環境を想定する．SADBはこのような環境において放送データを高速に受信・処理するシステムとして設計する．

- 伝送速度: 現在のCSデジタル放送と同様に1Mbps程度とする．
- 帯域幅の非平衡: 上り回線は存在しないか，存在しても電話回線などの狭い帯域である．
- データ形式: 放送データの形式は，PC用のデータやHTMLデータなどであり，一般の番組を放送する場合も，MPEGファイルとして送られるとする．また，それぞれのデータアイテムにはキーワードなどのメタデータが付加されているとする．
- データサイズ: 現在の地上波データ放送では，一般に1つのデータアイテム(HTMLデータや画像データなど)は，数10kバイト程度である．本研究においても1アイテムのデータサイズは数10kバイト程度と想定するが，番組を動画ファイルとして送る場合のようにファイルサイズの上限は数100Mバイトまで大きくなる可能性があるとする．

表 4.1: SADBのイベント

名称	内容
SELECT	テーブルに対するデータ参照
INSERT	テーブルに対するタップルの挿入
DELETE	テーブルのタップル削除
UPDATE	テーブルのタップル更新
RECEIVE	データパケットの受信
TIMER	設定したタイマの発火

表 4.2: NEW/OLDデータの内容

イベント	NEW	OLD
SELECT	参照タップル	–
INSERT	挿入タップル	–
DELETE	–	削除タップル
UPDATE	更新後タップル	更新前タップル
RECEIVE	到着パケット内容	–
TIMER	タイマ識別子	–

表 4.3: SADBのアクション

名称	内容
QUERY([クエリー内容])	データベース操作
SEND([宛先], [送信内容])	データの送信
INSERT_ECA([ルール内容])	ECAルール格納
DELETE_ECA([ルール識別子])	ECAルール削除
ENABLE_ECA([抽出条件])	ECAルール有効化
DISABLE_ECA([抽出条件])	ECAルール無効化
GROUPING_ECA([条件])	ルールグループ化
ADDTO_GROUP([条件])	グループにルール追加
RELEASE_GROUP([グループ名])	グループ化解除
SET_TIMER([タイマ条件])	新たなタイマの設定
KILL_TIMER([タイマ識別子])	タイマの削除
VBSCRIPT([スクリプト])	スクリプト実行

- データ到着頻度: 伝送速度を1Mbps, データサイズを20kバイトから100Mバイトとした場合，データの到着頻度は1秒あたり1/1000回程度から7回程度になる．
- データの格納率: データの格納率はユーザが閲覧可能なデータ量に依存する．放送データがHTMLデータであった場合，ユーザが1アイテムを閲覧するのに1分かかったとすると，格納率は400分の1程度となる．実際は常にユーザがアプリケーションを利用しつづけるわけではないので，平均すると格納率はさらに下がると考えられる．

4.2.2 SADBのECAルール

SADBはデータ受信イベントを処理するECAルールのコンディション記述を用いて受信データのフィルタリングを行い，アクションによって必要なデータを格納する．SADBにおけるECAルールの記述構文は2章で述べたAMDSの記述構文と同様である．SADBで利用することができるイベント，アクションを表4.1，4.3に，NEWデータおよびOLDデータを表4.2に示す．

ECAルールの記述例を図4.1，4.2に示す．ルール1は，データを受信したときに，その内容が'HotTopics'であればデータの格納を行う，というルールである．このようなルール群によりフィルタリングが実現される．ルール2は有料データに対する課金を行うための

ルールであり，データが参照されたときに，そのデータが有料であればユーザに対して課金情報を書き込む．ルール3は機能追加のためのルールであり，ECAルールが放送されてきたときに，その放送元が'Nishio-lab'であればそのECAルールを信頼してシステムに格納するルールである．ルール4は，データの新鮮さを管理するルールであり，一定時間ごとにデータの新鮮さを減少させ，古くなったデータは自動的に削除する．このように，ECAルールを用いたキャッシュ管理を行うことで，データ内容やユーザの興味を考慮したキャッシュ管理を行える．

また，図4.2に示す3つのルールは，連鎖的に実行させることで他のサーバへの情報要求機能を実現している．クライアント上でユーザがデータを参照したとき，クライアントルール1が発火し，ユーザの参照したデータに関する詳細情報をサーバに要求する．要求パケットによりサーバルール1が自動的に発火し，詳細情報を検索して結果パケットを返す．結果パケットにより，クライアントルール2が発火し，受信した詳細情報を表示する．

4.2.3　システムの構成

SADBのシステム構成を図4.3に示す．データ受信部では，放送されるデータを受信してデータ受信イベントを発生させる．DB管理部では，データベースに対する要求を受け，実際にデータベースの操作を行い，データベース操作イベントを発生させる．ルール処理部では，システム上に起こるイベントを検出し，ECAルールの検索，コンディションの評価を行い，アクションを実行する．アプリケーションインタフェース部では，オンラインでつながる他のSADBやアプリケーションとの通信を行う．ユーザインタフェース部では，ユーザにグラフィカルなインタフェースを提供し，ECAルールの入力や，データベース操作などを容易に行える環境を提供する．

SADBのルール処理

SADB上でECAルールが入力されたり，ECAルールを受信すると，そのルールは，高速化のため中間形式に変換されたのちにECAルールデータベースに格納される．さらに格納ECAルールのクラスタリングを行うことで高速なECAルール検索を実現する．何らかのイベントが発生した場合，ECAルールデータベースを検索し，ヒットするルール群を抽出してコンディションの評価，実行を行う．

しかし，このようなルール処理機構では，衛星放送を想定した場合，ルール評価のプロ

```
create rule ルール1 on RECEIVE
  where new.header = 'HotTopics'
  then do
   QUERY( "INSERT INTO DataStore
         VALUES( new.id,new.data )" );

create rule ルール2 on SELECT
  where new.price > 0
  then do
   QUERY( "UPDATE PayInfo SET ACOUNTING
          = ACOUNTING + %s", new.price );

create rule ルール3 on RECEIVE
  where new.header = 'ECA_RULE'
        new.from = 'Nishio-lab'
  then do
    INSERT_ECA( new.eca_difinition );

create rule ルール4 on TIMER
  where new.name = 'UPDATE_TIMER'
  then do
    QUERY( "UPDATE DataStore SET Verdure
           = Verdure - 1" );
    QUERY( "DELETE * FROM DataStore
           WHERE Verdure = 0" );
```

図 4.1: ECA ルール例 1

```
create rule クライアントルール1 on SELECT
  then do
   SEND( 'DataServer', 'RequestDetail',
         'name:', new.name );

create rule サーバルール1 on RECEIVE
  String ResponseData
  where new.header = 'RequestDetail'
  then do
   ResponseData = QUERY( "SELECT FROM DetailData
         WHERE Name=%s", new.name );
   SEND( new.from, 'ResponseDetail', 'data:' ,
         ResponseData );

create rule クライアントルール2 on RECEIVE
  where new.header = 'ResponseDetail'
  then do
    VBSCRIPT( "MSGBOX %s", new.data );
```

図 4.2: ECA ルール例 2

セスがボトルネックとなり，実用に耐え得る処理速度が得られない可能性がある．3.2.1 節で示した想定環境においては 1 秒間に 7 回程度データを受信することになるが，データ受信ごとにイベントが発生するため，それぞれのイベントに対していちいちルールデータベースを検索し，マッチするイベントを探すことはコストの高い処理である．

そこで，SADB に放送に特化した処理モード (高速モード) を実装した．一般に，アクティブデータベースでは ECA ルールの最適化や実行順序の制御により高速化を行っている [21][34]．しかし，SADB は放送に特化したアクティブデータベースであるため，イベントの発生頻度に注目した高速化を行う．想定環境では，受信イベントが頻繁に，また継続的に発生するがデータ格納イベントはほとんど発生しない．また，検索イベント等のユーザやアプリケーションによって起こるイベントも，データ受信イベントに比べて非常に少ない頻度でしか発生しない．

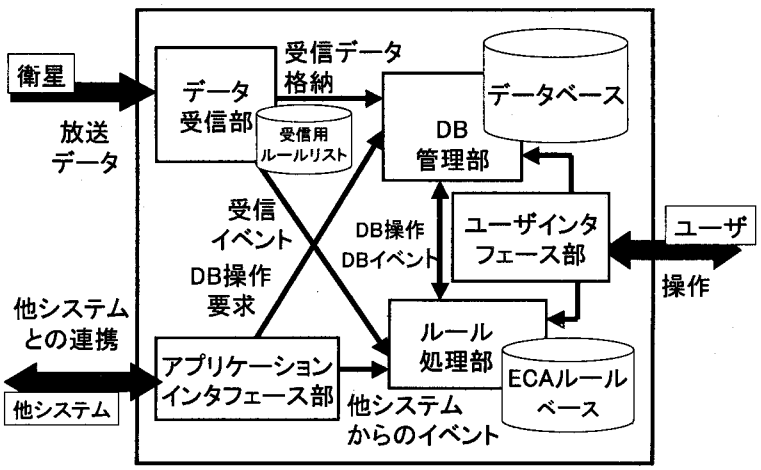


図 4.3: SADBのシステム構成

以上から，データ受信イベントの処理時間がシステムのパフォーマンスのボトルネックとなる．そこで，高速モードでは，受信イベントの一次フィルタリングをデータ受信時に行い，ボトルネックの解消をはかる．

高速モードのルール処理の流れを図4.4，4.5に示す．データ受信イベントを処理するECAルールは一般のECAルールデータベースには格納されず，中間言語に変換されてデータ受信部がもつルールリストに登録する．ルールリストの格納形式は，ECAルールデータベースに格納する情報からイベント情報を省いたものであり，ルールリストを利用すること自体にはそれほど高速化の期待はできない．そこで，さらにそのコンディションが単純なヘッダ比較であった場合，そのルールをヘッダ比較リストに登録する．一般に，データ受信時にはヘッダ比較によってフィルタリングを行うため，このタイプのECAルールがほとんどとなる．あるデータが到着したとき，データ受信部ではまずヘッダ比較リストを参照して適合するものが無いか調べる．その後自分が持つルールリストを検索する．データが不要と判断された場合，ルール処理部にイベントを通知せず，次のデータ処理に移る．こうすることで，イベントとしてルール処理部にデータを渡し，ECAルールデータベースを用いて処理を行う手間が省け，効率のよい処理が実現できる．また，1次フィルタリングにマッチせず，ルールリストを用いて処理を行う場合も，1次フィルタリングの結果を用いてコンディション判定を行うため，従来の手法に比べて多少の高速化が期待できる．

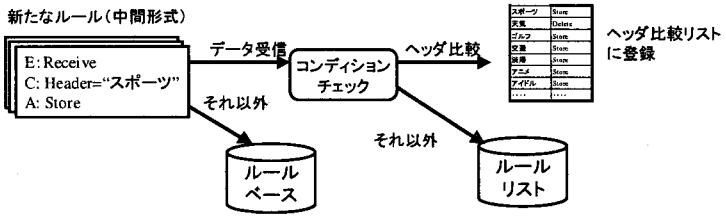


図 4.4: ルール登録時の処理

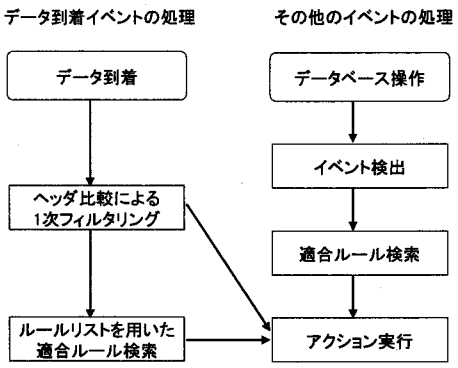


図 4.5: 高速モードのルール処理

4.3 性能評価

前節で述べた高速ルール処理機構を実装し，性能を評価した．評価は，ECA ルールにより放送データを受信・蓄積・閲覧するシステムを Windows95 搭載の携帯端末 (CPU: Mobile PentiumII 266Mhz，Memory: 128M バイト) 上で実際に動作させて行った．放送サーバから HTML データおよびイメージデータを繰り返し放送し，通常のルール処理時間と高速モードのルール処理時間を測定した．結果を図 4.6，4.7，4.8に示す．放送されるデータアイテムはそれぞれ一つ以上の HTML データおよび画像データ (各データのサイズは数 Kb から数 10Kb) を含む．また，データの格納率は 1/400 程度になるように放送データを調整し，放送サーバのデータ放送速度は 1Mbps 程度になるようにした．さらに平均 1 秒に 1 回程度，データベース参照などを強制的に行うことで，データ受信以外のイベントを発生させた．図 4.6，4.8の評価に使用した ECA ルールは，システムの動作に必要なルールを除いてランダムに生成したものを用いるが，実際にシステムを運用した際の比率を参考に，データ受信を処理するルールとそれ以外のイベントを処理するルールを 7:3 の割合で生成するようにした．また，図 4.6，4.7における受信ルールに関しては，単純なヘッダ比較を用いたルールと，そうでないルールの発生比率は，平均 1:1 になるようにした．「そうでないルール」とは，単純なヘッダ比較にシステム変数の参照による条件などを加えたもので，計算量はそれほど増えないが，ルールリストを用いた処理を行う必要があるルールである．すべてのグラフにおいて，30 イベントあたりの処理時間を縦軸に示す．

図 4.6は，システム内の ECA ルール数を 10 から 50 まで変化させたときの処理時間の変化を示したものである．ECA ルールを用いたアプリケーションでは，通常 10 から 20 程度の

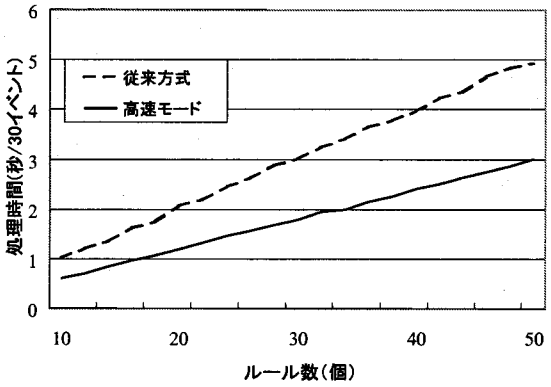


図 4.6: ルール数による性能の変化

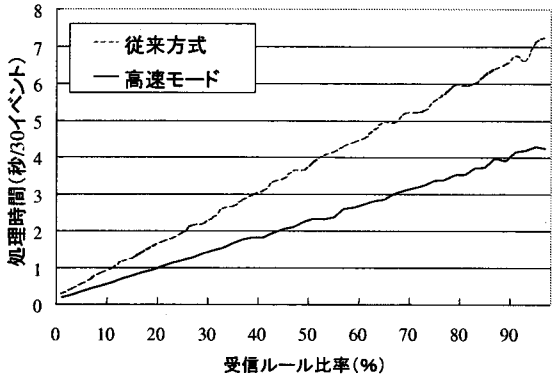


図 4.7: 受信ルールの比率による性能の変化

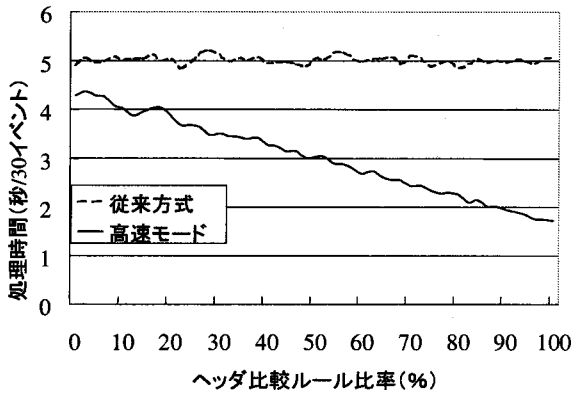


図 4.8: ルールの複雑さによる性能の変化

ルール数でシステムが動作しているが，フィルタリングの複雑さに応じてフィルタリングのためのECAルール数も増加する．グラフより，高速モードではルール数の増加に対して処理時間の増加が抑えられていることがわかる．

図4.7は，ルール数を50に固定し，受信イベントを扱うルールとそれ以外のデータを扱うルールとの比率を変化させたときの処理時間を示したものである．発生するイベントのほとんどがデータ受信ルールであるため，両手法ともデータ受信ルールの割合が増加すると，発火するルールが増えて処理時間が増加している．受信ルールがないときは両方式とも同じ処理を行うため，ほとんど差が現われない．しかし，受信ルールの比率が多くなるにつれて，本手法の効率の違いが見られる．一般に，放送環境ではルールが多くなると受信ルール比率が多くなる．これは，フィルタリング条件を記述するECAルールが多くなるためである．以上のことから，本手法は従来手法に比べて放送環境で有効であるといえる．

図4.8は，ルールをすべて受信に関するルールに限定した場合において，単純なヘッダ比較ルールと，そうでないルールとの比率を変化させたときの処理時間を示したものである．3.2節で述べたように，ヘッダ比較ルールはヘッダ比較リストに登録して処理するため高速化が可能である．特に，受信ルールがすべてヘッダ比較であった場合は，従来手法に比べて大幅に効率がよくなる．

想定環境では，システムに最低限必要なECAルールの総数は少なく，後からフィルタリング用のルールを追加していくといった使い方が一般的である．また，フィルタリング用のルールの多くは,「スポーツのデータが欲しい」「競馬のデータはいらない」といったように，単純な比較で済むものが多い．したがって，提案手法は評価したあらゆる面において，想定する環境にふさわしい手法であると言える．さらに，すべての場合において，最悪の場合でも提案手法が従来手法よりも高速であることがわかる．しかし，このようにルール処理を二つのモジュールに分けて行うことで，ルールの実行順序が保証できなくなる．このことが問題となるようなアプリケーションでは，従来手法を用いる必要がある．

4.4 応用システム

SADBを用いることで，放送されるデータを柔軟に取り扱えるようになった．そこで本節では，実際にルール機構を使用したシステム構築例として，3つの放送型データ受信システムについて述べる．まず，情報源統合および放送データ閲覧のための放送型データ受信システムアクティブ情報ストア(Active Information Store: AIS)について述べ，次にフィルタリング関数の枠組みを利用した処理方法最適化機構をもつ情報フィルタリングシステムについて述べる．最後に，放送型データベースシステムに対する問合せ処理機構をもつ放送型データ受信システムについて述べる．

4.4.1 放送データ受信システムアクティブ情報ストア

アクティブ情報ストア(AIS)は，データ受信基盤にSADB，フィルタリングアルゴリズムに木構造をもつ情報フィルタリング機構を用いることで放送型データを受信・格納する．また，ユーザに対しては木構造型のインタフェースを用いて容易なデータアクセスを提供する．

現在行われている放送型サービスを受信するためには，専用のアプリケーションが必要

であり，データを閲覧するためにはそれぞれの閲覧用ソフトを利用する必要があった．そこで，AISでは，さまざまな情報ソースから放送されるデータをSADBによって統合し，シームレスに扱えるようにした．AISを用いることで，情報源の違いを意識することなくデータにアクセスできる．

情報フィルタリングアルゴリズム

AISはユーザインタフェースとして木構造を持つビューを用いる．木の各節点はデータ分類の概念をあらわし，葉に向かうにつれてより詳細な概念になる．たとえばある節点‘ニュース’があるとすると，その葉節点として‘政治’や‘スポーツ’が考えられる．

ある放送データを受信したとき，そのデータは1つまたは複数の節点に分類される．そのデータが実際に格納されるかどうかは，その節点の価値や更新頻度，そのジャンルに対するユーザの興味などを総合的に判断して決定される[47]．データの分類，格納に利用される木の構造は，木の偏りやアクセスの変化などの要因に応じて動的に再構築されるためユーザに特化したデータの分類ができ，ユーザは常に木をたどる操作のみで必要な情報にアクセスできる．

システムの構成

AISのシステム構成を図4.9に示す．まず，放送データをSADBによって受信し，データ内容やサイズ等の情報を情報フィルタリング部に通知する．情報フィルタリング部においては，そのデータの評価値を計算し，情報の必要性を判断する．ユーザが情報を必要とするときは，ユーザが木構造の節点においてデータを指定すると，フィルタリング部がSADBに情報を要求し，アクティブデータベース部が必要とされた情報を返し，それをユーザに提示する．また，必要か不要かが確実にわかっているデータに関するフィルタリング情報はECAルールとして記述され，SADBに格納される．時間の流れにより情報の価値が変化するデータに関しては自動的に評価の値を更新し，期限付きデータの期限が切れた場合にはデータを削除する．

実装

アクティブ情報ストアのプロトタイプを実装した．また，放送サーバとしては，実際に衛星を利用した放送型データを受信するのではなく，(1) 地上波データ放送サービスであ

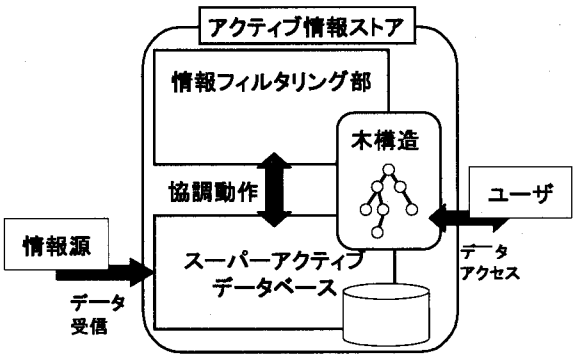


図 4.9: アクティブ情報ストアの構成

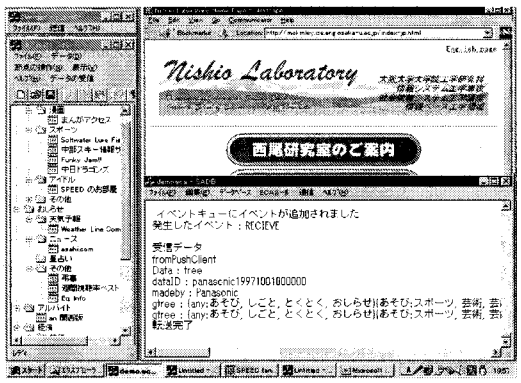


図 4.10: システム稼働図

るADAMS(朝日放送)の放送データ，(2)インターネットを利用したデータ放送サービスであるPOINTCASTの放送データ，(3)独自に開発したロボットを用いて収集したインターネット上のホームページデータの3種のデータを放送するプログラムを開発した．システムの実行例を図4.10に示す．放送されるデータはHTML文書，画像ファイル，音声ファイルなど多岐にわたる．実際に放送サーバが放送するデータ量は，ADAMS, POINTCAST, ロボットによる収集データを合わせて一日あたり1000アイテム，8Mバイト程度であった．これらのデータを受信し，情報フィルタリング部においてジャンルごとに分類し，ユーザに提示する．ユーザインタフェース上では，どの情報源からのデータも同様に扱われており，ユーザは放送元の違いを意識することはなく情報を利用できる．ユーザがコンテンツを選択したときは，選択したコンテンツをブラウザ上に表示する．

システムの機能は，データ受信処理，ルール受信処理，データ表示処理などを行う10個のECAルールにより実現されている．1週間程度利用した後，システム内のECAルール数は18個に増加していた．これは，フィルタリングのためのルールや，送信側がデータの自動削除を行うために送信してきたルールがシステムに格納されたためである．放送サーバでは，天気予報など有効期限をもつデータを送信する際にはそのデータを自動的に削除するためのルールもあわせて放送している．これらのルールは一日に数個が放送されているが，ルール自体の有効期限がくれば削除されるため，システム内のルール数をそれほど増加させることはない．

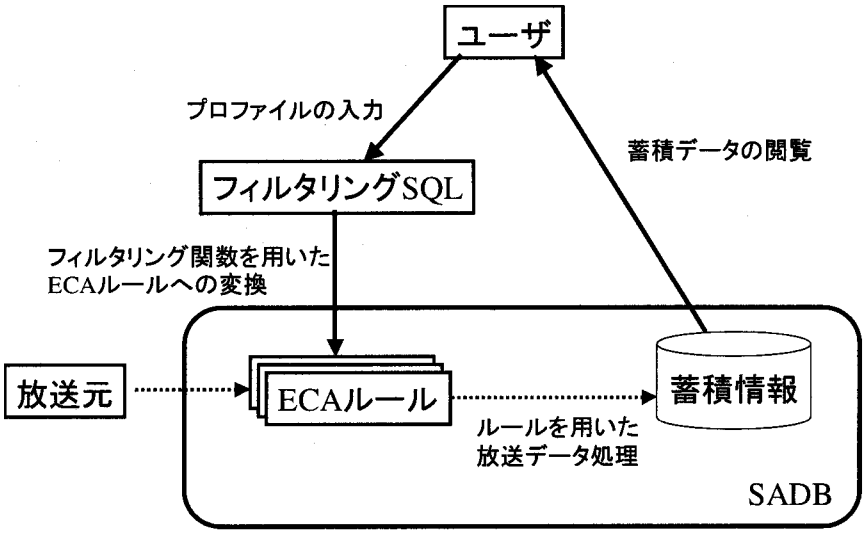


図 4.11: システムの概要

4.4.2　処理方法最適化機構をもつ情報フィルタリングシステム

一般に，情報フィルタリングシステムでは，データを受信するたびに取捨選択を行う逐次処理を行うのが一般的である [6]．しかし，大量の受信データを処理することを考えた場合，逐次処理では受信データを逐一処理するため，受信機の処理コストが非常に高くなる可能性がある．フィルタリングの処理コストを軽減するためには，複数の受信機で処理を分散する並列処理や，ある程度データをためておき一度に処理する一括処理が有効である．最適な処理方法は受信機の数や空きリソース，ネットワークの負荷など環境により変化するため，状況に応じて処理方法を変換する必要があるが，処理方法の変換を行うには，システム稼動中に処理方法を変更しても一貫したフィルタリング結果が得られることを保証する必要がある．

そこで，フィルタリング結果の一貫性を保証しつつ処理方法の動的変換を行うために，筆者らが提案しているフィルタリング関数の数学的性質を利用した情報フィルタリングシステムを構築する．提案システムは逐次処理，一括処理なと各種の処理を行うECAルール群により動作し，数学的性質により結果の一貫性を保ちながらECAルールを選択的に利用してフィルタリングの処理方法を動的に変換する．

本システムの概要を図4.11に示す．処理の流れは次のようになる．

1. フィルタリングSQLによるプロファイル記述．

```
EXTRACT <属性>
FROM    <データリソース>
WHERE   <ユーザの嗜好>
        <付加条件>
```

図 4.12: フィルタリングSQL基本構文

```
EXTRACT *
FROM A放送
WHERE best(ESTIMATE, 20, DESC)
  PREFER NEWS TO SPORTS WITH Intensity
  PREFER 天気予報 TO レジャー WITH Littleness
  PREFER SPORTS TO グルメ
  PROHIBIT アニメ
```

図 4.13: フィルタリングSQLの例

2. フィルタリング関数に基づく等価な処理方法の決定.

3. ECAルールへの変換.

4. 環境に応じた最適な処理方法の実行.

以下，それぞれのステップについて述べる.

フィルタリングSQLによるプロファイル記述

本システムでは，フィルタリングポリシーの記述に，筆者らの研究グループで提案したフィルタリングSQLを用いる．フィルタリングSQLとは，データベースへの問合せ言語であるSQL(Structured Query Language)を基にフィルタリング要求を記述できるようにした言語で，ユーザのプロファイル記述およびデータ容量制限などのデータ管理ポリシーを記述できる．フィルタリングSQLの基本構文を図4.12に示す．EXTRACT句で受信データアイテムのうち蓄積する属性を指定し，FROM句で放送のチャンネルを指定する．WHERE句においてユーザの嗜好を記述し，必要に応じて付加条件を記述する．詳細な言語仕様は文献[48]に記述されている.

図4.13にフィルタリングSQLの記述例を示す．この例では，ジャンル$SPORTS$よりも$NEWS$が非常に好きであり，$AMUSEMENT$よりも$WEATHER$が好きであるが，$ANIME$は嫌いであるというユーザの嗜好に対して，A放送から受信したデータのうち最も評価値が高いデータ20個を蓄積する．このようにフィルタリングSQLを用いることで，ユーザはフィルタリングポリシーを宣言的に記述できる.

フィルタリング関数に基づく等価な処理方法の決定

本システムは，処理方法を変化させてもフィルタリング結果の一貫性を保証するために，フィルタリングを関数として表現したフィルタリング関数の数学的性質を利用する．フィルタリング関数の枠組みでは，フィルタリングの性質は，関数が満たす制約条件によって表されており，フィルタリングの定義が決まれば等価な処理手法も決定される．フィルタリング関数の体系では，次の4つの処理手法を取り扱っている．

- 逐次処理: データが受信機に到着するたびに，その受信データと前回までに蓄積した結果を合わせてフィルタリングする処理方法．フィルタリングシステムにおいて一般に最も用いられている．
- 一括処理: 受信データをある程度ためておき，一括してフィルタリングする処理手法．
- 分配処理: 複数の受信機でデータを分けて受信し，並列にフィルタリングした後，それらの結果を合わせたものをフィルタリング結果とする処理方法．
- 並列処理: 分散処理した結果をさらにもう一度フィルタリングしてその結果をフィルタリング結果とする処理手法．

また，本体系では次のようなフィルタリング手法を取り扱い，その性質を数学的に明らかにしている．

- セレクション: データ間の相関性を考慮せず，コンテンツや属性などからデータごとに取捨選択を決定するフィルタリング手法．キーワードマッチングや，一定値以上の属性値をもつデータを蓄積するようなフィルタリングがセレクションである．
- ランキング: ユーザの嗜好に応じて重要度の高い順にデータを並べ，上位の特定の数のデータだけを蓄積するフィルタリング手法．
- データの相関性を考慮する手法: フィルタリングするデータをひとつひとつ判断するだけでなく，特定のデータが揃うことでデータの評価値が上下するフィルタリング手法．
- ディスク容量を制限する手法: 蓄積するディスク容量に制限を設け，制限を超えない限りは蓄積し続ける手法．

表 4.4: 単一のフィルタリング手法における等価な処理

フィルタリング手法		等価な処理方法
セレクション		一括処理 逐次処理 並列処理 分配処理
ランキング		一括処理 逐次処理 並列処理
相関性を考慮する手法	特定のデータにより評価を上げる手法	なし
	特定のデータにより評価を下げる手法	一括処理 逐次処理 並列処理
ディスク容量を制限する手法	重要度のみを考慮する手法	なし
	重要度以外の条件も考慮する手法	一括処理 逐次処理 並列処理

表 4.5: 複数の手法を組合せたフィルタリングにおける等価な処理

フィルタリング手法			等価な処理方法
セレクション → セレクション			一括処理 逐次処理 並列処理 分配処理
ランキング(蓄積数n) → ランキング(蓄積数n')	同じ属性を基準とした ランキングの組合せ		一括処理 逐次処理
	異なる属性を基準とした ランキングの組合せ	$n \leq n'$	並列処理
		$n > n'$	なし
セレクション → ランキング			一括処理 逐次処理 並列処理
ランキング → セレクション			なし

これらの手法は，単独で，または組み合わせて利用される．これらのフィルタリング手法において，それぞれ等価な処理手法は表4.4，4.5に示すように明らかになっている．証明など導出に関する詳細は文献[49][50][51][52]に記述している．この表を利用することで，フィルタリングSQLによって要求が記述されたとき，そのフィルタリング処理がどの処理方法の間で等価であるかが判断できる．例えばフィルタリングSQLの記述内容が，セレクションを行ってからランキングを行う内容であれば，そのフィルタリングは一括・逐次・並列のどの処理手法に変更してもフィルタリング結果の等価性が保たれる．

ECAルールへの変換

フィルタリングSQLでユーザ要求が記述されたとき，その要求に対して等価な処理手法はフィルタリング関数の体系を利用することで明らかとなる．そこで，等価な各処理手法について，それぞれの処理を実行するECAルール群を生成する．ここでは例として図4.14に示すようなフィルタリングSQLが記述された場合を考える．この記述は「A放送から受信したデータのうち，ジャンルが$NEWS$であるデータが欲しい」という要求を意味するが，これは単一のセレクションによるフィルタリングを意味するため，逐次・一括・並列・分散の各処理のフィルタリング結果の一貫性が保証される．そこで，4つの処理それぞれについてフィルタリング処理を実行するECAルール群を作成する．作成されるECAルールを図4.15〜4.19に示す．また，本システムは，等価な処理方法に対しての変換処理自体もECAルールで記述する．図4.20に示すように，CPU利用率の変化や複数の受信機が利用可能かどうかをルールで判断し，処理の変更を行う．例えば，図4.21は，CPU利用率が50％以上であれば一括処理へ変換するルールである．

システムの実装

以上に示したフィルタリングシステムを実装した．放送するコンテンツとしては，地上波データ放送であるADAMSとBITCASTを利用した．クライアントシステムはフィルタリングSQLを記述するためのエディタおよび蓄積データを参照するためのビューアを実装した．システムを利用している様子を図4.22に示す．

4.4.3　問合せ機構をもつ放送型データ受信システム

これまでのシステム想定してきた環境では，放送データを単なるデータアイテムとして取り扱ってきた．本節では，特にリレーショナルデータベースの内容を繰り返し放送するような放送型データベースシステムに着目する．放送型データベースシステムとは，図4.23に示すように，サーバがリレーショナルデータベースの内容を周期的に放送し，PDAなどの非力な端末も含めたクライアントが放送を受信してサービスを受けるシステムである．サーバからクライアントへの放送帯域は，データベース内容を放送するための広帯域のメイン放送帯域と，それ以外の内容を放送するための狭帯域のサブ放送帯域に分けられる．クライアントからサーバへは狭帯域のアップリンクが存在し，クライアントはアップリンクを利用して問合せを送信できるとする．

1.3-5mm

```
EXTRACT *
FROM A放送, B放送
WHERE GENRE = NEWS
```

図 4.14: フィルタリング記述例

```
ルール1
E: META_RECEIVE
C: (NEW.RESOURCE = A放送 OR
    NEW.RESOURCE = B放送) AND
    NEW.GENRE = NEWS
A: QUERY(''INSERT INTO 蓄積テーブル'')
ルール2
E: CONTENT_RECEIVE
C: DB.StoreTable.ID = NEW.ID
A: STORE_FILE
```

図 4.15: 逐次処理の ECA ルール

```
ルール1
E: META_RECEIVE
C: (NEW.RESOURCE = A放送 OR
    NEW.RESOURCE = B放送)
A: QUERY(''INSERT INTO 一時テーブル'')
ルール2
E: TIMER
A: QUERY(''SELECT * FROM 一時テーブル'')
ルール3
E: SELECT 一時テーブル
C: NEW.GENRE = NEWS
A: QUERY(''INSERT INTO 蓄積テーブル'')
   STORE_FILE
```

図 4.16: 一括処理の ECA ルール

```
ルール1
E: META_RECEIVE
C: NEW.RESOURCE = B放送
A: QUERY(''INSERT INTO 一時テーブル'')
ルール2
E: TIMER
A: QUERY(''SELECT * FROM 一時テーブル'')
ルール3
E: SELECT 一時テーブル
C: NEW.GENRE = NEWS
A: SEND_FILE 依頼元端末にデータを送信
```

図 4.17: 分配処理の ECA ルール (依頼先用)

```
ルール1
E: META_RECEIVE
C: NEW.RESOURCE = A放送
A: QUERY(''INSERT INTO 一時テーブル'')
ルール2
E: TIMER
A: QUERY(''SELECT * FROM 一時テーブル'')
ルール3
E: SELECT 一時テーブル
C: NEW.GENRE = NEWS
A: QUERY(''INSERT INTO 蓄積テーブル'')
   STORE_FILE
ルール4
E: META_RECEIVE
C: NEW.ADDRESS = 依頼先端末
A: QUERY(''INSERT INTO 蓄積テーブル'')
ルール5
E: CONTENT_RECEIVE
C: NEW.ADDRESS = 依頼先端末
A: STORE_FILE
```

図 4.18: 分配処理の ECA ルール (依頼元用)

```
ルール1
E: META_RECEIVE
C: NEW.RESOURCE = A放送
A: QUERY(''INSERT INTO 一時テーブル1'')
ルール2
E: TIMER
A: QUERY(''SELECT * FROM 一時テーブル1'')
ルール3
E: SELECT 一時テーブル1
C: NEW.GENRE = NEWS
A: QUERY( ''INSERT INTO 一時テーブル2'')
ルール4
E: META_RECEIVE
C: NEW.ADDRESS = 依頼先端末
A: QUERY( ''INSERT INTO 一時テーブル2'')
ルール5
E: INSERT 一時テーブル2
C: 依頼先・依頼元端末のフィルタリング結果を全て格納
A: QUERY( ''SELECT * FROM 一時テーブル2'')
ルール6
E: SELECT 一時テーブル2
C: NEW.GENRE = NEWS
A: QUERY( ''INSERT INTO 蓄積テーブル'')
   STORE_FILE
```

図 4.19: 並列処理の ECA ルール (依頼元用)

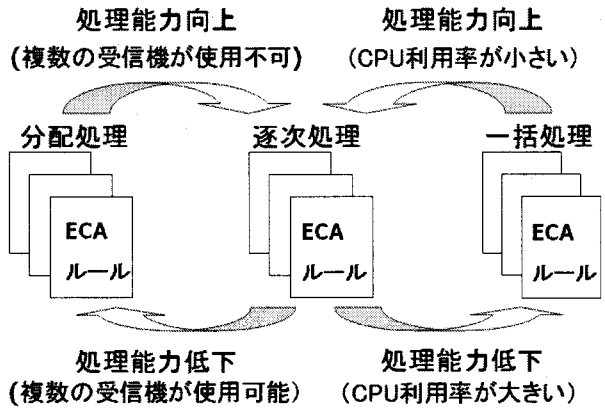


図 4.20: 処理方法の変換

E　TIMER
C　CPU_USAGE ≥ 50%
A　DISABLE_RULE 逐次処理
　　ENABLE_RULE 一括処理

図 4.21: CPU利用率による変換例

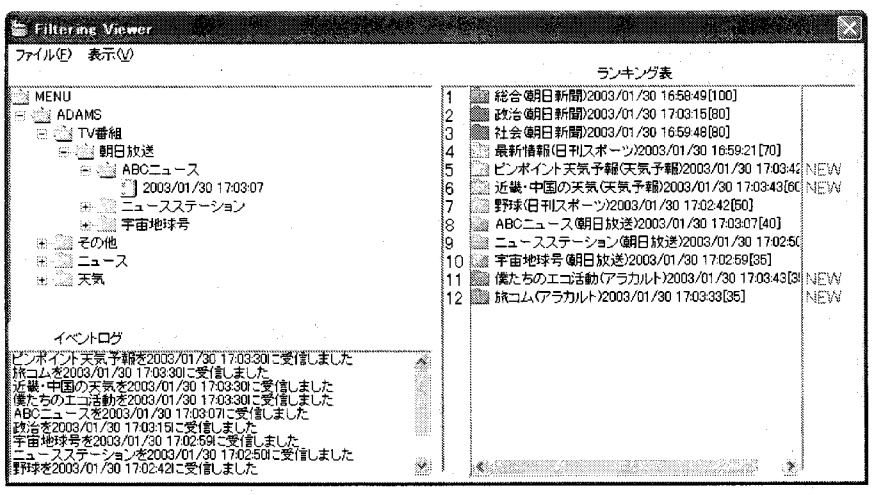


図 4.22: フィルタ結果のビューア

このようなシステムの利用例としては，サーバがショッピングセンター内の広告情報や店舗情報，また店舗で扱っている商品情報を含むデータベースを放送し，ユーザは携帯端末を持ち歩きながら放送される情報を受信して利用するといった形態が考えられる．このとき，通常クライアントは放送を受信することのみで要求を満たしているが，「商品Aの画像とその商品を扱う店舗の地図が欲しい」といった複雑な情報検索を行いたい場合には，サーバに対して問合せを発行する必要がある．そこで，本節では放送型データベースシステムにおいてECAルールを用いた効率的な問合せ処理を行える放送型データ受信システムを構

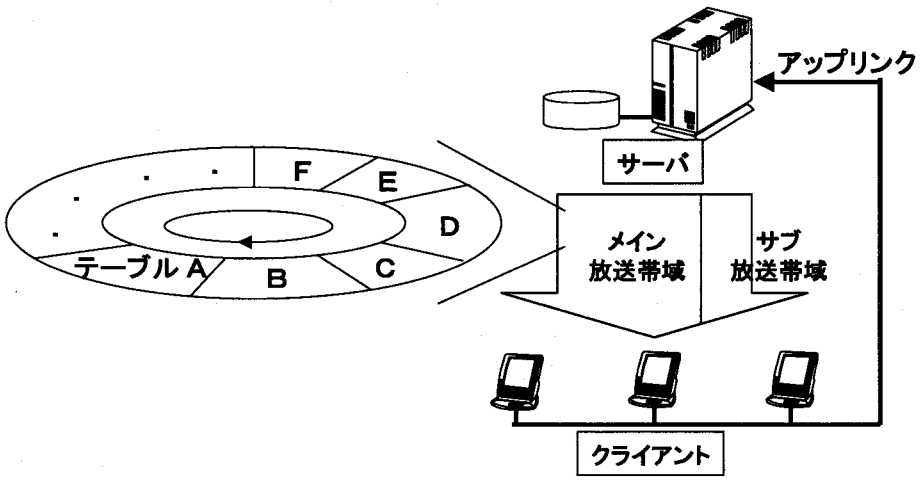


図 4.23: 放送型データベースシステム

築する．具体的には，サーバとクライアントが協調して問合せ処理を行うことで，問合せ処理の際に必要なクライアントの記憶領域および応答時間を削減する．

放送型データベースシステムに対する問合せ方式

放送型データベースシステムにおいて放送されるデータに問合せを行う場合，一般に次の2つの方式が考えられる．

- クライアント型方式

  クライアント型方式とは，クライアントのみで問合せを処理する方式である．クライアント上でSQLによる問合せが発生すると，クライアントはメイン放送帯域を監視し，問合せに関係するすべてのテーブルを受信してローカルディスクに蓄積する．その後，蓄積したテーブルに対して問合せ処理を行い，結果を得る．クライアント型方式では，クライアント上で問合せ処理が完結するため，クライアント数が増加しても，1放送周期以内に問合せに関係する必要なすべてのデータを蓄積でき，問合せ結果が得られる．また，アップリンクを用意できない環境でも利用できる．

- オンデマンド型方式

  オンデマンド型方式とは，サーバが問合せを処理する方式である．クライアントがアップリンクを利用して問合せをサーバに送信し，サーバが問合せ処理を行った後で

サブ放送帯域を用いて問合せ結果をクライアントに配信する．オンデマンド型方式では，問合せ処理のすべてをサーバが実行するため，クライアントは問合せを処理するためのディスク領域を必要としない．また，発生する問合せ数が少ない場合，問合せ結果が放送されるまでの待ち時間がなく，すぐに結果を取得できる．

しかし，これらの方式にはいくつかの問題点がある．まず，クライアント型方式では問合せに関連するすべてのテーブルをディスクに蓄えて処理するため，クライアントに必要とされる作業用領域が非常に大きい．テーブル内のデータのうち，結果に現れるタップルは一般にわずかであるが，結合演算等を行うためには他のタップルの情報や不要な属性の情報も用いるため，テーブル内容すべてを蓄積する必要がある．特に，クライアントが十分なディスク領域をもっていない時は，問合せを処理できない場合がある．また，問合せ処理は一般に負荷の高い処理となるため，クライアントの処理能力が低い場合，問合せ処理に処理能力のほとんどを奪われてしまう．処理コストは問合せの複雑さによって大きく変化するため，必要なスペックをあらかじめ予想することが難しい．

オンデマンド型方式に関しては，問合せが頻繁に起こる場合や問合せの結果サイズが大きい場合にサブ放送帯域が枯渇するため，クライアントが問合せ結果を受信するまでに非常に長い時間がかかる可能性がある．

協調型方式

このように，クライアント型方式とオンデマンド型方式ともに，状況によってはクライアントが問合せ結果を長時間受け取れなかったり，問合せ処理自体を行えない可能性がある．そこで，そこで，この問題を解決するため，放送型データベースシステムにおいてECAルールを用いて効率的な問合せ処理を行う協調型方式を提案する．協調型方式では，サーバとクライアントが協調して問合せ処理を行うことで，クライアント型方式に比べてクライアントのディスク使用量を小さくし，オンデマンド型方式に比べて応答時間を低減する．協調型方式の特徴を以下に示す．

- サーバによる問合せ処理の補助: クライアントは，アップリンクを利用して問合せをサーバに送信する．問合せを受け取ったサーバは問合せを処理し，問合せ結果に含まれるタップルに処理用の識別子を付加する．クライアントはメイン放送帯域を用いて放送されるデータベースのうち，識別子を参照して必要なデータのみを蓄積するため，ディスク使用量を低減できる．

- ルールによる受信データ処理の指定: サーバは，タップルに識別子を付加した後，クライアントがデータを処理するためのECAルールを作成し，サブ放送帯域を用いて放送する．クライアントが自分宛に放送されたルールを受信すると，ルールが自動的に問合せ結果に含まれるタップルのみを蓄積し，問合せ結果を再現する．一般にルールのサイズは非常に小さいため，問合せ結果をそのまま放送するのに比べて放送帯域を圧迫しない．本システムにおけるECAルールはSADBにおいて使用できるECAルールと同様であるが，自然結合演算のためのマッチングを行うアクション$MATCH$を新たに追加している．

問合せ処理アルゴリズム

提案方式の問合せ処理手順は以下のとおりである．

1. 識別子の書き込み
   サーバは問合せを受け取ると，その問合せに対して一意の識別子($QueryID$)を割り当てる．また，放送するデータベースの各タップルには，処理用の識別子を書き込むための属性として，問合せ識別子Q_ID，組合せ識別子C_IDを用意しておく．サーバは受信した問合せを解析し，問合せ結果に含まれるタップルに$QueryID$を付加する．クライアントはこのIDを参照して必要なタップルのみを蓄積する．また，結合演算のように複数のテーブルから問合せ結果が得られる場合，問合せ結果を構成する複数のタップルに対して，組合せ識別子C_IDの領域に同じ値を書き込む．クライアントは，組合せ識別子C_IDが等しいタップルを結合することで問合せ結果を再現する．

2. ECAルールの作成と放送
   サーバは，クライアントが問合せ結果を得るためのECAルールをテンプレートを用いて作成する．作成されるECAルールは，問合せ結果に含まれるタップルが放送される時間だけ放送帯域を監視するため，消費電力を低減できる．ECAルール群はサブ放送帯域を用いて放送される．

3. 問合せ結果の獲得
   問合せを行ったクライアントはこれらのECAルールを受信し，これらのECAルールがメイン放送帯域から自動的に必要なタップルを受信して問合せ結果を作成する．

```
SELECT    X.Attri-1 , Y. Attri-2
FROM      X, Y
WHERE     X.ID = Y.ID
AND       Y.Attri-2 <= 15
```

図 4.24: 問合せ例

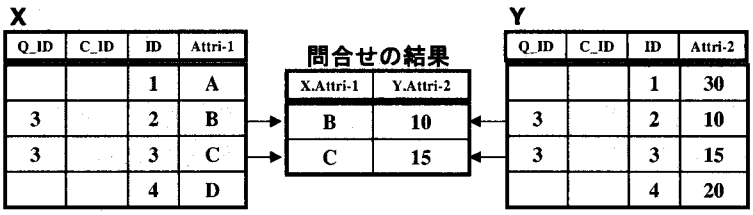


図 4.25: 問合せ識別子の書き込み

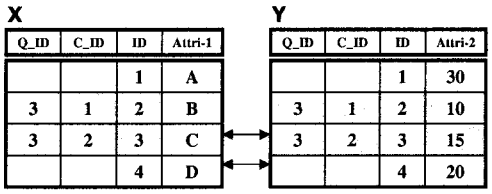


図 4.26: 組合せ識別子の書き込み

問合せ実行例

クライアントが，図4.24に示すSQL文によって問合せを行う場合の提案方式の実行例を示す．サーバは，受信した問合せに対して$QueryID$を決定する．ここでは仮に3とする．次に，図4.25に示すように，問合せの結果に含まれるタップルを調べ，それぞれのタップルの問合せ識別子に$QueryID$を書き込む．そして，図4.26に示すように，結果テーブルの各タップルが元のテーブルのどのタップルの組によって構成されるのかを調べ，該当するタップルの組合せ識別子に同じ値を書き込む．

次にECAルールを作成する．この例では図4.27に示すルール群が作成される．Rule3-1は，問合せ結果を作成するためのルール，Rule3-2，Rule3-3は，放送データ受信と問合せ結果表示のタイミングを計るためのルール，Rule3-4，Rule3-5，Rule3-6，Rule3-7は，データを受信し蓄積するためのルールである．Rule3-8，Rule3-9は結果を表示し，その後すべてのルールおよびタイマを削除するルールである．クライアントがタップルを受信する場合，テーブルXとテーブルYの放送順序によって，結合の順序や結果を表示するタイミングが異なる．そこで，テーブルXが最初に放送されてきた場合には，Rule3-4，Rule3-6，Rule3-8が使用されるように，Rule3-2でタイマをセットする．また，テーブルYが最初に放送されてきた場合には，Rule3-5，Rule3-7，Rule3-9が使用されるように，Rule3-3でタイマをセットする．ルール中のNowは現在時刻を表し，Rule3-4では識別子が付加されたタップルが放送される時間である50秒から70秒の間に受信したタップルを蓄積する．

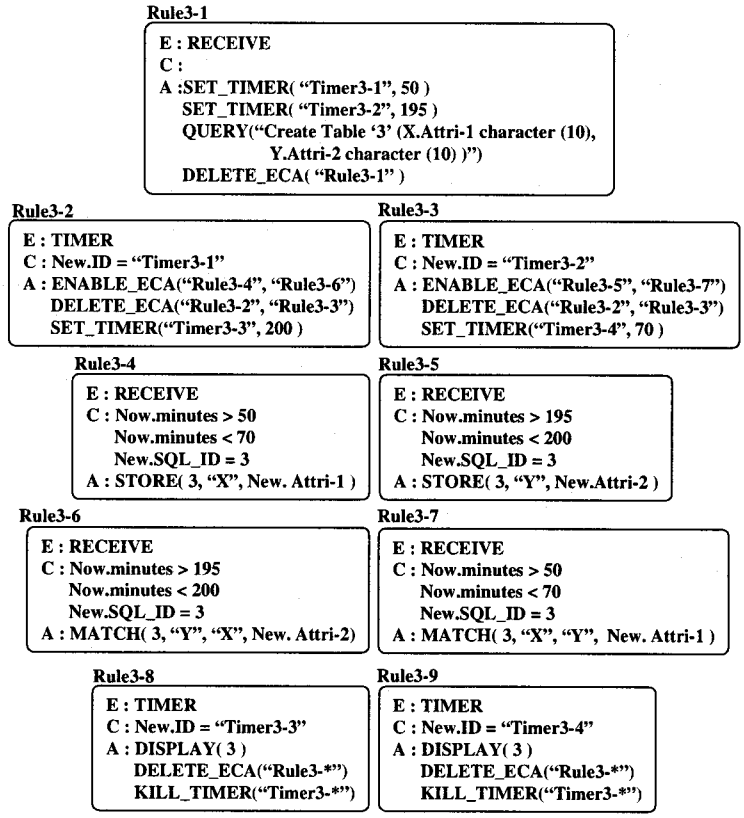


図 4.27: 作成されるECAルール

評価

本方式の有効性を調べるために，クライアント型方式およびオンデマンド型方式との比較評価を行った．評価は，クライアントが問合せ処理に必要とするディスク使用量および問合せの応答時間を基準として行った．

図4.28に，タップル利用率を変化させたときの各方式のディスク使用量の変化を示す．タップル利用率とは，テーブル中のすべてのタップルのうち，問合せ結果に現れるものの割合である．オンデマンド型方式では，タップル利用率に比例して問合せ結果のサイズが大きくなるため，ディスク使用量は右上りの直線になっている．クライアント型方式のディスク使用量は，問合せ結果のサイズと問合せに関係するテーブルのサイズの和で表されるため，オンデマンド型方式のディスク使用量に比べ，一定量大きくなっている．一方，協調型方式では，問合せ結果に現れるタップルのみを蓄積できるため，クライアント型方式に比べディスク使用量は小さくなっている．一般的な環境でのタップル利用率が高々1% 程

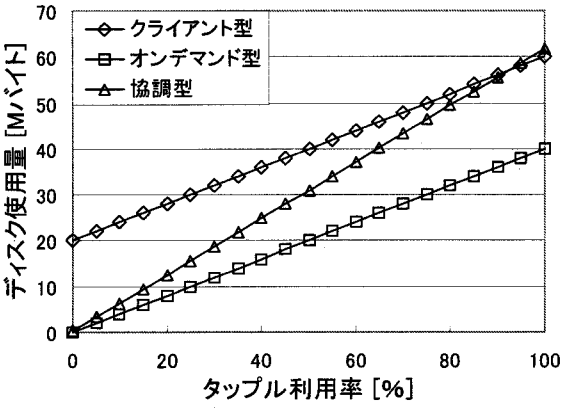


図 4.28: データ利用率とディスク使用量

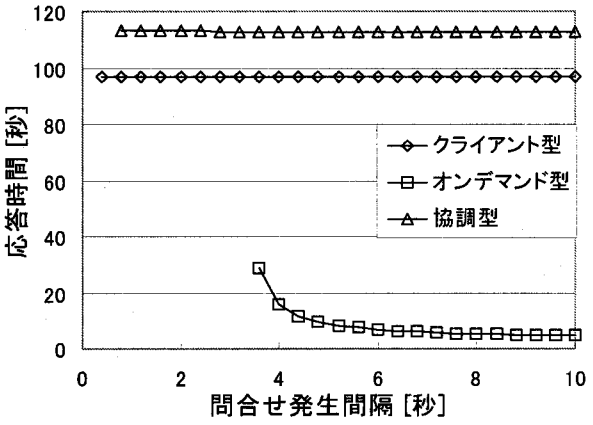


図 4.29: 問合せ発生間隔と応答時間

度と仮定した場合，協調型方式のディスク使用量は約600K バイトとなる．クライアントが，PDA のように数M バイト程度しか記憶領域をもたない場合，クライアント型方式では，問合せ処理に必要なすべてのデータを蓄積できない場合があるが，協調型方式では十分に蓄積できる．

図4.29に問合せ発生間隔を変化させたときの応答時間の変化を示す．クライアント型方式では，クライアント自身がすべての問合せ処理を行うため，問合せ発生間隔は応答時間に影響しない．オンデマンド型方式では，問合せ発生間隔が3.4秒以下では，グラフが切れている．これは，サーバが問合せ結果を放送する速度よりもキューに新たに入る問合せ結果の量が大きくなり，待ち行列が破綻するためである．破綻した待ち行列における応答時間は非常に大きくなる．協調型方式はクライアント型方式に比べてわずかに応答時間が長くなっている．これは，各テーブルが識別子保持用の領域分だけ大きくなり，放送周期が長くなるためである．協調型方式では，問合せ発生間隔が0.6秒以下の領域ではグラフが切れている．これは，問合せの到着間隔が識別子領域が開放されるまでの時間に比べ小さくなり，待ち行列が破綻してしまうためである．破綻した待ち行列における応答時間は非常に大きくなる．問合せ発生間隔が小さい場合，オンデマンド型方式に比べ協調型方式の応答時間は小さいため，協調型方式の方が有効である．一方，問合せ発生間隔が大きい場合はオンデマンド型方式の応答時間が短くなる．

4.5 むすび

本章では，放送型データ受信のためのアクティブデータベースシステムであるSADBの設計とその応用システムについて述べた．SADBはECAルールで動作するため，さまざまなタイプのアプリケーションを構築するために充分な柔軟性をもつ．さらにECAルールの処理は放送データの受信処理に最適化されているため，放送型データ受信システムの構築に適している．応用システムとして構築した3つのアプリケーションはそれぞれ目的が異なるシステムであるが，SADBを基盤として用いることでアプリケーションが容易に構築できていることがわかる．今後の課題としては，効率的なコンディション評価やアクションの並列実行などさらなるECAルール実行の高速化が挙げられる．

第5章

状況依存コンテンツのリアルタイム提示のためのアクティブデータベース

5.1 まえがき

近年，マルチメディア技術の進歩に伴い，マルチメディア素材をデータベース中に格納し，リアルタイムプレゼンテーションに用いる技術が注目されている．リアルタイムプレゼンテーションとは，あらかじめ提示する素材を決定しておくのではなく，図5.1に示すようにシナリオおよびユーザからの入力など周囲の状況から最適な素材をリアルタイムに抽出して提示するものであり，インタラクティブな商品説明システムやカーナビゲーションシステムなどさまざまな分野での活用が期待されている．リアルタイムプレゼンテーションは，周囲の状況の変化に対応して適切な素材を動的に抽出する必要があるため，イベント駆動型システムの枠組みが有効であるといえる．本章では，リアルタイムプレゼンテーションの例として，日本の代表的な娯楽の一つであるカラオケを取り上げる．

カラオケは歌うことが第一の目的であるが，市場の8割を酒屋とカラオケボックスが占め，個人契約のネットワークカラオケはそれほど普及していない．このことからもわかるように，カラオケは単に歌うだけが目的ではなく，人とのコミュニケーションを図る重要な手段のひとつであるといえる[76]．そのため，カラオケシステムは，常に新しい技術を取り込みつづけ，よりカラオケが盛り上がるようなさまざまな工夫を凝らしている．

カラオケを構成する要素としては，歌い手の声，BGM，背景映像，その他の付加要素が挙げられるが，例えば声に関しては，ボイスエフェクト機能を用いて一人でデュエットす

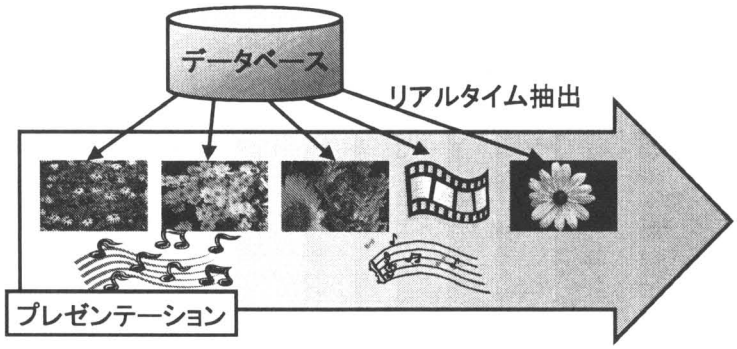


図 5.1: リアルタイムプレゼンテーションシステム

る機能などにより，カラオケを盛り上げることができる．BGMに関しては，移調して歌いやすい音域に合わせることでユーザは気持ちよく歌えるようになる．また，その他の付加要素として，画面に指示されるとおり踊りながら歌を歌うDAM-DDR(DAM)やフリカラ(BeMAX'S)，歌うのに消費したカロリーを表示してくれるカロリーカラオケ(DAM)など，各社さまざまな機能を提供している．

一方，カラオケの背景映像に関しては，曲を選択してもいくつかの背景映像のなかからある程度曲調を考慮して選択された映像を流すだけのものが大半である．そこで本章では，カラオケの背景映像をリアルタイムプレゼンテーションとして捉え，アクティブデータベースを用いて背景を動的に生成し，カラオケを盛り上げるシステムであるアクティブカラオケを構築する．

リアルタイムプレゼンテーションを実現するためには，シナリオの進行に合わせてリアルタイムでコンテンツを検索して提示したり，状況の変化に応じた表示効果の適用や特別なコンテンツの提示を行う必要がある．したがって，アクティブカラオケではシナリオの進行や状況の変化をイベントとして処理できるようにアクティブデータベースを拡張することで，ECAルールを用いたリアルタイムプレゼンテーション制御を実現する．また，リアルタイム検索を実現するため，キーワードからコンテンツへの相互変換辞書を用意して検索の高速化を実現する．以下，5.2節ではアクティブカラオケの概要と，アクティブカラオケにおけるECAルール仕様について述べる．5.3節ではシステムの設計と実装について述べ，5.4節で本システムの考察を行う．最後に5.5節で本章のまとめを行う．

図 5.2: システムのイメージ図

5.2 アクティブカラオケ

アクティブカラオケのシステムイメージを図5.2に示す．BGV素材データベースにはカラオケの背景となる画像が格納されており，それぞれの素材にはキーワードなどのタグ情報が付加されている．音楽情報データベースには実際の曲データや歌詞データが格納され，歌詞データには，歌詞の内容や表示タイミングのほかに，テーマやサビといった楽曲の構成情報（パート情報）などの付加情報も含まれる．ルールベースにはECAルールが格納されており，ルールはどの画像を選択するか判断したり，特殊効果を加えるために用いられる．

アクティブカラオケは，演奏が行われている間，音楽情報データベース内の情報と，外部からの入力，ECAルールでの制御によってリアルタイムに素材を検索し，カラオケの背景として表示する．

アクティブカラオケは以下の特徴をもつ．

- 曲に応じた画像選択

  画像データベースの中から，歌詞や曲調の変化，盛り上がりに応じて適切な画像を選択し，カラオケの背景画像として表示する．

- さまざまな演出効果

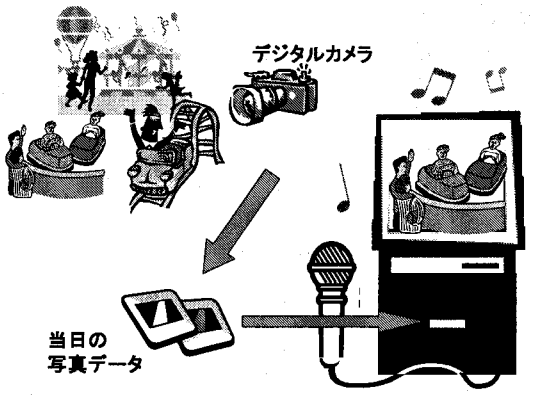


図 5.3: 利用イメージ1

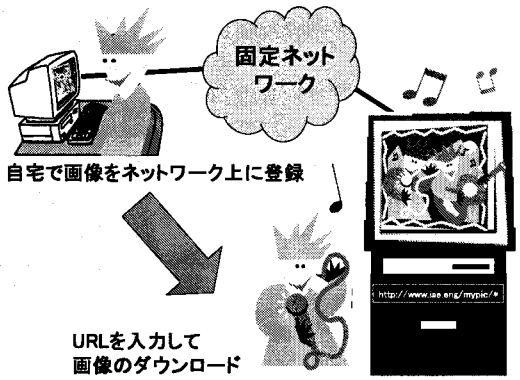


図 5.4: 利用イメージ2

背景画像の表示の際，さまざまな画像効果や表示効果などの演出効果を加えることで，カラオケを盛り上げる．たとえば，落ち着いた曲に対しては，落ち着いた色合いで画像を表示するといった効果が加えられる．

- オリジナル素材の利用

  デジタルカメラで撮影した写真などのオリジナル素材をその場で登録できる．背景画像に自分や知り合いの画像が使われることで，一般の画像のみを用いる場合と比べて歌い手や聴衆の興味をひくことができる．

これらの機能を用いたアクティブカラオケの利用モデルとしては以下のような形態が考えられる．

- 街角で撮った写真をすぐに利用してそれを背景にして歌う (図5.3).

  デジタルカメラを持って友人とアミューズメント施設などに遊びに行き，その場でたくさんの写真を撮っておく．帰りにカラオケボックスに寄ってカラオケをするとき，デジタルカメラの記録メディアをカラオケ機器に挿入することで，当日の写真を背景に歌うことができる．デジタルカメラにコメント付加機能があれば，写真を撮った際に付けたコメントをもとに，曲中の適切な場面で表示することも可能になる．

- あらかじめ用意しておいた画像データをダウンロードして背景にして歌う (図5.4).

  自分が用意しておいた写真にキーワードをつけてWWWサイト上にあらかじめ置いておく．カラオケボックスに行ったときに，カラオケ機器上で画像を置いてあるURL

を入力すると，カラオケ機器がネットワークから画像データをダウンロードし，それらの画像を背景に歌うことができる．URLごとにテーマを決めて，複数の画像サイトを準備しておくことで，用いる画像のテーマを選択できる．

5.2.1 アクティブカラオケのECAルール

アクティブカラオケにおけるECAルールの記述構文は2章で述べたAMDSのものと同様である．アクティブカラオケで取り扱うことのできるイベント，アクションを表5.1，5.2に，NEWデータ，OLDデータの内容を表5.3示す．*MUSIC_START*，*MUSIC_STOP*イベントは曲の再生開始，終了時に起こるイベントで，曲ごとの指定を行うルールを発火させるために用いる．*MTYPE_CHANGE*イベント，*MPART_CHANGE*イベントは，曲のジャンル，パートが変化したときに起こるイベントで，前者はロック，ポップスなど曲のジャンルに応じた処理を行うためのもの，後者はAメロ，サビなど曲中の区切りに応じた処理を行うためのものである．*FIND_WORD*イベントは歌詞中に特定の文字が出てきたときに起こるイベントで，特定の歌詞に対して特別な写真を表示したい場合などに用いる．*VOLUME_CHANGE*イベントは，マイクからの入力音量が大きく変化した場合に起こるイベントで，「歌が盛り上がったときには特定の画像を表示する」といったルールが記述できる．*QUERY*アクションは状況に応じて画像を削除したり，他の場所にある画像データをデータベースに追加するルールが記述できる．*APPLY_EFFECT*アクションは，画像の出力に効果を与えるアクションで，効果範囲・効果の種類を記述する．*APPLY_EFFECT*アクションで記述できるエフェクトの種類を表5.5に示す．エフェクトは位置に関するもの，画像に対する効果に関するもの，表示の方法に関するものの3つのパターンに分類でき，それぞれを組み合わせて利用できる．また，指定できる効果範囲を表5.4に示す．*DISPLAY_GRAPHIC*アクションは，表示時間と表示画像を指定して特定の画像を表示するアクションである．表示時間は*APPLY_EFFECT*アクションの場合と同様に指定する．

ECAルール記述例を図5.5に示す．ルール例1は曲を再生したときに，曲の種類がバラードであれば，雰囲気を出すために表示画像をすべてセピア色に変換してから出力するルールである．ルール例2はサビを歌うときは自分の気にいった写真をバックに歌いたいという要求を満たすためのルールで，曲を再生中，パートがサビの部分になったとき特定の画像を表示するルールと，サビが終わったときにその表示を解除するルールから構成される．ルール例3では，前奏や間奏など，キーワードがない部分で表示する画像を決めるために，

表 5.1: アクティブカラオケのイベント

名称	内容
SELECT	データ参照
INSERT	データ挿入
DELETE	データ削除
UPDATE	データ更新
MUSIC_START	曲の開始
MUSIC_STOP	曲の停止
MUSICTYPE_CHANGE	曲の種類変更
MUSICPART_CHANGE	曲の部分変更
FIND_SPECIFIC_WORD	特定の歌詞出現
INPUTVOLUME_CHANGE	入力音量の変化
RECEIVE	外部機器からの入力

表 5.2: アクティブカラオケのアクション

名称	内容
QUERY([クエリー内容])	データベース操作
INSERT_ECA([ルール内容])	ECA ルール格納
DELETE_ECA([ルール識別子])	ECA ルール削除
ENABLE_ECA([抽出条件])	ECA ルール有効化
DISABLE_ECA([抽出条件])	ECA ルール無効化
APPLY_EFFECT([効果指定])	画像に効果を加える
DISPLAY_GRAPHIC([表示画像])	特定画像の表示
SETFLAG([フラグ指定])	フラグの ON/OFF

表 5.3: NEW データと OLD データ

イベント	NEW	OLD
SELECT	参照データ	–
INSERT	挿入データ	–
DELETE	–	削除データ
UPDATE	更新後データ	更新前データ
MUSIC_START	開始曲情報	–
MUSIC_STOP	–	終了曲情報
MTYPE_CHANGE	変化後曲種	変化前曲種
MPART_CHANGE	変化後パート	変化前パート
FIND_WORD	見つけた場所	–
VOLUME_CHANGE	変化後音量	変化前音量
RECEIVE	入力データ	–

表 5.4: 適用範囲指定

指定	意味
ALL	曲中すべてに適用する.
RANGE([開始タイミング], [終了までの時間])	一定区間に効果を適用する. 開始タイミングにシステム変数を利用することで, 現在からという指定ができる.
UPTO([フラグ指定])	指定したフラグが満たされるまで効果を適用する.
ONLY_ONE	イベント発火の次のタイミングの画像にのみ効果を与える.

表 5.5: 効果の種類

名称	種類	内容
POS_NORMAL	位置	普通に画面全体に表示.
CHILD	位置	子画面を含む2画面表示. 子画像は引数で直接指定.
RANDOM	位置	ランダム画像を1/4の大きさで4隅に順番に表示.
SMALL	位置	1/4画像を画面のランダム位置に表示.
EFF_NORMAL	画像	画像をそのまま表示.
EFF_GRAY	画像	画像をグレースケール化して表示.
EFF_SEPIA	画像	画像をセピア化して表示.
EFF_FLASH	画像	画像にフラッシュ効果を与える.
DISP_NORMAL	表示	そのまま表示.
SLIDEIN_LEFT	表示	左からスライドイン表示. 同様に, 右, 上, 下からのスライドインも指定できる.
WIPEIN_LEFT	表示	左からのワイプイン表示. 同様に, 右, 上, 下からのワイプインも指定できる.
CENTER_ZOOM	表示	中央からズームインして表示.

曲間においては画像データベース中からランダムに画像を選び出し, 4分割で表示するようにしている. ルール例4は複雑なエフェクトを加える例として, サビになったときに, 特定画像を5秒間表示し, その表示にフラッシュ効果を与えると同時に, 中央からズームインしてくるように表示するルールである. ルール例5は, 外部からの入力を受け取る例で,

・ルール例1(セピア化)

```
CREATE RULE セピア化 ON MTYPE'CHANGE
WHERE NEW.TYPE = 'BALLADE'
THEN DO
 APPLY'EFFECT( ALL, EFF'SEPIA );
```

・ルール例2(画像の強制表示)

```
CREATE RULE 強制表示 ON MPART'CHANGE
WHERE NEW.TYPE = 'CLIMAX'
THEN DO
 DISPLAY'GRAPHIC( UPTO(0),
          'C:¥graphics¥climax.jpg');

CREATE RULE 強制表示解除 ON MPART'CHANGE
WHERE OLD.TYPE = "CLIMAX"
THEN DO
 SETFLAG(0, ON);
```

・ルール例3(曲間はランダム表示)

```
CREATE RULE ランダム表示 ON MPART'CHANGE
WHERE NEW.TYPE = "INTERLUDE"
THEN DO
 APPLY'EFFECT( UPTO(1), RANDOM );

CREATE RULE ランダム解除 ON MPART'CHANGE
WHERE OLD.TYPE = "INTERLUDE"
THEN DO
 SETFLAG(1, ON);
```

・ルール例4(複雑なエフェクト)

```
CREATE RULE エフェクト効果 ON MPART'CHANGE
WHERE NEW.TYPE = 'CLIMAX'
THEN DO
 DISPLAY'GRAPHIC( RANGE(SYS'NOWTIMING,5),
                  'C:¥climax.jpg' );
 APPLY'EFFECT( RANGE(SYS'NOWTIMING,5),
                     EFF'FLASH );
 APPLY'EFFECT( RANGE(SYS'NOWTIMING,5),
                     CENTER'ZOOM );
```

・ルール例5(外部入力)

```
CREATE RULE マラカス入力 ON RECEIVE
WHERE NEW.TYPE = "maraca"
THEN DO
 APPLY'EFFECT( ONLY'ONE, EFF'FLASH );
 DISPLAY'GRAPHIC( RANGE(SYS'NOWTIMING,1),
                  'C:¥maraca.jpg' );
```

図 5.5: ECA ルール例

外部入力を受け取ったときに，発信元がマラカスであれば，入力タイミングに合わせて画面をフラッシュさせながらマラカス用画像を表示するルールである．

5.3 システムの設計と実装

アクティブカラオケのシステム構成を図5.6に示す．音楽データベースには曲のデータおよび歌詞データが格納されている．BGVデータベースには画像データが格納されており，データベースがアクセスされたときにはデータベース関連イベントをイベント検出部に通知する．関連語辞書データベースには，キーワードとその関連語情報が格納されており，それぞれのキーワードに対応する画像データのIDも格納されている．演奏処理部では，音楽データベースを参照して曲の演奏を行う．演奏中は，歌詞データを字句解析してキーワードを抽出し，関連語辞書データベースを用いてキーワードに関連する画像をBGVデータベースから検索して表示する．イベント検出部では演奏イベントやデータベースイベント，外部からの入力イベントを検出し，ECAルール検索機構においてECAルールベースを検索して，対応するルールがあった場合はECAルール処理機構で実行する．インタフェース部では，ユーザにグラフィカルなユーザインタフェースを提供し，曲の選択，再生，停止などの操作を行わせる．

これまでにで述べたアクティブカラオケのプロトタイプシステムを実装した．実装はマイクロソフト社のWindows2000上でVisual Basic6.0を用いて行った．音楽データベースにはMP3形式およびMIDI形式の楽曲40曲程度を登録し，BGVデータベースには，静止画像1000枚程度を格納した．画像登録の際には各画像に手動で複数のキーワードを付加して登録した．実際にいくつかのECAルールを入力して動作させ，ルールが適切に動作することを確認した．また，加速度センサを用いた仮想マラカスを作成し，マラカスからの入力を処理するルールが正しく動作することを確認した．システムの稼動状況を図5.7に示す．実装したシステム上に前節で述べたルール例1〜5を格納して利用している様子を図5.8に示す．画面1はオープニングの部分で，ルール例3によりランダム画像が4分割され，ルール例1によってセピア色に変換されてから表示されている．画面2は曲中で，特にマッチするルールがないので単純に歌詞のキーワード(図の場合は「夏」)に合った画像が提示されている．画像3はサビの部分で，ルール4により，サビ用の特別な画像がエフェクトと共に表示されている．画像4は，マラカスからの外部入力があったときに，ルール5によってマラカス用の画像を表示している．

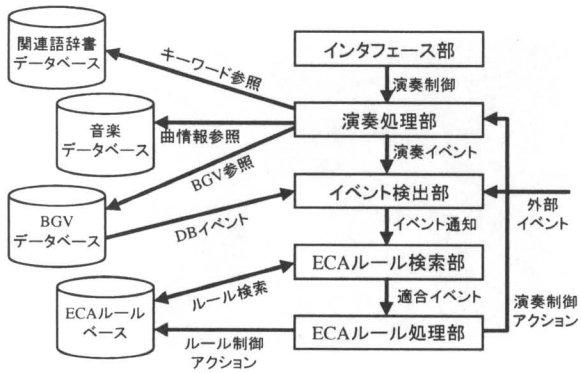


図 5.6: システム構成

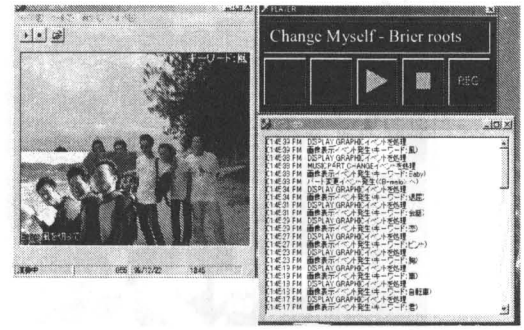


図 5.7: システム稼動状況

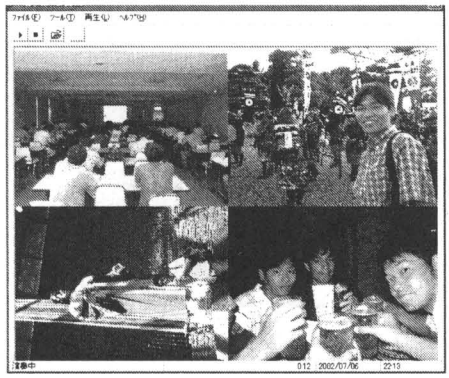
5.4 考察

ECA ルール利用のメリットについて

提案システムはECA ルールで動作を記述するため，各ユーザがそれぞれ自分用のルールを持ち歩くことで，自分用にカスタマイズされたカラオケシステムを利用できる．また，各ユーザがもつ個人的なルール群を統合したり，カラオケボックスの部屋間のネットワークを用いてECA ルールの統合や交換を行うといった利用方法も有効である．このように，ルールを用いたシステム制御の枠組みは，カラオケシステムにおける新たなコラボレーションやネットワーク利用を可能にすると考えられる．

処理速度について

システムの処理速度に関しては，キーワードから画像を検索する部分がボトルネックとなる．実際の処理は，キーワードから関連語辞書を用いて関連キーワード群を形成し，その群にマッチする画像を検索するという手順で行うが，本システムでは，キーワードからキーワード群へのマッピングをあらかじめ行っておくことでシステムの高速化を計っている．本手法は，処理の高速化が期待できる反面，キーワード辞書の更新ごとに再マッピングを行う必要がある．実際に，CPU がPentiumIII600Mhz，メモリ 192Mbyte のPC を用い，1000 枚程度の画像を格納して実験したが，0.1 秒以下の時間で検索できた．数千枚程度であればストレスのない処理が可能であると考えられる．


画面1 オープニング

画面3 サビ

画面2 曲中

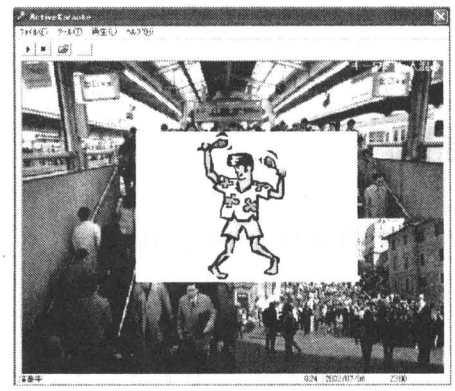

画面4 マラカス入力

図 5.8: ルール実行例

検索結果の妥当性について

検索結果の妥当性については画像を登録する際のキーワードと，関連語辞書に左右される．現在のシステムでは，単純なキーワードマッチングで検索を行っているため，例えばラブソング中に出てくる「会話」というキーワードに対し，けんかをしている写真や恋人同士が語り合っているもの，大笑いしているものなどたくさんの画像候補の中から適切に恋人同士の画像を提示することは難しい．画像検索に関してはさまざまな研究が行われているので，画像がもつ意味を考慮した検索手法を取り入れることで，より検索の正当性を上げることができる．しかし，意外な画像が検索されることでかえって面白いという場合もあるため，本システムでは検索の正確さを追及することだけが目的ではないことを考慮して今後の拡張を行う必要がある．

ストーリー性の付加について

歌は1つのストーリーであることが多い．現在のシステムで抽出する画像は特にストーリーを意識せず，その場その場で合致するものを表示しているだけである．もともと歌詞自体がストーリーを表しているので，現在の手法でもある程度のストーリー性は実現できているといえるが，流れを意識した画像提示を行うことでよりストーリー性の高いシステムが実現できる．複数の提示画像候補があった場合に，映画の撮影手法などの映像理論を用いることでよりわかりやすい画像表示を行えることが示されており[15][20]，本システムにこれらの枠組みを適用することが有効である．

キーワードの付加について

本システムの前提として，マルチメディア素材にはキーワードなどのメタデータがあらかじめ付加されているとしているが，現状ではユーザが手入力でデータを付加する必要がある．しかし，たとえばデジタルカメラで撮影した画像に関しては，今後のデジタルカメラの高機能化により，撮影した地点や時間情報が自動的にメタデータとして付加されるようになる可能性は高い．そのような情報を用いれば，撮影した地名や天気などが明らかになるため，それらをキーワードとして利用できると考えられる．

関連研究

リアルタイムプレゼンテーションシステムの例としては，SuperSQLを用いたインタラクティブプレゼンテーション生成システムがある[57]．このシステムは，ユーザが要求をデータベース問合せの形で記述すると，データベース中の素材を用いてプレゼンテーションを生成するシステムであり，ECAルールを用いてデータベースの更新に対処するという点でアクティブカラオケと類似している．SuperSQLの質問文はデータベース問合せをベースとしているため「俳優Aが出演している映画を連続して見たい」といったような1つの問合せで直接的に表現できるシンプルなプレゼンテーション生成には適しているが，歌詞に沿って関連する画像を次々と提示するアクティブカラオケのような処理には適していない．また，このシステムにおけるECAルールの利用はデータベース内のデータ管理に限られているが，アクティブカラオケではデータ更新処理に加えてプレゼンテーションの演出効果を制御するためにECAルールを用いている．

カラオケの背景画像を提示するシステムとしては$\mathrm{VIP}_{KARAOKE}$がある[26]．$\mathrm{VIP}_{KARAOKE}$

は，ビデオカメラで取得したユーザの画像をカラオケの背景に重ねて表示するシステムで，画面上のユーザが，同じく画面上に表示されるアイコンに触れることで拍手音を鳴らすといった機能をもつ．$VIP_{KARAOKE}$はユーザの動きをもとにしたインタラクティブなシステムを実現しているが，イベントとなるのはユーザが画面上のオブジェクトに触れるという動作のみであるため，提供される機能はリモコンでも実現できるようなものに限られている．また，常にユーザが表示されているため，曲にあった背景表示は実現できていない．アクティブカラオケは曲に合った背景表示を実現しており，ルールを用いてさまざまな機能が実現できる．文献[53]は，カラオケの背景を3DCGのアニメーションで表現し，音楽と同期をとる方式を提案している．しかし，提案されているシステムはあらかじめ曲データに合わせた固定の動作をさせることしかできず，状況に応じて動作を変えるといった処理は行えない．

5.5 むすび

本章では，アクティブデータベースを用いたカラオケの背景作成システムアクティブカラオケの構築を行った．アクティブカラオケは，カラオケを盛り上げるための機構を組み込んでおり，アクティブカラオケを利用することで，よりカラオケを楽しめるようになる．ユーザは自分用のルールや画像を持ち歩くことで，いつでもどこでも自分用にカスタマイズされたカラオケシステムを利用できるようになり，カラオケ店は多数の面白いルールを客に提供することで他店との差別化を図れるようになる．

今後の課題を以下に示す．

- 高度な画像検索の実現

  画像のグルーピングを行なうことでグループ情報を用いた検索を行なう．また，モンタージュ理論など画像の前後関係を考慮した画像検索を行なう．

- 高度な画像提示の実現

  より多くのアクションを提供し，画像に多数のエフェクトをかけられるようにする．また，楽曲のテンポ情報やジャンル情報を提示に反映させ，さらに高度な画像提示機構を実現する．

- 平面画像以外の素材の利用

プロトタイプシステムでは背景素材に平面画像しか利用できないが，動画や3Dオブジェクトを扱えるようにする．

- 音声認識の利用

  現在の実装では，入力音声に関しては音量を取得しているだけである．もし，音声認識を利用できれば，歌詞を間違えたというイベントが実現できる．また，歌い手が勝手に替え歌を歌ったときにその替え歌に画像を追随させるなどの機能が実現できる．ただし，歌声の音声認識は非常に高度な技術であり，簡単に実現できる機能ではない．そこでまず歌声の音程を検出することで，音程がずれたというイベントを実現する予定である．

- その他のイベント・アクションの追加

  環境内に存在するセンサからの入力をイベントとして取得できるようにする．例えば動体センサを用いて，部屋の人たちが激しく動いていたら場の雰囲気が盛り上がっていると判断する機能を提供する．アクションとしては，曲の構成をダイナミックに変化させるアクションを提供し，「場が盛り上がっていないときには2番を飛ばしてエンディングへ」といったルールが記述できるようにする．その他にも，接続されたデバイスを制御するアクションや，ネットワークで接続された他のシステムに対してデータを送受信するアクションなどを実現する予定である．

- 他のプレゼンテーションシステムの構築

  本システムは，音楽に合わせて画像を検索するシステムである．今後はスライドショーなどの画像群に合わせて音楽を付加するシステムや，画像と音楽が格納されたデータベースを用いて自動的に環境ビデオを作成するシステムを構築していく予定である．

第6章

結論

6.1 本論文のまとめ

本論文では，アクティブデータベースの概念を用いたルールベースのサービス提供の枠組みについて述べた．マイクロエレクトロニクス技術の発展により，日常生活のあらゆる部分にコンピュータが入り込み，あらゆるものが電子化されるようになった現在，それらのコンピュータを統合的に取り扱ってユーザに柔軟なサービスを提供する枠組みが強く求められているといえる．

このような観点から，まず第1章では人々の要求を因果的に記述するためには，アクティブデータベースの動作記述言語であるECAルールが適していることを挙げ，アクティブデータベースをサービス提供の枠組みとして利用する際に検討しなければいけない事項について議論した．その結果，ECAルールを用いて柔軟なサービスを提供するためには，環境にあった言語仕様，処理系が必要であり，具体的な環境を想定しながらシステムの設計を行うという本論文の目的を明らかにした．

第2章ではモバイルコンピューティング環境のためのアクティブデータベースシステムであるAMDSについて述べた．モバイル環境では移動体の接続・切断など突発的な事象を取り扱うサービスに対する要求が大きく，ルールベースの枠組みを用いることでモバイル環境における多様なサービスを提供できることを示した．また，モバイル環境ではECAルールの構成が頻繁に変化するため，ルールの無限連鎖などの異常動作が起こる可能性がある．そこでこの章では，ルールの連鎖カウンタやルールの実行関係を表すグラフをモバイル環境に適用することで，異常動作を検出する枠組みを提案した．これらの機構を用いること

でモバイル環境においてルールベースシステムを安全に運用できるようになり，AMDSをモバイル環境におけるアプリケーションプラットフォームとして利用できる．

第3章では地理情報システムのためのアクティブデータベースであるActiveGISについて述べた．屋外でXML形式で標準化された地理情報が配信されている環境を想定し，ECAルールを用いて受信データを処理することでカスタマイズされた地図の作成や道案内などの位置依存サービスを提供できることを示した．ActiveGISにおけるECAルールであるECA-MLは，XML形式で記述されているため，地理情報記述G-XMLの記述中に含めて配信し，期限付きオブジェクトや受信者限定型のデータ配信などが実現できる．

第4章では放送環境のためのアクティブデータベースについて述べた．大量のデータが放送される環境では，必要なデータのみを抽出する情報フィルタリング技術が重要になる．この章ではユーザのフィルタリング要求をECAルールとして表現することで，柔軟な情報フィルタリングを実現するアクティブデータベースシステムSADBを提案した．SADBでは放送環境における大量のデータを処理できるように，データ受信イベントに対するルール処理の高速化を行っている．この章で構築した3つの応用システムは，それぞれECAルールのもつ処理の柔軟さを利用しており，環境の変化に合わせた処理手法の変更や，放送データごとの処理方法変更，ネットワーク状況にあわせた問合せ処理の変更など柔軟な放送データ処理を実現している．

第5章ではリアルタイムプレゼンテーション生成のためのアクティブデータベースシステムについて述べた．モバイル環境や放送環境の普及により，ユーザの端末には多数のデータが蓄積されるようになった．4章ではいかにデータを蓄積するかという点にECAルールを用いたが，この章では蓄積したデータからいかに必要なデータを抽出して提示するかという点に着目して言語の設計を行った．プロトタイプシステムであるアクティブカラオケは，データベース中に存在するマルチメディア素材に対し，曲の進行に合わせて適当な素材をリアルタイムで検索して提示するシステムであり，ECAルールを用いることで状況の変化に応じてプレゼンテーション内容を動的に変化させる機能を実現している．

6.2 統合型ルールベース環境へ向けて

2章や3章では，ユーザが携帯端末を持ち歩くモバイル環境におけるサービスを考えてきた．近年，さらなる計算機の小型化・軽量化によって，屋外でのコンピューティングスタイルはモバイルコンピューティングから，計算機を身に着けるウェアラブルコンピュー

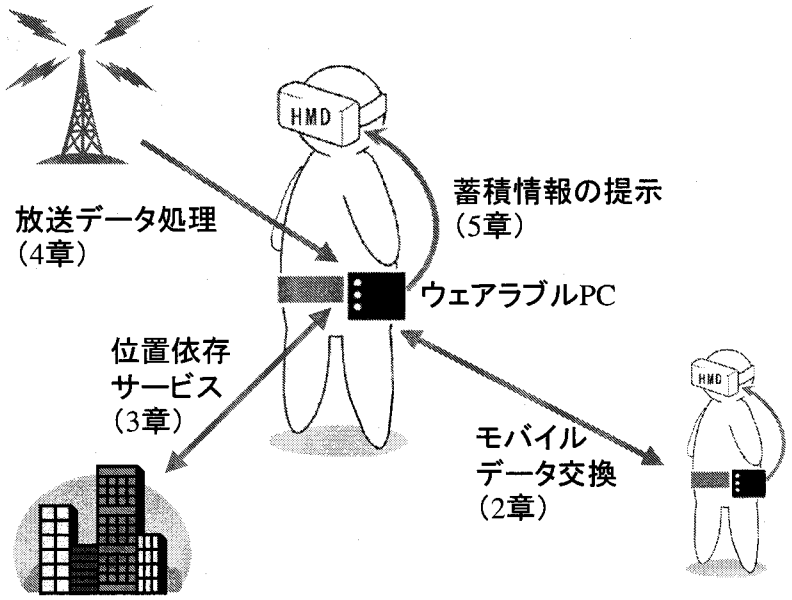


図 6.1: 統合型ルールベース環境

ティングへと移り変わると期待されている[16][36]. ウェアラブルコンピューティング環境では，常にユーザが計算機と共に移動し，情報は頭部に装着するヘッドマウントディスプレイ (HMD) を用いて閲覧する[24][67].

あらゆる人々がウェアラブルコンピュータを身に着けるようになったとき，求められる機能としては，ウェアラブルコンピュータを身につけた人同士での適切な情報交換機能や，放送型通信で配信される情報の取捨選択・蓄積機能であるといえる．また，ユーザは常にヘッドマウントディスプレイを見ているため，継続的に受信・蓄積するデータをいかにユーザに見せるかといった技術が重要になる[17][18][31]. これらの技術は，図6.1に示すように，筆者が本論文で述べたメカニズムを統合した環境であり，各章で述べたシステムを統合的に扱うことが重要になることを示している.

したがって，本研究の成果を統合し，統合システム内でECAルールを有機的に協調動作させることで，複雑な，あたかも知能をもっているかのような機能を実現することが今後の課題であるといえる.

ウェアラブルコンピューティングはそのサービスの多様性や日常性から，全業種を巻き込む実世界IT化の大きな流れのきっかけとなるものであり，近い将来には100兆円を超える市場規模を獲得すると期待されている．そのころには，これまでの章で述べたさまざま

なサービスが実現されており，紛失物を検索エンジンで検索したり，ジョギングしながら演奏するといった光景が当たり前になっている可能性がある．そのような状況において，本研究で提案したECAルールに基づくシステムが中心的な役割を果たしていることを期待したい．

ECAルールによる統合型サービス実現のために必要とされる今後の課題には次のようなものがある．

- ECAルールの統合

  本研究で述べたそれぞれのECAルール処理システムは別々の仕様をもっている．例えば，地理情報環境のためのシステムでは地理情報を取り扱うためにXML形式でルールが記述され，XML処理関連のアクションを持っている．一方，放送環境のためのアクティブデータベースでは独自フォーマットのルールを用い，放送に特化したシステム構成にすることで高速化を図っている．このように，各システムはルール仕様やシステム構成に違いがあり，それらのシステムを統合的に取り扱うために，下記のようなアプローチについて検討している．

  - メッセージレベルでの互換性確保

    言語仕様は異なるままで，メッセージの送受信のみ互換性を付加する．この場合，各システム間のコミュニケーションはメッセージを送信するSENDアクションと，メッセージを受信するRECEIVEイベントを処理するルール群によって行われる．このレベルでもある程度の統合サービスは提供可能である．

  - ルール変換

    メッセージレベルの互換性では，実現できない統合サービスがある．例えば，ウェアラブルコンピュータ上で動作していたECAルールを，負荷軽減のための環境内に配置された移動体サーバ上に移動させて，結果だけを受け取るような機構はメッセージレベルの互換性では実現できない．しかし，ECAルールの仕様をすべて統一することは，各システムやデバイスのスペックや目的の違いから現実的ではない．そこで，ウェアラブルコンピュータと移動体サーバなどシステムが異なる接点となる部分にゲートウェイを設置し，ルールの変換を行う．ウェアラブルコンピュータ上で動作する複雑なルールで実現している機能を環境側に移動するときには，ゲートウェイ上のコンバータが環境側のコンピュータで

動作するルール群に変換し，自動的に配置する．このような自動変換を実現するためには，変換アルゴリズムや適切な端末を選んでルールを送り込む機構などさまざまな機能が必要になる．

- イベントやアクションの動的追加機能の実現

  ウェアラブル環境では，ユーザの情報を取得するためにさまざまなセンサデバイスを身に着けることを想定している．例えば3章で述べた地理情報サービスにおいて，現在ではGPSを用いた位置データしか用いていないが，地磁気センサを用いてユーザが向いている方向を取得したり，カメラデバイスを用いて現在の状況を記録するといった機能があればより便利になる．このように，新たなデバイスを用いるためには対応するイベント記述・アクション記述を追加する必要があるため，システム全体に渡る修正を加えなくても言語仕様が拡張できる機構が必要となる．そこで，プラグイン形式を用いて動的にシステム構成や言語仕様を拡張できる方式について現在検討している [37][38][42][68].

- システムの安全性とアクセス制御

  ECAルールを送りあうことは，プログラムを送りあうことと同様であるため，安全を確保することは重要である．特にウェアラブルコンピュータには，ユーザの個人情報やセンサからのユーザ情報が蓄積されているため，容易に外部からアクセスされることを防ぐ必要がある．そこで，ルールを受け取るときに自動的にルール内部を解析し，ユーザの個人情報にアクセスするかどうかなど，ルールの安全性を判別してシステムに格納するかどうかを決める手法について検討している．

- 処理機構の改善

  ウェアラブルコンピュータは常に電源が入っていることを想定している．バッテリの持続時間に関しては，燃料電池の普及などにより改善する可能性が高いが，システム単体でもできるだけ消費電力を抑えてバッテリの持続時間を長くすることが重要である．そこで，ウェアラブルコンピュータにおいて，使用する機能に応じて付加デバイスの電源を制御するなど消費電力を抑える機構を提案する．

- 開発環境の提供

  ECAルールはユーザの要求を直感的に記述できるとはいえ，一般ユーザがテキストエディタを使ってそのまま記述するのは困難である．そこで，ルールベース環境向

けにグラフィカルなエディタ，シミュレータ，デバッガなどの各種ツール群を用意し，開発者および一般ユーザがECAルールを用いてさまざまなサービスを記述したりカスタマイズしたりできるようにする．

- 他分野への適用

  本研究ではこれまで，モバイル・地理情報・放送などさまざまな分野にECAルールのメカニズムを応用してきたが，今回取り組んだ分野だけでは人々の要求すべてをカバーすることはできない．例えば，仮想現実や拡張現実と呼ばれる研究分野では，ヘッドマウントディスプレイに仮想世界を提示したり，ディスプレイから見える現実空間にアノテーションを付加するといった目的で使用できる．このような仮想オブジェクト処理や重ねあわせ処理にルールベースの枠組みを適用することで，より個人化された便利なシステムが構築できると考えている．

謝辞

本研究全般に関して，懇切なる御指導，御助言と格別なる御配慮を賜りました大阪大学大学院情報科学研究科マルチメディア工学専攻 西尾章治郎教授に謹んで深謝の意を表します．西尾教授は私の勤務先である大阪大学サイバーメディアセンターのセンター長を兼任されており，仕事・研究の両面にわたって多大なご支援を頂きました．厚く御礼申し上げます．

本論文をまとめるにあたり，貴重な時間を割いて頂き，懇切なる御指導と有益な御助言を賜りました大阪大学大学院情報科学研究科マルチメディア工学専攻 薦田憲久教授，岸野文郎教授に心より感謝致します．

講義・学生生活を通じて，学問に取り組む姿勢をご教授頂きました大阪大学大学院情報科学研究科情報ネットワーク学専攻 村上孝三教授，同情報システム工学専攻 藤岡弘教授，同バイオ情報工学専攻 赤澤堅造教授，大阪大学サイバーメディアセンター 下條真司教授，および白川功名誉教授に厚く感謝申し上げます．

大阪大学サイバーメディアセンターサイバーコミュニティ部門 吉田勝行教授には，懇切なる討論と御指導を頂き，また様々な面でお世話になりました．ここに深謝致します．

大阪大学情報科学研究科マルチメディア工学専攻 塚本昌彦助教授には，本研究のすべてに関して，直接の御指導を頂き，研究の進め方や論文の書き方など研究生活の全面に渡って多大なるご支援を頂きました．心から感謝いたします．

本研究において，有益な御指導，御討論を頂いた大学大学サイバーメディアセンター 春本要講師，惜しみない御助言，研究活動を進めるにあたっての多大なる御支援を頂いた大阪大学大学院情報科学研究科マルチメディア工学専攻 原隆浩助手に深謝致します．また，本研究を進める上で有益な御討論を頂いた大阪大学サイバーメディアセンター 秋山豊和助手，小川剛史助手，住友電気工業株式会社 村瀬亨氏に感謝致します．

情報フィルタリング技術について共に研究を進め，有益な議論および多大なご協力を頂

いたSomnuk Sanguantrakul氏，Loh Yin Huei氏，大阪大学大学院情報科学研究科マルチメディア工学専攻 澤井里枝氏，小寺拓也氏に深謝致します．また，本研究をともに進め御協力いただいた東芝ITソリューション株式会社 加下雅一氏に感謝致します．

本研究に関して有益な御討論を頂いた大阪大学大学院情報科学研究科マルチメディア工学専攻西尾研究室の諸氏および私の研究活動をいろいろな面でご支援いただいた多くの方々に厚く御礼申し上げます．

最後に，研究生活を送る上で，暖かい御支援と多大なる御理解を頂いた両親，そして本研究を陰で支え，応援してくれた妻 朋美に心からの感謝と御礼を申し上げます．

参考文献

[1] S. Acharya, M. Franklin, and S. Zdonik: Broadcast Disks: Data Management for Asymmetric Communication Environments, *in Proc. of ACM SIGMOD International Conference on Management of Data*, pp. 199–210 (1995).

[2] A. Aiken, J. Widom, and J. M. Hellerstein: Behavior of database production rules: Termination confluence, and observable determinism, *in Proc. of ACM SIGMOD International Conference on the Management of Data*, pp. 59–68 (1992).

[3] D. Aksoy and M. Franklin: Scheduling for Large-Scale On-Demand Data Broadcasting, *in Proc. of IEEE INFOCOM*, pp. 651–659 (1998).

[4] E. Baralis, S. Ceri, and S. Paraboschi: Run-Time Detection of Non-Terminating Active Rule Systems, *in Proc. of Fourth International Conference on Deductive and Object-Oriented Databases (DOOD'95)*, pp. 59–68 (1995).

[5] E. Baralis and J. Widom: An Algebraic Approach to Rule Analysis in Expert Database Systems, *in Proc. of the 20th VLDB Conference*, pp. 475–486 (1994).

[6] N. J. Belkin and W. B. Croft: Information filtering and information retrieval: two sides of the same coin?, *Communications of ACM*, Vol. 35, No. 12, pp. 29–38 (1992).

[7] S. Ceri and J. Widom: Deriving Production Rules for Constraint Maintenance, *in Proc. of the 16th VLDB Conference*, pp. 566–577 (1990).

[8] S. Chakravarthy, B. Blaustein, A. Buchmann, M. Carey, U. Dayal, D. Goldhirsch, M. Hsu, R. Jauhari, R. Ladin, M. Livny, D. McCarthy, R. McKee, and A. Rosen-

thal: HiPAC: A research project in active, time-constrained database management, *Technical Report XAIT-89-02*, Xerox Advanced Information Technology (1989).

[9] O. Diaz, A. Jaime, and G. al Qaimari: Supporting Dynamic Displays Using Active Rules, *ACM Sigmod Record*, Vol. 23, No. 1, pp. 21–26 (1994).

[10] 道路用Web記述言語RWMLの提案: http://www2.ceri.go.jp/its-win/RWML.htm.

[11] N. Gehani and H. V. Jagadish: Ode as an active database: constraints and triggers, *in Proc. of the 17th VLDB Conference*, pp. 327–336 (1991).

[12] G-XMLホームページ: http://gisclh.dpc.or.jp/gxml/contents/.

[13] 箱守 聰, 田辺雅則, 石川裕治, 井上 潮: 放送型通信／オンデマンド型通信を統合した情報提供システム, 情報処理学会研究報告, Vol. 34, No. 8, pp. 55–60 (1997).

[14] E. N. Hanson: Rule condition testing and action execution in Ariel, *in Proc. of the ACM SIGMOD International Conference on Management of Data*, pp. 49–58 (1992).

[15] 平石絢子, 井上亮文, 重野 寛, 岡田謙一, 松下 温: 映画の撮影手法に基づいた会議の自動撮影, マルチメディア, 分散, 協調とモバイル(DICOMO2002)シンポジウム論文集, pp. 285–288 (2002).

[16] 廣瀬通孝: ウェアラブル・コンピュータの展開, 情報処理, Vol. 40, No. 9, pp. 873–877 (1999).

[17] T. Hollerer, S. Feiner, and J. Pavlik: Situated Documentaries: Embedding Multimedia Presentations in the Real World, *in Proc. of 3rd International Symposium on Wearable Computers (ISWC'99)*, pp. 79–86 (1999).

[18] T. Hollerer, S. Feiner, T. Terauchi, G. Rashid, and D. Hallaway: Exploring MARS: Developing Indoor and Outdoor User Interfaces to a Mobile Augmented Reality System, *Computers and Graphics*, Vol. 23, No. 6, pp. 779–785 (1999).

[19] Q. Hu, D. Lee, and W. Lee: Performance evaluation of a wireless hierarchical data dissemination system, *in Proc. of ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99)*, pp. 163–173 (1999).

[20] 井上智雄, 岡田謙一, 松下 温: テレビ番組のカメラワークの知識に基づいたTV会議システム, 情報処理学会論文誌, Vol. 37, No. 11, pp. 19–24 (1996).

[21] Y. E. Ioannidis and T. K. Sellis: Conflict Resolution of Rules Assigning Values to Virtual Attributes, *in Proc. of the ACM SIGMOD International Conference on Management of Data*, pp.205–214 (1989).

[22] H. Ishikawa and K. Kubota: An active object-oriented database: a multi-paradigm approach to constraint management, *in Proc. of the 19th VLDB Conference*, pp. 467–478 (1993).

[23] 石川 博: アクティブデータベース, 情報処理, Vol. 35, No. 2, pp. 120–129 (1994).

[24] 板生 清: ウェアラブル情報機器の実際, オプトロニクス社 (1999).

[25] 伊藤 悟, 岡部篤行, 奥貫圭一, 東明佐久良, 秋田義一, 小坪宏則, 大喜多祐司, 後藤 寛, 金子忠明, 足立俊雅, エリックバーズリー: 都市計画基礎調査におけるモバイルGIS利用の試み(その1), 地理情報システム学会講演論文集, No. 7, pp. 137–140 (1998).

[26] 岩田憲治: カラオケはマルチメディアの実験場, BIT3月号, 共立出版, pp. 16–24 (1994).

[27] カント(篠田英雄訳): 純粋理性批判(上), 岩波文庫 (1961).

[28] A. P. Karadimce and S. D. Urban: Refined Trigger Graphs: A Logic-Based Approach to Termination Analysis in an Active Object-Oriented Database, *in Proc. of the Twelfth International Conference on Data Engineering (ICDE'96)*, pp. 384–391 (1996).

[29] 加下雅一, 寺田 努, 原 隆浩, 塚本昌彦, 西尾章治郎: データベース放送システムのためのサーバと移動型クライアントによる協調型問合せ処理方式, 情報処理学会論文誌 : データベース (2003).

[30] M. Kashita, T. Terada, T. Hara, M. Tsukamoto, and S. Nishio: An Adaptive Query Processing Method according to System Environments in Database Broadcasting Systems, *in Proc. of ISCA 18th International Conference on Computers and Their Applications (CATA 2003)*, pp. 174–179 (2003).

[31] T. Kurata, T. Okuma, M. Kourogi, T. Kato, and K. Sakaue: VizWear: Toward Human-Centered Interaction through Wearable Vision and Visualization, *in Proc. of IEEE Pacific Rim Conference on Multimedia (PCM2001)*, pp. 40–47 (2001).

[32] S. Y. Lee and T. W. Ling: A Path Removing Technique for Detecting Trigger Termination, *in Proc. of 6th International Conference on Extending Database Technology (EDBT'98)*, pp. 341–355 (1998).

[33] S. Y. Lee and T. W. Ling: Unrolling Cycle to Decide Trigger Termination, *in Proc. of 25th VLDB Conference*, pp. 483–493 (1999).

[34] F. Llirbat, F. Fabret and E. Simon: Eliminating Costly Redundant Computations from SQL Trigger Executions, *in Proc. of the ACM SIGMOD International Conference on Management of Data*, pp.428–439 (1997).

[35] S. Mann: Wearable Computing: A First Step Toward Personal Imaging, *IEEE Computer*, Vol. 30, No. 2, pp. 25–32 (1997).

[36] S. Mann: Definition of 'Wearable Computer', *Keynote Address for the First International Conference on Wearable Computing (ICWC98)* (1998).

[37] 三浦直樹，宮前雅一，寺田 努，塚本昌彦，西尾章治郎: Aware-Mail: ウェアラブルコンピューティング環境のためのイベント駆動型メールシステム, 第65回情報処理学会全国大会講演論文集 (5) pp. 207–210 (2003).

[38] 宮前雅一，中村聡史，寺田 努，塚本昌彦，西尾章治郎: ウェアラブルコンピューティングのための拡張可能なルール処理システム, 情報処理学会研究報告 (情報家電コンピューティング研究グループ 2002-IAC-3), Vol. 2002, pp. 41–46 (2002).

[39] J. Mostafa, S. Mukhopadhyay, W. Lam, and M. Palakal: A multilevel approach to intelligent information filtering: model, system, and evaluation, *ACM Transaction on Information Systems*, Vol. 15, No. 4, pp. 368–399 (1997).

[40] T. Murase, M. Tsukamoto and S. Nishio: Active Mobile Database Systems for Mobile Computing Environments, *IEICE Transaction on Infomation and Systems*, Vol. E81-D, No. 5, pp. 427–433 (1998).

[41] 村瀬 亨，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースの安全性解析手法, 情報処理学会論文誌, Vol. 43, No. 12, pp. 3785–3793 (2002).

[42] 中村聡史，宮前雅一，寺田 努，塚本昌彦，柳瀬康宏，釣 裕美，堀 雅和，西尾章治郎: ウェアラブルコンピューティングのためのルール処理システムを用いたサービス, 第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 217–218 (2002).

[43] J. L. Oliveira, C. Q. Cunha, and G. C. Magalhaes: Object Model for Dynamic Construction of Visual Interfaces, *in Proc. of 9th Brasilian Symposium on Software Engineering*, pp. 143–158 (1995).

[44] J. L. Oliveira, C. B. Medeiros, and M. A. Cilia: Active Customization of GIS User Interfaces, *in Proc. of 13th International Conference on DATA ENGINEERING*, pp. 487–496 (1997).

[45] Open GIS Consortium, Inc: http://www.opengis.org/.

[46] Point Of Interest eXchange Language: http://www.w3.org/TR/poix/.

[47] S. Sanguantrakul, T. Terada, M. Tsukamoto, S. Nishio, K. Miura, S. Matsuura, and T. Imanaka: User Customized Classification and Selection for Broadcast Data, *in Proc. of Semantic Issues in Multimedia Systems, IFIP TC-2 Working Conference*, pp. 596–601 (1999).

[48] 澤井里枝, 寺田 努, 塚本昌彦, 西尾章治郎: フィルタリングSQL: フィルタリングのためのユーザ要求記述言語, 電気情報通信学会第11回データ工学ワークショップ (DEWS2000) 論文集 (CD-ROM) (2000).

[49] 澤井里枝，塚本昌彦，寺田 努，Loh Yin Huei，西尾章治郎: 情報フィルタリングの関数的性質について, 電子情報通信学会論文誌，Vol. J85-D-I, No. 10, pp. 939–950 (2002).

[50] 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: フィルタリング関数におけるセレクションとランキングについて, 情報処理学会論文誌 : データベース，Vol.43, No. SIG12(TOD16), pp. 80–91 (2002).

[51] 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: 合成フィルタリング関数の性質について，情報処理学会論文誌：データベース，Vol. 44，No. SIG3(TOD17)，pp. 43–53 (2003).

[52] 澤井里枝，塚本昌彦，寺田 努，西尾章治郎: 情報フィルタリングの実行順序に関する関数的性質について，情報処理学会論文誌：データベース，Vol. 44，No. SIG3(TOD17)，pp. 54–64 (2003).

[53] 白井 豊，林 克彦，徳永真志，鈴木誠一，大久保晴代: 3次元ＣＧを用いた通信カラオケの開発，情報処理学会研究報告 グラフィックスとCAD 90-7, Vol. 98, No. 90, pp. 37–42 (1998).

[54] J. スター，J. エステス著，岡部篤行，貞広幸雄，今井修共訳: 入門 地理情報システム，共立出版 (1992).

[55] M. Stonebraker and G. Kemnitz: The POSTGRES nextgeneration database management system, *Communications of the ACM*, Vol. 34, No. 10, pp. 78–92 (1991).

[56] 垂水浩幸，森下 健，中尾 恵，上林弥彦: 時空間限定型オブジェクトシステム: SpaceTag，日本ソフトウェア科学会第6回インタラクティブシステムとソフトウェアに関するワークショップ (WISS'98), pp. 1–9 (1998).

[57] 多田光伸，遠山元道: SuperSQLによるインタラクティブプレゼンテーションの自動生成，情報処理学会研究報告 (2001-DBS-125), Vol. 2001, No. 125, pp. 159–166 (2001).

[58] 高橋克巳: モバイル環境下での情報収集を支援するエージェント，人工知能学会誌，Vol. 14, No. 4, pp. 14–21 (1999).

[59] 寺田 努，莫 君，村瀬 亨，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースのECAルール実行監視機構の設計と実装，情報処理学会研究報告 (データベースシステム研究会 99-DBS-119)，Vol. 99，No. 7，pp. 369–374 (1999).

[60] 寺田 努，塚本昌彦，西尾章治郎: G-XMLをサポートするアクティブデータベースシステム，情報処理学会研究報告 (モバイルコンピューティングとワイヤレス通信研究会 2000-MBL-14)，Vol. 2000，No. 87，pp. 123–130 (2000).

[61] 寺田 努，塚本昌彦，西尾章治郎: カラオケの背景を動的に作成するアクティブデータベースシステム, 第61回情報処理学会全国大会 (2000).

[62] 寺田 努，塚本昌彦，西尾章治郎: アクティブデータベースを用いた地理情報システム, 情報処理学会論文誌，Vol. 41，No. 11，pp. 3103–3113 (2000).

[63] 寺田 努，塚本昌彦，西尾章治郎: 放送型データ受信のためのアクティブデータベースの設計と実装, 電子情報通信学会論文誌，Vol. J83-D-I，No. 12，pp. 1272–1283 (2000).

[64] 寺田 努，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースのシミュレーション環境について, 情報処理学会研究報告 (データベースシステム研究会 2001-DBS-125(1))，Vol. 2001，No. 70，pp. 351–358 (2001).

[65] 寺田 努，塚本昌彦，西尾章治郎: 移動体計算環境におけるアクティブデータベースの動的トリガグラフ構築機構の設計と実装, 情報処理学会論文誌：データベース，Vol. 43, No. SIG12(TOD16), pp. 52–63 (2002).

[66] 寺田 努，塚本昌彦，西尾章治郎: アクティブデータベースを用いたカラオケの背景作成システム, 情報処理学会論文誌，Vol. 44, No. 2, pp. 235–244 (2003).

[67] 塚本昌彦: モバイルコンピューティング，岩波科学ライブラリ77, 岩波書店 (1999).

[68] 塚本昌彦，寺田 努，堀 雅和: ウェアラブルコンピューティングのためのシステム基盤, 第1回情報科学技術フォーラム (FIT2002) 論文集第4分冊，pp. 213–214 (2002).

[69] A. Vaduva, S. Gatziu, and K. Dittrich: Investigating Termination in Active Database Systems with Expressive Rule Languages, *in Proc. of Rules in Database System '97 (LNCS 1312)*, pp. 149–164 (1997).

[70] L. van der Voort, and A. Siebes: Termination and confluence of rule execution, *in Proc. of 2nd International Conference on Information and Knowledge Management*, pp. 245–255 (1993).

[71] M. Weiser: The Computer for the Twenty-first Century, *Scientific American*, Vol. 265, No. 3, pp. 94–104 (1991).

[72] P. Wellner, E. Machay, R. Gold: Computer-Augmented Environments: Back To The Real World, *Communications of the ACM*, Vol. 36, No. 7, pp. 24–97 (1993).

[73] J. Widom, R. J. Cocheane and B. G. Lindsay: Implementing set-oriented production rules as an extension to Starburst, *in Proc. of the 17th VLDB Conference*, pp. 275–285 (1991).

[74] J. Widom and S. Ceri: ACTIVE DATABASE SYSTEMS, Morgan Kaufmann Publishers, Inc. (1996).

[75] E. Yajima, T. Hara, M. Tsukamoto, and S. Nishio: Scheduling Strategies of Correlated Data in Push-Based Systems, *Information Systems and Operational Research (INFOR)*, pp. 152–173 (2001).

[76] 全国カラオケ事業者協会ホームページ: http://www.japan-karaoke.com/.