卒業論文　　1999年度(平成11年度)

情報家電ネットワークに適したイベント通知機構の構築

指導教員

慶應義塾大学環境情報学部

徳田　英幸

村井　純

楠本　博之

中村　修

慶應義塾大学　環境情報学部

岩井 将行

tailor@sfc.wide.ad.jp

卒業論文要旨　　1999年度(平成11年度)

情報家電ネットワークに適したイベント通知機構の構築

本論文では，通信インタフェースを内蔵した情報家電機器間の相互利用を，動的分散イベント通知モデルを提案する事によって実現した．既存の家電機器の多くは，独立した機器として使用されるように設計されていた．しかし近年，家庭内へのネットワークの導入により，単体で動作する家電機器に代わり，ネットワークを利用してユーザが操作可能な情報家電機器が登場し始めた．本論文は，そのような情報家電機器間が相互接続している環境を情報家電ネットワークと定義する．現在， 情報家電ネットワークにおいて情報家電機器間の相互利用が，パソコンとその周辺機器といった集中管理でしか活用されておらず，多くの情報家電機器間は，いまだ相互の活用がなされていない．その問題点を３つ挙げる．

第一に情報家電ネットワークでは，異なるメーカや異なるOSの機器間の通信規格が定まっておらず，機器間の情報のやり取りが実現されていない．

第二に，機器間の情報のやり取りを行うプログラムを機器の埋め込む場合には，開発にコストが生じる．

第三に，ユーザが機器間を結びつける場合，複雑な設定を要求される．

上記の問題点を解決し，より多くの情報家電機器を相互に利用する枠組みを提供するためには，ネットワーク障害への対処，未知の情報家電機器の検索への対応，情報家電間の組み合わせを動的に変更できる動的変更機能の実現が必要である．本論文では，上記の機能をすべて満たす分散プログラムモデルである動的分散イベント通知モデルKeio Media Space Family-Dynamically RedirectionAble Graphical Operating eNvironment/Event Model(kmsf-DRAGON/EM)を提案する．また，同イベント通知モデルに基づいたシステムを構築するためのツールキットKeio Media Space Family-Dynamically RedirectionAble Graphical Operating eNvironment/Tool Kit(kmsf-DRAGON/TK)を提供する．このツールキットにより，その動的分散イベント通知モデルを実装した情報家電機器を容易に開発できる．本ツールキットを用いたシステムでは，情報家電機器のネットワーク上の移動に対処できる，またそれらの機器に動的に通知先をユーザが動的に変更できる．さらに，本ツールキットでは，ユーザが容易にイベントの通知先を変更できるGraphical User Interface(GUI)プログラムを提供する．本論文で提案するkmsf-DRAGON/EMおよび，kmsf-DRAGON/TKにより，情報家電機器間を動的に組み合わせ情報家電の家庭内での役割を広めるられる．

慶應義塾大学　環境情報学部
岩井 将行

Abstract of Bachelor's Thesis

Dynamic Distributed Redirectionable Event Model for Intelligent Home Appliances Network

In this research , mutual communication mechanism of networked appliances with a built-in communication interface is proposed ,designed and implemented. Many of typical appliances have been designed to be used alone. However as the network spreads over the home, the networked appliances that can be controlled through network have appeared in place of the appliances that are alone. In this research the environment in which networked appliances are connected each other is defined as intelligent appliance network. The present time mutual communication of networked appliances is done only as the concentrated administration by personal computers. Many of networked appliances are not used automatically. There are three problems. First ,there is few communication protocols between vendors or devices. Second ,it costs much to implement the program which communicates between the networked appliances is implemented. Third ,the set up of connection between appliances is difficult for many users. To solve the problems above and to provide the structure in which more networked appliances are connected each other, network fault obstacle, searching of unknown networked appliances, and function of dynamically changing the combination of each networked appliances are needed. In this paper, distributed dynamic redirectionable event model named kmsf-DRAGON/TKwhich fulfills all the requirements above is proposed, and the toolkit which constructs this redirectionable event model is described. This toolkit makes it easy to develop networked appliances that implemented distributed dynamic event model.Furthermore this toolkit provides Graphical User Interface(GUI) that enables users to to redirect the events. kmsf-DRAGON/TKenables the function of networked appliances.

Masayuki Iwai

Faculty of Environmental Information
Keio University

目 次

第 1 章 序論 **1**
1.1 本研究の背景 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 1
1.1.1 家電機器の変遷 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 1
1.1.2 通信ネットワークの変遷 ・・・・・・・・・・・・・・・・・・・・・・・・ 1
1.2 問題意識 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 2
1.3 本研究の目的 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 4
1.3.1 各機器を結ぶイベント通知モデルの提案 ・・・・・・・・・・・・・・・・ 5
1.3.2 ツールキットの提案 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 5
1.4 本論文の構成 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 6

第 2 章 情報家電ネットワーク **7**
2.1 情報家電ネットワークの概要 ・・・・・・・・・・・・・・・・・・・・・・・・ 8
2.1.1 用語の定義 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 8
2.1.2 情報家電ネットワークの動向 ・・・・・・・・・・・・・・・・・・・・・・ 9
2.2 情報家電ネットワークの特徴 ・・・・・・・・・・・・・・・・・・・・・・・・ 10
2.3 必要な機能 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 11
2.4 本章のまとめ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 13

第 3 章 動的分散イベント通知モデル (kmsf-DRAGON/EM) **14**
3.1 イベントモデルの分類 ・・・・・・・・・・・・・・・・・・・・・・・・・・・ 15
3.2 イベント通知モデル ・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 15
3.2.1 非登録型イベント通知モデル ・・・・・・・・・・・・・・・・・・・・・・ 15
3.2.2 登録型イベント通知モデル ・・・・・・・・・・・・・・・・・・・・・・・ 16
3.3 イベント問い合わせモデル ・・・・・・・・・・・・・・・・・・・・・・・・・ 17
3.3.1 非登録型イベント問い合わせモデル ・・・・・・・・・・・・・・・・・・ 17
3.3.2 登録型イベント問い合わせモデル ・・・・・・・・・・・・・・・・・・・ 17
3.4 イベント通知モデルに関する既存の研究 ・・・・・・・・・・・・・・・・・・ 19
3.4.1 ECJ イベントモデル ・・・・・・・・・・・・・・・・・・・・・・・・・・ 19
3.4.2 JavaBeans ローカルイベントモデル ・・・・・・・・・・・・・・・・・・ 20
3.4.3 Jini 分散イベントモデル ・・・・・・・・・・・・・・・・・・・・・・・・ 21
3.4.4 関連研究に対する共通した問題点 ・・・・・・・・・・・・・・・・・・・ 21

3.5 動的分散イベントモデルの提案 ・・・・・・・・・・・・・・・・・・・・・・・・ 22
3.6 動的分散イベントモデルの手順 ・・・・・・・・・・・・・・・・・・・・・・・・ 22
3.7 機能 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 24
3.8 本章のまとめ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 25

第4章 kmsf-DRAGON/TKの設計 **26**
4.1 kmsf-DRAGON/TKの概要 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 27
4.2 設計方針 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 27
4.3 全体構成 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 29
4.3.1 ハードウエア構成 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 29
4.3.2 ソフトウエア構成 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 30
4.4 各システム設計 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 30
4.5 動作手順 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 33
4.6 本章のまとめ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 37

第5章 実装 **38**
5.1 実装環境 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 39
5.2 kmsf-DRAGON/TK： モジュール部 ・・・・・・・・・・・・・・・・・・・・ 39
5.2.1 OneTemplateWatcherクラス ・・・・・・・・・・・・・・・・・・・・ 40
5.2.2 ServiceUpperクラス ・・・・・・・・・・・・・・・・・・・・・・・・ 40
5.2.3 EventProducerImplクラス ・・・・・・・・・・・・・・・・・・・・・ 40
5.2.4 BaseBinderUIクラス ・・・・・・・・・・・・・・・・・・・・・・・・ 41
5.2.5 BaseBinderClientクラス ・・・・・・・・・・・・・・・・・・・・・・ 42
5.2.6 ProducerProxyクラス ・・・・・・・・・・・・・・・・・・・・・・・ 42
5.2.7 ConsumerProxyクラス ・・・・・・・・・・・・・・・・・・・・・・・ 43
5.2.8 EventBinderクラス ・・・・・・・・・・・・・・・・・・・・・・・・・ 43
5.2.9 BindingAppletGUIクラス ・・・・・・・・・・・・・・・・・・・・・・ 43
5.2.10 BindingAppletClientクラス ・・・・・・・・・・・・・・・・・・・・ 44
5.3 パッケージを用いたシステムの実装例 ・・・・・・・・・・・・・・・・・・ 44
5.4 本章のまとめ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 46

第6章 評価 **49**
6.1 定量的評価 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 50
6.1.1 イベント伝搬にかかる時間の測定 ・・・・・・・・・・・・・・・・・・ 50
6.1.2 イベント通知先切り替えにかかる時間の測定 ・・・・・・・・・・・・・ 52
6.2 定性的評価 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 54
6.3 本章のまとめ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 54

第7章 結論 **56**
7.1 今後の課題 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 57
7.2 まとめ ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 60

図 目 次

1.1 主要通信メディアの世帯普及率10%までの所要時間 ・・・・・・・・・・・ 2
1.2 情報家電機器間の連携 ・・・・・・・・・・・・・・・・・・・・・・・・・ 4
1.3 イベント通知先動的変更の例 ・・・・・・・・・・・・・・・・・・・・・・ 5

3.1 非登録型イベント通知モデル ・・・・・・・・・・・・・・・・・・・・・・ 15
3.2 イベント通知とイベント登録 ・・・・・・・・・・・・・・・・・・・・・・ 16
3.3 登録型イベント通知モデル ・・・・・・・・・・・・・・・・・・・・・・・ 16
3.4 非登録型イベント問い合わせモデル ・・・・・・・・・・・・・・・・・・・ 17
3.5 登録型イベント問い合わせモデル ・・・・・・・・・・・・・・・・・・・・ 18
3.6 ECJの概念 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 20
3.7 Jini Distribute Event Modelの概要 ・・・・・・・・・・・・・・・・・・・ 22
3.8 動的分散イベント通知モデルの概念 ・・・・・・・・・・・・・・・・・・・ 23
3.9 通知先の動的変更 ・・・・・・・・・・・・・・・・・・・・・・・・・・・ 25

4.1 ハードウエアの構成 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 29
4.2 システム全体の概要 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 30
4.3 Remote Event Objectの保持する内容 ・・・・・・・・・・・・・・・・・・ 31
4.4 Producer Proxyの詳細な動作手順 ・・・・・・・・・・・・・・・・・・・・ 35
4.5 Consumer Proxy Serviceの動作手順 ・・・・・・・・・・・・・・・・・・・ 35
4.6 Event Binder の動作手順 ・・・・・・・・・・・・・・・・・・・・・・・・ 36

5.1 パッケージDragonTKの構成 ・・・・・・・・・・・・・・・・・・・・・・・ 39
5.2 OneTemplateWatcherクラスの状態遷移 ・・・・・・・・・・・・・・・・・・ 41
5.3 ServiceUpperクラスの状態遷移 ・・・・・・・・・・・・・・・・・・・・・ 42
5.4 ProducerProxyのクラス構成図 ・・・・・・・・・・・・・・・・・・・・・ 43
5.5 ConsumerProxyのクラス構成図 ・・・・・・・・・・・・・・・・・・・・・ 43
5.6 EventBinderのクラス構成図 ・・・・・・・・・・・・・・・・・・・・・・ 44
5.7 BindingAppletGUIのクラス構成図 ・・・・・・・・・・・・・・・・・・・・ 44
5.8 BindingAppletClientのクラス構成図 ・・・・・・・・・・・・・・・・・・・ 45
5.9 KeyInputEventProducerの実装 ・・・・・・・・・・・・・・・・・・・・・ 46
5.10 KeyInputEventConsumerの実装 ・・・・・・・・・・・・・・・・・・・・・ 47
5.11 BindingAppletGUIの画面 ・・・・・・・・・・・・・・・・・・・・・・・・ 48

6.1 複数のイベント消費サービスにイベント通知中の画面 · · · · · · · · · · · 50
6.2 通知時間測定結果 · 52
6.3 各区間の説明 · 52
6.4 切り替え時間測定結果 · 53

7.1 Event Generator Serviceの概要 · 59
7.2 イベントメールボックスの概要 · 60
7.3 イベントフィルタボックスの概要 · 61

表 目 次

1.1 ネットワークの形態の広がり ・・・・・・・・・・・・・・・・・・・・・・・・ 3

2.1 キャンパスネットワークと情報家電ネットワークの特徴比較 ・・・・・・ 10

3.1 アプローチ方法の比較 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 19
3.2 ローカルイベントモデルと分散イベントモデルの違い ・・・・・・・・・・ 21
3.3 各のイベント通知モデルのとの評価 ・・・・・・・・・・・・・・・・・・・ 22

4.1 サービスの特定方法 ・・・・・・・・・・・・・・・・・・・・・・・・・・・ 28
4.2 通知先変更命令 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 34
4.3 通知先変更命令の一例 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 34

6.1 測定環境・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 51
6.2 通知時間測定結果・・・・・・・・・・・・・・・・・・・・・・・・・・・・ 51
6.3 切り替え時間測定結果 ・・・・・・・・・・・・・・・・・・・・・・・・・・ 53
6.4 関連するイベントモデルとの機能比較 ・・・・・・・・・・・・・・・・・・ 55

第 1 章

序論

1.1 本研究の背景

家庭に存在する家電機器は多種多様であるが，主にそれらは個々で独立して動作するように設計されている．しかし今日，ネットワーク接続機能を備えた家電機器が生産されるようになった．また，通信ネットワーク形態の多様化に伴い，一般的に家庭内で用いられるネットワークの形態もEthernet，IEEE1394[6]などと多様化している．

1.1.1 家電機器の変遷

近年家電機器は，大きな普及を果たしている．わが国における 1998 年時点の全世帯に対する普及率でみると，洗濯機が 99.1%，冷蔵庫が 98.4%，エアコンディショナ 84.4%と大変高い水準である．[22] また種類でみても，テレビ，ビデオ，ゲーム機，電話機，携帯電話機，洗濯機，掃除機，冷蔵庫など多種多様である．これらの家電機器の多くが個々独立して機能することを前提として設計されている．しかし，インターネットで使用される様々なプロトコルを搭載した小型ICチップが開発され，ネットワーク接続機能を備えた家電機器が低コストで提供可能になった．例を挙げると，インターネットテレビ，インターネット冷蔵庫，インターネット電子レンジ，インターネットファックス，インターネット電話などである．将来もこのような，ネットワーク接続機能を備えた家電機器が登場すると予想される．これら情報を送信，または受信する機能を備える機器を**情報家電機器（Networked Home Appliances）**と定義する．

1.1.2 通信ネットワークの変遷

近年，インターネットは，図1.1 が示すようにように今までのメディアに比べて急速な成長を遂げている．単純計算で述べると電話の 15 倍の速度，携帯電話の 3 倍の速度で普及している．その理由の一つが，家庭からインターネットにアクセスするための通信回線として既存の基盤を利用した規格や新たな様々な規格が登場し始めた．その際家庭からインターネットへアクセスするネットワークは，電話回線を使ったモデ

ム通信が主流だった．しかし現在では，ISDN(64kbps)[14]，ADSL[13](下り側 512kbps) などの新しい規格が登場した．さらに，既存の CATV(30Mbps) や衛星通信 (2Mbps) を利用したインターネットアクセス形態の多様化に伴い通信速度が向上し，ユーザが利用する機会が増化している．

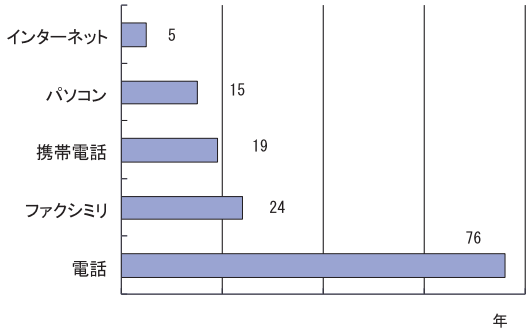

図 1.1: 主要通信メディアの世帯普及率 10%までの所要時間

また，それに伴い，家庭内で用いられる通信回線も様々な規格が登場している．10BASE-T 及び 100BASE-T，IEEE8021.11，IEEE1394，BlueTooth などである．表 1.1 に，ネットワーク形態の種類の幅が広がった事を示す．表 1.1では，インターネットの家庭までの接続を提供するアクセス系と表示し，家庭内での利用するネットワークをインハウス系と表示している．

以上で述べたことから，将来は様々なネットワーク対応の家電機器を，様々な通信ネットワークで接続することで必要な機能をやり取りし，相互に協調動作でき環境が構築可能となる．

1.2 問題意識

以上の背景から，通信機能を内蔵した情報家電機器が増加し，家庭内に普及し始めたネットワークに情報家電機器が接続される条件は整いつつある．既存の家電機器においては，家電機器の状態の変化をまず人間が読み取り判断して別の機器に操作を加えていた．例えば，図 1.2の様に，目覚まし時計のアラームからの通知を人間が聞取り，ライトの操作を加える行為は，人間がアラームの検知を行い，その通知を基に人間がライトの電源スイッチを入れている．しかし，人間が，多くの家電機器制御の仕組みを理解し，機器の状態の変化を見張りながら，他の情報家電機器を操作する事は負担になる．

上記の背景を踏まえ，家庭内において，メーカや用途の異なる情報家電機器間を相互接続し，情報家電機器間での状態の変化の通知を実現する．それにより，人間の中継を経由せずに機器間で状態の変化を通知でき，情報家電機器を人間がより有効に利用でき，生活環境の助けになる．そういった新しいネットワークを利用した情報家電

表 1.1: ネットワークの形態の広がり

形態	結線状態	名称	帯域 (bps)
アクセス系	有線	一般電話回線	54K
アクセス系	有線	ISDN	64K
アクセス系	有線	ADSL	512K
アクセス系	有線	CATV	30M
アクセス系	有線	FTTH	10M
アクセス系	無線	衛星通信	2M
アクセス系	無線	携帯電話	9.6K
インハウス系	有線	LAN	100M
インハウス系	有線	IEEE1394	400M
インハウス系	有線	電話線通信	56K
インハウス系	有線	HomePNA[18]	1M
インハウス系	無線	IEEE802.11b	11M
インハウス系	無線	赤外線通信	4M
インハウス系	無線	HomeRF(SWAP)	1-2M
インハウス系	無線	HomeRF(PHS)	64K
インハウス系	無線	BlueTooth	721K

機器の利用では図 1.2 の様に，人間は機器の設定を最初に行うだけで，人間の情報家電機器に対する操作が大幅に減少できる．

しかしそういった情報家電機器間相互利用には，いくつか問題点がある．

様々な機器を統合する汎用的な規格の欠如

複数の情報家電機器を互いに自律的に制御する仕組みを構築する場合，HAVi[15] などの複数のメーカが協調している規格があるが，テレビとビデオなどのように相互の機器の役割があらかじめ定まっている組み合わせに対してしかサポートしていない．そのため様々な機器の混在するネットワークに対処できない．

開発コスト

情報家電機器は，ユーザの都合により通信ネットワークに対し接続，切断，移動が頻繁に行われる可能性がある．したがって情報家電機器の開発者は，このような問題を解決するための複雑な機能を設計，開発にしなければならない．そのため，情報家電機器の開発コストは普通の家電機器よりも高くなってしまう．

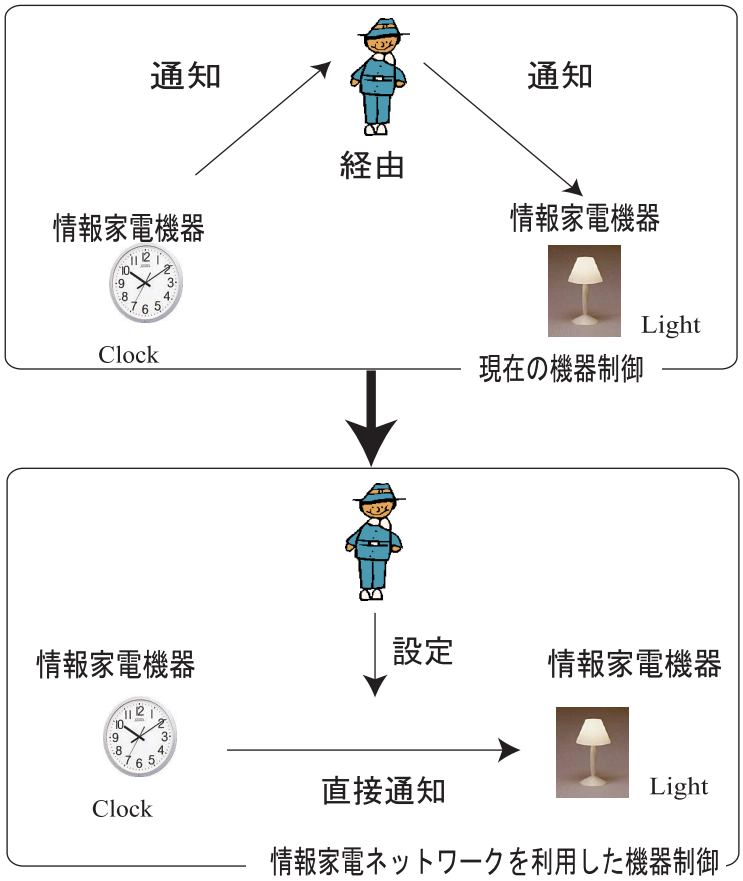

図 1.2: 情報家電機器間の連携

ユーザの利用コスト

家電機器は，高機能化するにつれ，設定や操作が複雑になりがちである．そのため，特に幼年者や高齢層のユーザは，複雑な家電機器操作方法を取得できない可能性がある．したがって，幼年者から高齢者まで様々な人にネットワークを利用した情報家電機器の相互利用を促すためには，操作インタフェースを簡単にする必要がある．

1.3 本研究の目的

情報家電機器間を相互利用するためは，異なるメーカや異なるOSの機器間での通信規格を定め，ある家電機器の状態の変化（以下 イベント）が，他の家電機器間へシームレスに通知される必要がある．上記の目標を実現するために，情報家電機器の状態の変化の通知を動的に組み合わせる動的分散イベント通知モデルを提案する．またユーザが情報家電機器の組み合わせを容易に設定，変更可能なGUIプログラムを含めたツールキットを提供する．

1.3.1　各機器を結ぶイベント通知モデルの提案

既存のイベント通知機構は，情報家電機器を中心をしたネットワークにおいて，情報家電機器間の状態の変化の通知を行うモデルを構築できない．既存の通信プログラムモデルがネットワーク障害を想定しておらず，動的に通知先を変更できないためである．その欠点を補う動的分散イベント通知モデルを提案する．これにより，情報家電機器を中心をしたネットワークに適したシステムが構築可能になる．

1.3.2　ツールキットの提案

本ツールキットにより，情報家電機器を動的に組み合わせ，統合的に利用できる．また，本ツールキットでは，視覚的な操作を用いて情報家電機器間のイベント通知先を組み合かえることを可能にし，情報家電機器間の相互利用を実現する．図1.3にGUIプログラムによりイベントの通知先を設定している様子を示す．左にイベントを送信する情報家電機器，右にイベントを利用する情報家電機器をアイコンによって表示し，両者のアイコンを結び合わせる事によって，二つの情報家電機器間のイベントのやり取り行えるように設定できる． 以下に本ツールキットの目標を述べる．

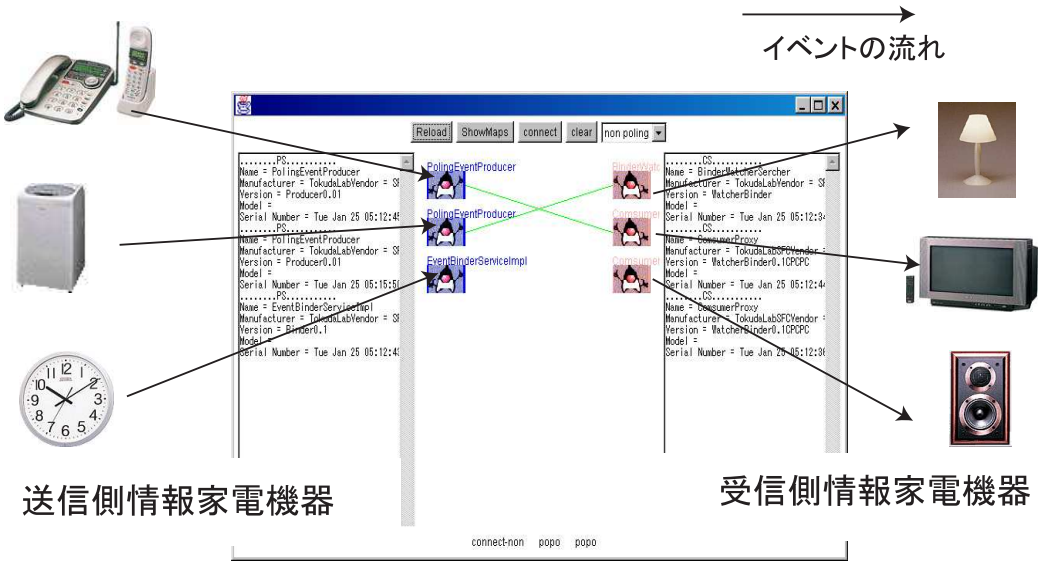

図 1.3: イベント通知先動的変更の例

情報家電機器を開発するコストの軽減

情報家電機器は，ネットワークから切り放さたり，ネットワーク上を移動したりする．しかし，システム全体はそれに影響されず機能しなければならない．そういった

煩わしいネットワークプログラムへの配慮を開発者が意識せず，情報家電機器を設計，開発できるツールキットを提供する．

情報家電機器間設定の複雑さの軽減

ユーザが，複雑な操作手順を踏まず，動的にイベント通知先を変更できるためのGUIプログラムをツールキットで提供する．このGUIプログラムにより各情報家電機器のイベント通知先に関する設定，変更を容易に行える．

1.4 本論文の構成

本論文の構成を以下に示す．

第2章で，情報家電ネットワークを定義し，情報家電ネットワークでの動向を述べるさらに情報家電ネットワークで考慮すべき課題を明らかにする．

第3章では，第2章で述べた課題を解決する方法は，既存の通信プログラムモデルでは不十分であることを示す．
そして，情報家電ネットワークに適したイベント通知モデル「動的分散イベント通知モデル」(Dynamically RedirectionAble Graphical Operating eNvironment/Event Model)を提案する．

第4章では，動的分散イベント通知モデル基にしたシステムを容易に構築できるツールキット: Keio Media Space FamilyDynamically RedirectionAble Graphical Operating eNvironment/Tool Kit(kmsf-DRAGON/TK)を提案し，このツールキットの目的と利用例を述べる．さらに，kmsf-DRAGON/TKの概要，全体構成，動作手順，設計方針，各システムの構成を述べる．

第5章では，kmsf-DRAGON/TKの実装環境，各システムの実装の詳細について述べ，応用例の実装方法につて述べる．

第6章では，kmsf-DRAGON/TKを基に構築したシステムの定量的評価を行う．また，kmsf-DRAGON/TKと，類似システムとの定性的評価を行う．

第7章で，全体のまとめとして，結論を述べ，今後の課題，目標について述べ，本論文を締めくくる．

第 2 章

情報家電ネットワーク

本章では，情報家電ネットワークの定義を行い，近年の情報家電ネットワークにおける研究の動向を述べる．情報家電ネットワークの特徴として，多様なネットワーク構成要素，幅広い利用者，幅広い利用用途が挙げられる．そういった特徴を持つ情報家電ネットワークが必要としている機能は，分散機能，検索可能機能，耐故障機能，動的変更機能がある．

2.1 情報家電ネットワークの概要

1.1節で述べた情報家電機器の発達と家庭内へのネットワークの導入により，情報家電機器を中心としたネットワークが構築可能になった．それらの情報家電ネットワークを本研究の想定環境とする．

2.1.1 用語の定義

情報家電ネットワークの定義を以下に示す．

> 有線および無線通信において以下のAとBの情報家電機器が2つ以上接続されているネットワーク
>
> **A** 通信デバイスにより情報を受け取り，操作を起こす，情報家電機器
>
> **B** 通信デバイスにより情報を発信し，他の情報家電機器を制御する情報家電機器
>
> すべで情報家電がインターネットにつながれている場面も想定している．

情報家電ネットワークは，家庭内やオフィスなどホームエリアを対象している．同時にホームエリア外部からのインターネット経由でのアクセスも想定している．

本論文で使用する用語の定義を行う．各用語の定義は，Sun Microsystems社の提案する分散環境 Jini[17]に基づいている．

イベント

本論文では，イベントを「機器の変化及び，ソフトウエアで記述された状態の遷移」と定義する．例として,「温度センサがある閾値を越えた」などの機器の状態の変化も含まれる．イベントを起こし，そのイベントを機器やソフトウエアに伝えることを**イベントを生産する**という．他のイベントを機器やソフトウエアから受信し，実世界へアクションを起こすことを**イベントを消費する**という．

サービス

本論文では，サービスを「センサやネットワークに対応した情報家電機器などのハードウエアやソフトウエア、あるいはそれらの組み合わせ」と定義する．サービスは，イベントに関してイベントを消費する側とイベントを生産する側の2種類に分類される．

- イベント生産サービス(Event Producer Service): イベントを発生させるサービス（センサ，警報器，アラーム，リモコンなどの外界感知機能を持ったサービス）
- イベント消費サービス(Event Consumer Service): イベントを受取り，実際にアクションを起こすサービス（テレビ，スピーカなど出力機器）

これらの機能を両方兼ね備えたサービスも存在する．

2.1.2 情報家電ネットワークの動向

近年，情報家電ネットワークに対応できる様々な製品が出荷され始めている．その一つとして，Java組み込み式 情報家電機器を取り上げる．また，分散環境下にある情報家電機器の制御に関する研究もいくつかある．情報家電機器をネットワークを用いて遠隔制御する様々な規格が登場している．主な規格として，HAVi[15]，Jini，UPnP[19]などがあげられる．

Java組み込み式情報家電機器

Java言語[10]は，Sun Microsystems社を中心に開発されたマルチプラットフォーム指向の言語である．近年，KVM[8]，ChaiVM [2], EmbeddedJava[9],
などの組み込み式Java Virtual Machine（以下 JavaVM）[11]の研究によりクライアントコンピュータだけでなく，制御機器分野において注目されはじめている．プラットフォーム透過性，分散システムとの親和性などを重視し，JavaVMを情報家電機器に埋め込み，Javaによって機器を制御する事ができる．Duonus[21]，PalmIII[3]，次世代携帯電話などが登場し始めている．それらは，Javaの通信機能を備えているため，ネットワークのプログラムの構築も容易になりつつある．今後もこういった，JavaVM組み込みの機器が増加すると見込まれる．

Home Audio-Visual Interoperability (HAVi)

HAViは、Sonyや松下電器など家電機器メーカが中心となって制定された規格で、AV機器がIEEE1394規格のインタフェースで統合されたホームネットワーク内で利用できる。HAViアーキテクチャでは，デバイスを制御機器(Controllers)と被制御機器(Controlled Devices)に分類している。制御機器はDCM(Device Control Model)を介して被制御機器を制御する。被制御機器は、当該機器を制御するために必要なコードを保持しており、これにはユーザインタフェースも含まれる。制御機器としてテレビを想定し、被制御機器としてCDプレーヤを想定する。CDプレーヤがホームネットワークに接続されるとまず、制御機器であるテレビに対して制御コードと当該CDプレーヤ用のアイコンデータが送信される。利用者は、テレビ画面に表示されたアイコンと、制御コードを用いてCDプレーヤを制御できる。

Universal Pulg and Play

Universal Plug and Play (以下UPnP)[12]は、HTTP[5]によって統合されるネットワーク環境である。相互に接続された情報家電機器を、利用者側計算機上(UPnPではこれをUser Control Pointと呼ぶ)のWWWブラウザまたはその他のアプリケーションソフトウエア(以下制御ソフトウエアと呼ぶ)から制御するための規格を定めている。情報家電機器は、当該機器の制御に必要となる情報をXML[20]を用いて定義し、制御ソフトウエアはRehydratorと呼ばれるソフトウエアコンポーネントを通して定義され

表 2.1: キャンパスネットワークと情報家電ネットワークの特徴比較

項　　目	キャンパスネットワーク	情報家電ネットワーク
ネットワーク構成要素	PC，WS，大型計算機	様々な情報家電
場所	大学内	家庭内
サービスの状態	ほぼ安定	不安定
サービスの識別	ホスト名+アドレス	様々
サービスの設定者	コンピュータに精通	一般ユーザ
サービスの利用者	一般ユーザ	一般ユーザ及び機器

た情報を得る。Rehydratorは、制御ソフトウエアから呼び出し可能なAPIを、取得したXML定義から生成する。

Jini

Jiniは，応用範囲が広い分散環境構築技術研究の一つである．Jiniではサービスは，ディスカバリプロトコルを用いて，Lookup Service（以下 LUS）の発見を行う．LUSは，分散環境下にある各サービスの情報を登録，管理する．このとき、LUSにはサービスの遠隔参照オブジェクト（Stub）がコピーされる。サービスを利用するプログラムは、LUSに対して利用したいサービスの情報を与え、合致するサービスの遠隔参照オブジェクトを得て、遠隔メソッド呼び出し（RMI[16]）によって当該サービスを利用する。これにより情報家電機ネットワーク環境において位置透過的に，情報家電機器の遠隔制御を実現できる。また，登録されたサービスは，Leaseによって管理時間を利用した管理がなされる．LUSは，ある一定のLease期間が過ぎると保持していたサービスを廃棄する．サービス側は，そのリース期間前にサービスをLUSに再登録を繰り返す.

Jiniには，以上の様にLUSによる位置透過性，Leaseの概念による耐障害性などのメリットがある.

2.2　情報家電ネットワークの特徴

情報家電ネットワークには，さまざまな利用者や，多様なサービスの形態などいくつかの特徴がある．キャンパスネットワークとの比較を行い情報家電ネットワークの特徴を明らかにする．情報家電ネットワークと一般的なキャンパスネットワークとの違いを表2.1に示す.

多様なネットワーク構成要素

キャンパスネットワークの構成要素が主にPCやWS 固定されているのに対して，情報家電ネットワークはPCやWSを含めた様々な情報家電機器を対象としている．テレビ，洗濯機，電話，ライトなどメーカも用途も異なる情報家電機器を管理しなければならない．

幅広い利用者

キャンパスネットワークの利用者は，コンピュータにある程度精通した人物が対象である．しかし，主に家庭内での使用を想定する情報家電ネットワークの利用者は高度な機器の操作に精通していないユーザが多い．また，ネットワーク設定やサービス間の設定はユーザにまかせなければならない．高齢者から幼年者まで，様々な年代とコンピュータに対するスキルレベルのユーザが対象となる．

幅広い用途

情報家電ネットワークは，様々な用途に活用できる．以下に具体的に，利用例を挙げる．

外出先からの情報家電の遠隔利用

- ユーザが仕事中に社内のネットワークを利用して家庭内のビデオ機器を操作する．
- ユーザが祖父母の家の状態を常に感知する様に設定，異常があったら場合に家のスピーカに通知する．

家庭内での情報家電機器の相互利用

- 朝の目覚ましアラームに連動して，エアコンとライトをつける．
- 洗濯が終わるとテレビに通知する．

2.3 必要な機能

以上の様な特徴をもつ情報家電ネットワークにおいて，情報家電間の相互利用を可能とし，情報家電機器同士で状態の変化の通知を可能とするために必要な機能がある．分散機能，検索可能機能，動的変更機能，耐故障機能について以下に詳細を示す．

分散機能

一台のPCがネットワーク上のすべての情報家電機器を制御するといった中央集中制御ではなく，ネットワーク上に分散した個々のサービス間で自律的に制御されなければならない．さらに家庭内の情報家電機器へ，インターネットを利用して，外部から操作を可能とする必要がある．例えば，先に述べた自宅の情報家電機器を会社のオフィスやから制御するといった事例が当てはまる．

検索可能機能

情報家電ネットワークにおいては様々なサービスが存在する．ユーザが利用する場合，そういったサービスをイベント生産サービスおよび，イベント消費サービス双方から検索できなければならない．また，サービスを様々な検索条件を用いて検索できる必要がある．例えば，テレビやパソコンなどのディスプレイを持ったサービスを，そのサービスだけでなくディスプレイの解像度や，大きさ，カラー情報などからも特定可能である事が望まれる．

耐故障機能

情報家電機器は，頻繁にネットワークから切断されたり，接続地点を変更されたりする．そのため，ネットワーク的な障害等に対して，システム全体が影響を受けない耐久性と安全性を保持する必要がある．
例えば，ユーザが普段持ち運んでいるPDAを用いて玄関のインターホンから発生するイベントを検知しているとする．PDAは，ネットワーク上の移動や，電池切れなどによるサービスの停止などの障害が起こる．それらサービスの移動などに，対処できる機能が必要である．

動的変更機能

様々な情報家電機器の接続を想定する情報家電ネットワークでは，イベント生産サービスとイベント消費サービスが存在する．それらを互いに結び合わせる事は，動的にユーザによって変更可能なことが求められている．ユーザが，動的に設定変更できなければ，情報家電出荷時の設定を利用しなければならず，ビデオとテレビ間といった様な，規定された枠組みでしか情報家電機器を利用できない．動的に変更可能であれば，情報家電機器を停止する事なく通知する機器の組み合わせを変更できる．
例えば，ユーザが今,「洗濯機の終了を部屋テレビに文字で通知」と設定し表示させたい場合などである．各情報家電機器メーカは，そういった事を予め想定していなかったとしてもユーザは動的に情報家電機器の組み合わせを設定できる．

2.4　本章のまとめ

本章では，情報家電ネットワークの定義を行った．近年の情報家電ネットワークにおける動向を述べた．さらに情報家電ネットワークの特徴として，多様なネットワーク構成要素，幅広い利用者，幅広い利用用途が挙げた．情報家電ネットワークの必要とされる４つの機能は，分散機能，検索可能機能，耐故障機能，動的変更機能である．この必要機能を満たし，情報家電機器の状態の変化を他の情報家電機器に送信する分散プログラムモデルを考察する．

第 3 章

動的分散イベント通知モデル (kmsf-DRAGON/EM)

情報家電機器の状態の変化を，他の情報家電機器（イベント消費サービス）へ伝える手法は，イベント通知モデルと後者をイベント問い合わせモデルの２つが存在する．それぞれの特徴について説明し，前章で述べた情報家電ネットワークにおける必要機能を満たすためには，イベント通知モデルが優れている事を述べる．また，イベント通知モデルに関する既存の研究が情報家電ネットワークのイベント通知モデルを満たしていないことを示す．本章の後半では，情報家電ネットワークにおける必要機能を満たすための動的分散イベント通知モデル (Dynamically RedirectionAble Graphical Operating eNvironment/Event Model) を提案する．

3.1　イベントモデルの分類

情報家電ネットワークでは，各サービスは分散的に存在し，イベント生産サービスの状態の変化をイベント消費サービスへ伝えることを前提にしている．それら二つのサービスの通信方法には，イベント生産サービスが能動的に状態の変化を通知する方法と，イベント消費サービスが能動的に問い合わせを行う方法の二通りが存在する．前者をイベント通知モデル，後者をイベント問い合わせモデルと呼び比較を行う．

3.2　イベント通知モデル

二つの情報家電機器間（サービス間）の通信とイベントの発生という要素を考えた場合に，イベント生産サービス側が能動的にイベント発生を検知し，イベント消費に通知を行うモデルが考えられる．**イベント通知とは，イベント生産サービスから発生するイベントをイベント消費サービスに伝えること**である．

3.2.1　非登録型イベント通知モデル

非登録型イベント通知モデルとは，情報発信者が複数の情報受信者のいるネットワークにブロードキャスト通信方式（以下 ブロードキャスト）やマルチキャスト通信方式[4]（以下 マルチキャスト）を利用してイベントを送信する仕組みである．図3.1に非登録型イベント通知モデルを示す．リモートコマンダなどは赤外線通信によってブロードキャスト方式でイベントを通知先の機器に伝える．リモートコマンダのボタンが押されたと言う状態の変化を赤外線が届く範囲一体に通知する．リモコン受信部を備えたエアコンなどはこれを利用できる．非登録型イベント通知モデルは，イベントを採用するか否かは，イベント消費サービスが決定する．また，一度の通知で，すべてのイベント消費サービスにイベント通知できるため，移動するサービスに対し，イベントを送る場合は有効である．マルチキャスト通信使用できるネットワークカテゴリを**セグメント**と呼ぶ．非登録型イベント通知モデルの欠点はそのセグメント越えたネットワークからの操作ができない点である．

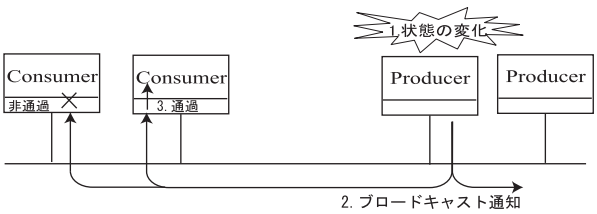

図 3.1: 非登録型イベント通知モデル

3.2.2　登録型イベント通知モデル

登録型イベント通知モデルとはイベント消費サービスがイベント生産サービスにイベント登録を行いイベント通知を受ける仕組みである．**イベント登録とは，イベント生産サービスに対して，イベント通知を受ける事を申請することである．**図3.2に，イベントの登録に関する一例を示す．図左の目覚まし時計がアラーム設定時間になると図右の電気スタンドを点灯させる場合を想定してみる．目覚まし時計は，イベントを発生させるサービスであり，設定時間になったらサービスの状態変化に対してイベントを発生させる．電気スタンドは，イベントを消費するサービスであり，電灯のON，OFFといったアクションを起こす．まず電気スタンド側が目覚まし時計に通知を受けたいという**イベント登録**を行ない，目覚まし時計はアラーム時間になるとイベントを発生させ電気スタンドに通知を行なえる．

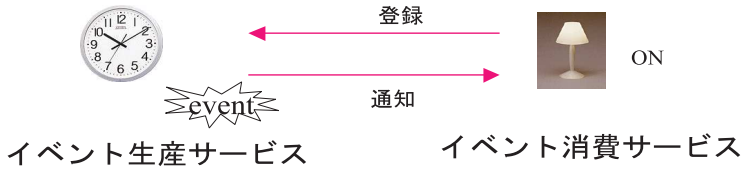

図 3.2: イベント通知とイベント登録

またイベント消費サービスは，一意なアドレス指定によりイベント生産サービスに登録を行うため，登録型イベント通知モデルの利点はセグメントを越えた通信が可能である．また通知頻度に関しては，イベント消費サービスがいない場合は発生せずであるため，イベント消費サービスが複数存在しない限りは低い．しかし，イベント消費サービスの数が増えれば，正比例して通知回数が増加する．また登録型イベント通知モデルは，登録したイベント消費サービスのアドレスを保持し通知を行うためサービス移動には対応できない．

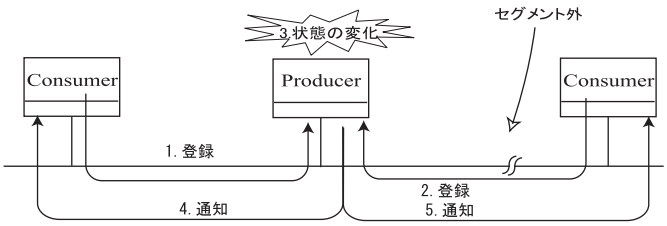

図 3.3: 登録型イベント通知モデル

3.3　イベント問い合わせモデル

イベント通知モデルとは逆に，イベント消費サービス側が必要なに応じてイベント生産サービスにイベントが発生していないか問い合わせるモデルである．

3.3.1　非登録型イベント問い合わせモデル

非登録型イベント問い合わせモデルとは，ポーリングにより一定周期にイベント生産サービスにイベントが発生していないか問い合わせるモデルである．その際，イベント消費サービスは，ブロードキャストやマルチキャストを用いて問い合わせを行い，返答を得る．非登録型イベント問い合わせモデルは，イベント生産サービスが単純なデバイスやセンサの時に有効である．デバイスやセンサは状態が変化している時にだけ，返答を返すだけで良いからである．ブロードキャストやマルチキャストを用いるため，イベント消費サービス，イベント生産サービスともに移動する場合に有効である．また欠点は，セグメントを越えた操作ができない事である．さらに，定期的な問い合わせのためイベント発生からイベント消費サービスがそれを受信するまでの間隔が大きくなる．それを改善するために，問い合わせ間の時間を短くすると無駄な通信が大量に発生することになる．

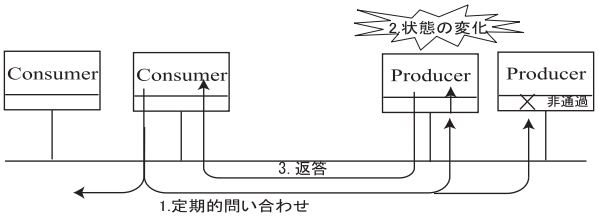

図 3.4: 非登録型イベント問い合わせモデル

3.3.2　登録型イベント問い合わせモデル

登録型イベント問い合わせモデルとは，イベント生産サービスがイベント消費サービスに登録を行い，イベント消費サービスの定期的な問い合わせによってイベント生産サービスがイベントが発生していたかどうかを返答するモデルである．登録型イベント問い合わせモデルの利点は，セグメントを越えることができる．また本モデルはイベント消費サービス側がイベント消費サービスへの参照を保持できるため，イベント消費サービスにとってが安全なイベント生産サービスを規定するセキュリティ機構を構築しやすい．また欠点は，イベント発生からイベント消費サービスがそれを受信するまでの時間が不規則ことである．

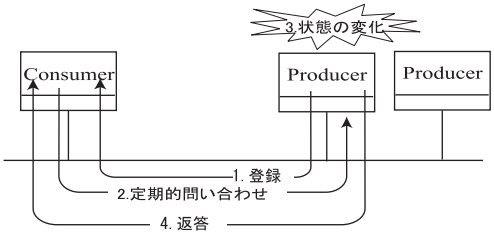

図 3.5: 登録型イベント問い合わせモデル

両イベントモデルの比較

各イベントモデルの比較を表3.1で示す．実装の容易さとは，サービスの管理を伴うプログラムであるかどうかを基準にしている．
セグメントを越えた通信とは，マルチキャストセグメントを越えてインターネット等を経由して通信を行う事ができるかどうかを基準としている．
サービスの移動とは，イベント生産サービスおよび消費サービスが移動する場合に移動したサービスに通知できるかどうかである．
通信を行うことができるかを基準としている．
タイムラグとは，イベントが発生してからイベント消費サービスに伝わるまでの時間のずれが大きいかどうかで判断している．
セキュリティ機構とは，有害なサービスへの盗聴を防ぐ仕組みが構築しやすいかどうかを判断している．
通信の無駄とは，イベントが発生していないイベント生産サービスある場合が存在するかどうかで判断している．
またはイベントを必要とするイベント消費サービスが存在しない場合に，通信が発生するかどうかが基準となっている．
イベント消費サービスの管理とは，イベント生産サービスがイベント消費サービスすべてを識別する仕組みを構築できるかどうかで判断してる．
イベント生産サービスの管理とは，イベント消費サービスがイベント生産サービスすべてを識別する仕組みを構築できるかどうかで判断している．
イベント消費サービスが非力な場合とは，イベント消費サービスがセンサなどの単純な機構しか持たない場合の事を述べている．**イベント生産サービスが非力な場合**とは，イベント生産サービスがセンサなどの単純な機構しか持たない場合の事を述べている．

イベント通知モデルは，イベント問い合わせモデルを比較してた場合，イベント通知モデルは，タイムラグのないモデルあり，通信の無駄が少ない．特に登録型イベント通知モデルをもちいれば，複数のイベント消費サービスを管理しやすい特徴をもち，マルティキャストを越えた通信を行うことも可能である．以上の理由から情報家電ネットワークの必要機能を解決する方法として，イベント通知モデルを情報家電ネットワー

表 3.1: アプローチ方法の比較

	イベント通知モデル		イベント問い合わせモデル	
名 称	非登録型	登録型	非登録型	登録型
実装の容易さ	容易	比較的容易	比較容易	複雑
セグメントを越えた通信	不可能	可能	可能	不可能
サービスの移動	対応	非対応	対応	非対応
タイムタグ	なし	なし	あり	あり
セキュリティ機構の構築	×	△	×	○
通信の無駄	あり	なし	あり	あり
イベント消費サービスの管理	×	×	×	○
イベント生産サービスの管理	×	○	×	×
イベント消費サービスが非力な場合	○	×	×	×
イベント生産サービスが非力な場合	○	×	○	×

クの課題解決に適しているイベントモデルであると判断できる.

3.4 イベント通知モデルに関する既存の研究

イベント通知モデルアプローチは，情報家電ネットワークに必要な分散機能を考えた場合，イベント通知モデルが適している，タイムラグのないイベント通知モデルを採用する．本節では，各種イベント通知機構および，デバイス間の協調に関する研究について述べ，既存のイベントモデルが，情報家電ネットワークのイベント通知モデルとしては，不十分であることを示す.

3.4.1 ECJ イベントモデル

ECJ[25]は，分散環境下での非同期イベント駆動型の分散システム構築用フレームワークである．発生頻度が高く，サイズの小さいイベントを高速に伝達する場合に適している．イベント駆動型システムはネットワーク経由の高速なイベント配送が可能になる．ECJの状態を図3.6に示す.

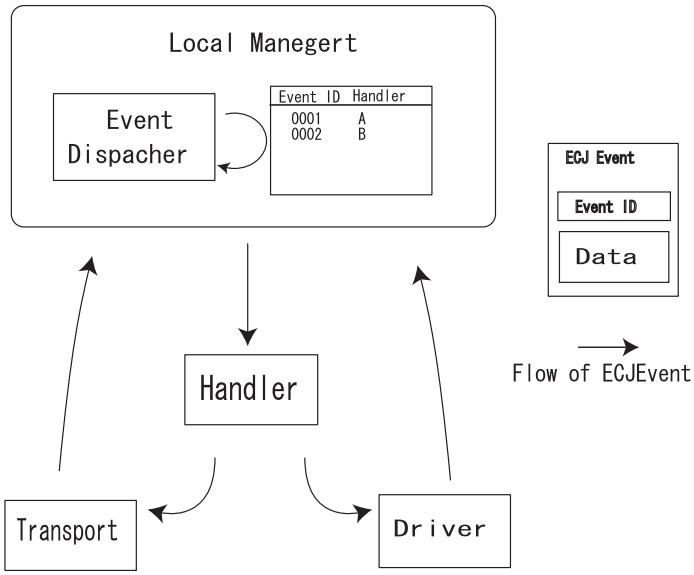

図 3.6: ECJ の概念

ECJEvent ECJ における抽象化されたイベント．イベント ID とデータからなる．

Local Manager Event Dispatcher と Event Table を持つ．Event Table はイベントの ID とイベントのハンドラの対応表である．この対応表をもとに，各 Driver や Transport からの ECJEvent を対応するハンドラへ渡たす．

Handler イベントのハンドラで，ECJEvent に対応する処理をプログラマが記述する．Local Manager によって，イベントの ID に応じて起動される．

Driver 既存の制御システム等で独自に規定されたイベント群を ECJEvent に変換，あるいは、その逆変換を行う．

Transport ネットワークなどへのアクセスを提供するドライバ群．通信層のプロトコル変更等にも柔軟に対応することできる．

ECJ は，イベントの ID とイベントハンドラのマップをさせたテーブルを Local Maneger が保持しているため，イベント通知先の動的な変更に対応できるが，サービスの検索やサービスの障害時の対応ができていない．このため情報家電ネットワークに適したイベントモデルとは言えない．

3.4.2 JavaBeans ローカルイベントモデル

JavaBeans は同一 JavaVM 上でのイベントモデルを細かに定義している．Beans はある一定の名前づけルールにしたがった，リスナ登録メソッドを持つことが，イベント生産オブジェクトの条件としている．また，そのリスナ登録メソッドを呼び出す統一的な手法で，イベント自身の登録が行われている．動的に，コンポーネント単位でプログラムを構築し，コンポーネント間のイベント通知によるプログラムを動的に設定

表 3.2: ローカルイベントモデルと分散イベントモデルの違い

項　　目	ローカルイベントモデル	分散イベントモデル
プラットフォーム	同一	分散
プロセス	同一プロセス	異なったプロセス
発生順到達	保証されない	保証される
伝達スピード	高速	低速
伝達信頼性	保証される	保証されない

し，その場で反映できる．しかし，JavaBeansは同一JavaVM上で動作するため，ネットワーク上に分散するプログラムとの違いがある．表3.2にローカルイベントモデルと分散イベントモデルとの比較を示す．

3.4.3　Jini分散イベントモデル

図3.7に，Jiniの分散イベントモデルの動作手順を示す．

1. イベント消費サービス（以下 Event Consumer Service）は，イベント生産サービス（以下 Event Producer Serviuce）の検索をルックアップサービス（以下 LUS）に行う
2. Event Producer Serviceの参照を持ったStubをコピーしてくる
3. その参照を基に Event Producer Serviceにイベント登録を行う
4. Event Producer Serviceでイベントが発生する
5. Event Producer Serviceは，登録に基きEvent Consumer Serviceにイベント通知を行う

Jiniを情報家電機器ネットワークに応用させるためには，いくつかの問題がある．登録型イベント通知モデルを参照しているため，動的なイベント通知先変更が不可能であると点とイベント消費サービスは，LUS登録されないため検索できない点である．

3.4.4　関連研究に対する共通した問題点

評価項目での記号「○」は評価項目を満たしていることを、記号「×」は満たしていないことを示す。また、記号「△」は評価項目に関する機能が部分的に提供されていることを示す。

表3.3からわかるようにに，各関連研究は，情報家電ネットワークに必要機能をすべて満たすイベント通知モデルなく，関連研究では情報家電ネットワークをの課題を

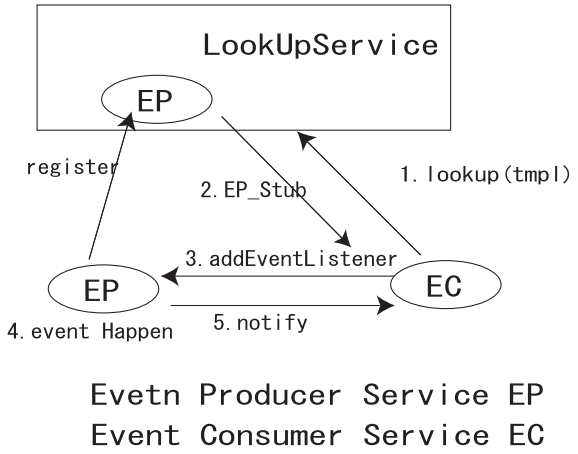

図 3.7: Jini Distribute Event Modelの概要

表 3.3: 各のイベント通知モデルのとの評価

研究名	分散機能	検索可能機能	耐故障機能	動的変更機能
ECJ	○	×	×	△
Beans	×	○	×	○
Jini	○	△	○	×

解決するシステムを構築できない.

3.5 動的分散イベントモデルの提案

既存のイベント通知モデルでは，情報家電ネットワークに必要である機能をすべて満たすことはできていない．そこで，動的分散イベント通知モデルを提案する．動的分散イベント通知モデルとは，情報家電ネットワークの必要機能である分散機能，検索可能機能，耐故障機能，動的変更機能を満たすイベント通知モデルである．このモデルをDynamically RedirectionAble Graphical Operating eNvironment/Event Modelと呼ぶ.

3.6 動的分散イベントモデルの手順

動的分散イベント通知モデル（以下kmsf-DRAGON/EM）では，イベント生産サービス（Producer）とイベント消費サービス(Consumer)のほかにルックアップサービス（以下LUS）および，イベントバインダサービス（以下Binder）の４つの構成要素から

なる．LUSは，イベント生産サービスの状態を保持，管理する．Binderは，イベント消費サービスの状態の保持，管理を行い．イベント消費サービスとイベント生産サービスを結びつける役割を持つ．kmsf-DRAGON/EMの手順を図3.8に示す．

1. LUSにイベント生産サービスが登録を行う．
2. LUSにBinderが登録を行う．
3. イベント消費サービスはLUSに対し，Binderの検索を行う．
4. Binderは，LUSに対してすべてのイベント生産サービスの検索を行う．
5. Binderは，イベント消費サービスのイベント登録を受ける．
6. Binderは，イベント消費サービスに通知先の変更命令を通知する．
7. イベント消費サービスは，命令に従いイベント生産サービスにイベント登録を行う．
8. イベント生産サービスはイベントを発生した段階で登録されているイベント消費サービスに対し通知を行う．

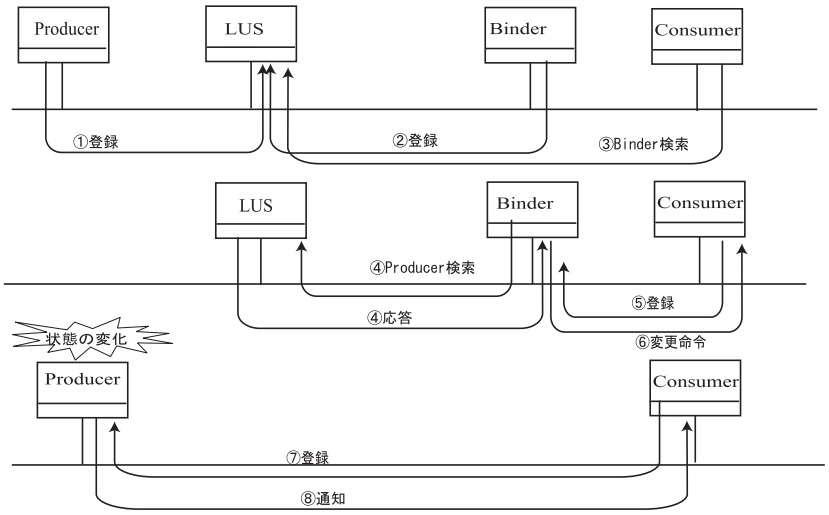

図 3.8: 動的分散イベント通知モデルの概念

3.7　機能

動的分散イベント通知モデルが提供する機能を述べる．各機能は，先にのべた情報家電ネットワークが必要とする機能に相当する．

分散機能

遠隔サービス利用機能

一意のアドレス指定により，家庭内の情報家電機器へ，インターネットを利用して，外部から利用できる仕組みを実現する．節3.2.2で述べた登録型イベント通知モデルを採用する事によってこの機能を満たす．これにより自宅の情報家電機器を会社のオフィスから制御するといったシステムを実装できる．

分散サービスイベント配送機能

一台のPCがネットワーク上のすべての情報家電機器を制御するといった中央集中制御ではなく，ネットワーク上に分散した個々のサービス間で自律的にイベントを送り互いに制御することを可能とする．

検索可能機能

イベント消費サービス検索機能

Jiniに代表される既存の分散システム多くはイベント生産サービスの検索だけを利用してきた．本モデルでは，サービスをイベント消費サービス，イベント生産サービス双方から検索できる機構を実現する．

柔軟なサービスの検索機能

情報家電ネットワーク上に存在するサービスを名前および属性情報などから検索を可能にする例えば，テレビやパソコンなどのディスプレイを持ったサービスを，そのサービスだけでなくディスプレイの解像度や，大きさ，カラー情報などからも検索できるようにする．

耐故障機能

サービスの動的発見機能

ネットワークの障害等でサービスが存在しなくなったり，移動してしまった場合に対処できる．サービスが分散的に管理および監視される事を実現する．イベント生産サービスは，一定間隔での登録の繰り返す要求する事で障害を検知する．また，サービスにネットワーク上の移動の関知を実現ために，イベント生産サービス，イベント

消費サービスとも管理サーバの発見は非登録型イベント通知モデルにより発見を行う．これによりPDAなどで，ネットワーク上を移動するサービスにも対処できる．

動的変更機能

イベント通知先動的変更機能

イベントを発生させる機器とイベントを受け取る機器は，動的にユーザによって変更可能にする．先に述べたイベント消費サービスの情報を管理するサービスに，ユーザ又は他のサービスからのイベント通知先変更命令を受け取る仕組みにする．これにより，登録先を常に変更する．kmsf-DRAGON/EMのイベント通知先動的変更機能を，図3.9にしめす．設定によって「電話の通知をテレビに表示」や,「エアコンの温度をテレビで表示」など情報家電機器が協調して，イベントをやり取りできる．

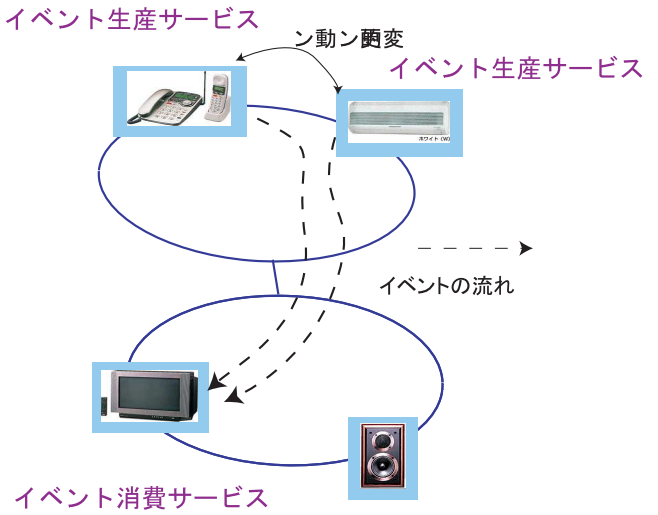

図 3.9: 通知先の動的変更

3.8　本章のまとめ

本章では，情報家電機器の状態の変化を，他の情報家電機器へ伝える手法である，イベント通知モデルとイベント問い合わせモデルの２つを比較し，イベント通知モデルが優れている事を述べた．また，イベント通知モデルに関する既存の研究が情報家電ネットワークのイベント通知モデルを満たしていないことを示した．本章の後半では，情報家電ネットワークにおける必要機能を満たすための動的分散イベント通知モデル(Dynamically RedirectionAble Graphical Operating eNvironment/Event Model)を提案をおこなった．

第 4 章

kmsf-DRAGON/TKの設計

kmsf-DRAGON/EMを普及させ，本論文の目標とする情報家電機器を開発するコストを軽減し，ユーザが設定する複雑さを軽減するために，本章ではkmsf-DRAGON/EMに基づいたシステムを構築を支援するツールキットを提案する．本章の後半ではツールキットの設計方針として，遠隔サービス利用機能，分散サービス間イベント配送機能，イベント消費サービス検索機能，柔軟なサービスの検索機能，サービスの動的発見機能，イベント通知先動的変更機能，簡易設定機能を述べる．さらに本システムの各モジュール部の設計について述べる．

4.1 kmsf-DRAGON/TKの概要

動的分散イベント通知モデル（kmsf-DRAGON/EM）をから広く利用可能にするためには，kmsf-DRAGON/EMに基ずくシステムが容易に構築されなければならない．本ツールキットは，kmsf-DRAGON/EMに基づいた分散システムを効果的に容易に構築する事を可能とするツールキットである．本ツールキットはDynamically RedirectionAble Graphical Operating eNvironment/Tool Kit（kmsf-DRAGON/TK）と命名する．kmsf-DRAGON/TKが上記の機能を果たしつつ目標とすべき問題を取り上げる．

開発者の負担の軽減

情報家電機器開発者に，本ツールキットを提供する事で，開発に関わるプログラムコストを軽減する，プログラムコストには，通信の障害やサービスの消滅とサービスの発生を検知し対処するプログラムコストと想定する事のできないサービスとの通信のためにかかるプログラムコストがある．

ユーザの負担の軽減

いくら，動的分散イベント通知モデルに適した情報家電機器を開発しても，その情報家電機器間のイベント通知設定がユーザの負担となるツールであってはならない．高齢者や，幼年者などの情報弱者を想定したシンプルなインタフェースでイベント通知先変更を設定でなければならない．

4.2 設計方針

本研究の目的の1つは，情報家電ネットワークに適した，動的分散イベント通知モデルの構築を効率的に行なえるツールキットを提供することである．

そのため本研究では，ネットワークプログラム部分を隠蔽し，情報家電機器間のイベント通知を動的に組み合わせるパッケージライブラリと，ユーザが情報家電機器の組み合わせを容易に設定，変更可能なGUIを構築するためのライブラリを構築する．

遠隔サービス利用機能

本ツールキットでは，IPセグメント的に全く異なる場所からのイベント通知やイベント切り替えを想定した場合に対応する．マルティキャストだけでなく，一意のアドレス指定により，家庭内の情報家電機器へ，インターネットを利用して，外部から利用できる仕組みを実現する．

表 4.1: サービスの特定方法

項　　目	サービスの検索方法
SOCKET	Hostname+ProtNum
RMI	Hostname+ObjectName
kmsf-DRAGON/TK	Object or Name or Atters

分散サービス間イベント配送機能

ネットワーク上に分散した個々のサービス間で自律的にイベントを送り互いに制御する仕組みを採用する．

イベント消費サービス検索機能

既存の Jini 分散イベントモデルでは，イベント生産サービスは，ルックアップサービスに登録させるため，他のサービスから検索されるが，イベント消費サービスはルックアップサービスに登録されないため，たのサービスが検索を行う事は不可能である．サービスを，イベント消費サービス，イベント生産サービス双方から検索できる仕組みを実現する．

柔軟なサービスの検索機能

表 4.1で示すように，既存のプログラム通信モデルでは，ソケット通信は，ホスト名とポート番号によって通信相手を特定し通信を行っていた．RMIにおいてもホスト名とサーバにあるオブジェクトの名前で通信相手を特定し通信を行っている．本ツールキットでは，情報家電ネットワーク上に存在するサービスを名前および属性情報などからの検索を実現する．例えば，テレビやパソコンなどのディスプレイを持ったサービスを，そのサービスだけでなくディスプレイの解像度や，大きさ，カラー情報などからも検索できるようにする．

サービスの動的発見機能

本ツールキットでは，サービスのネットワーク上の移動，消滅，発生を感知し，システム全体の円滑な動作を助ける．ネットワークの障害等でサービスが存在しなくなったり，移動してしまった場合に対処できる．サービスが分散的に管理および監視される事を実現する．イベント生産サービスは，一定間隔での登録の繰り返す要求する事で障害を検知する．また，サービスにネットワーク上の移動の関知を実現ために，イベント生産サービス，イベント消費サービスとも管理サーバの発見は非登録型イベント通知モデルにより発見を行う．これにより PDA などで，ネットワーク上を移動するサービスにも対処できる．

イベント通知先動的変更機能

本ツールキットでは，イベントの通知先をプログラムを停止する事なく変更でき，プログラム外部からのイベント通知先変更命令を常に受け入れることを可能とする．

簡易設定機能

本ツールキットでは，ユーザが容易に利用可能になっている．本システムで提供するGUIは，JavaBeans[7]のGUI構築画面のように視覚的にサービス単位で追加，削除，通知先変更，通知イベント変更を可能とする．JavaBeansのGUI構築画面とは，プログラムのコードを見ることなく，マウス操作のみで，ボタンやラベルの張り付けや切取などの操作を行なえるように設計されている画面のことである．このため，情報家電機器制御に慣れていないユーザにも容易に操作できる環境を提供できる．

4.3　全体構成

本節では，kmsf-DRAGON/TKの全体の構成をハードウェア，ソフトウェア両面から述べる．

4.3.1　ハードウエア構成

想定しているハードウエア環境を4.1示す．イベント消費サービスにはJavaVM組み込みテレビ，イベント生産サービスにはJavaVM組み込み電話機やJVM組み込み洗濯機を想定した．また，設定を行うGUIは，小型携帯端末機を用いているものと想定している．これらは，現在の市場ではあまり出回っていないが将来的には，十分想定できる環境にある．

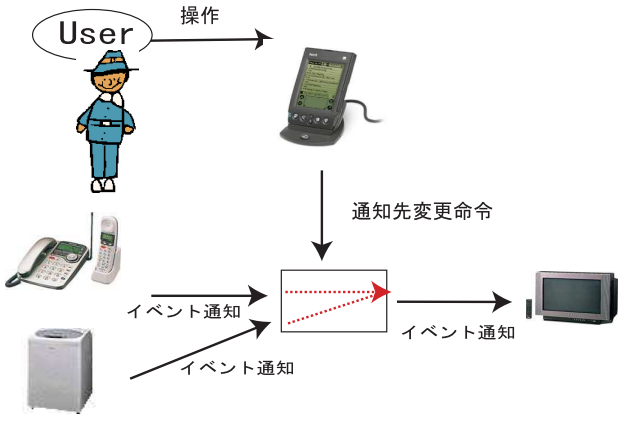

図 4.1: ハードウエアの構成

4.3.2　ソフトウエア構成

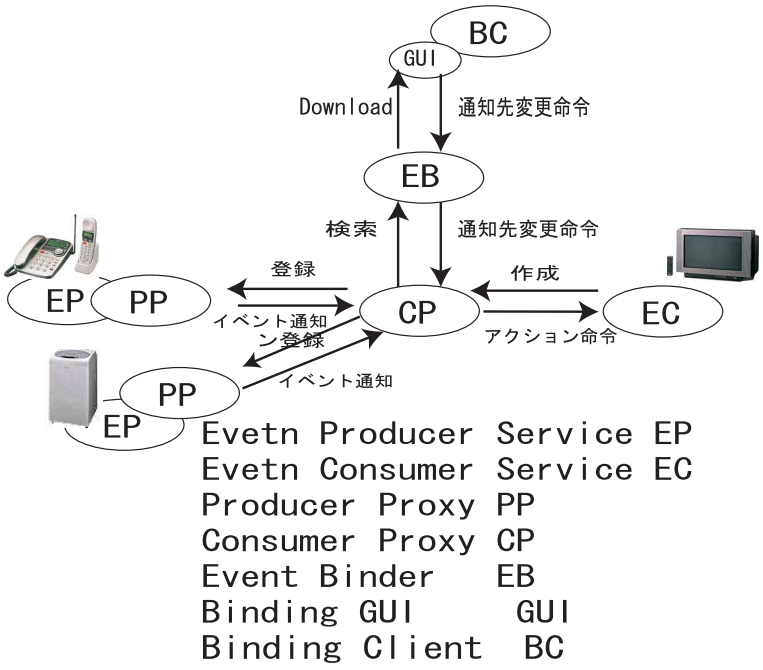

図 4.2: システム全体の概要

システム全体の流れを図 4.2に示す．左の洗濯機及び電話機の情報家電機器をイベント生産サービスとし，それぞれ洗濯終了イベントや電話呼び出しイベントを発生できると仮定する．右のテレビはイベント消費サービスとし，テレビに文字を表示したり，アラームを鳴らすなどのアクションを起こすことが可能である．
仮に，ユーザは，聴覚に障害を抱えており，電話やインターフォンの鳴っている状態をテレビで視覚的に確認したいと考えているとする．ユーザが行なうことは，手元の PC や小型端末機器（PDA）等を利用して，サービスが表示されている GUI 上のテレビのアイコンと電話機のアイコンを結び付けるだけである．その通知先変更命令は，サービス間の結び合わせを管理しているサービスに送信される．その通知先管理サービスは，通知先変更命令を，テレビサービス上で起動しているイベント消費サービスに通知する．テレビサービス上で起動しているイベント消費サービスは，通知先変更命令に従い電話サービスに対して自分自身の登録を行なう．この段階で電話が鳴るイベントが発生した場合，上記の登録にしたがってイベントがテレビサービスのイベント消費サービスに通知され，テレビサービスが「文字で知らせる」という行動を起こす．

4.4　各システム設計

本節では，各構成要素の説明と役割を行う．構成要素は，Remote Event Object, Event Producer Service，Producer Proxy，Event Consumer Service，Consumer Proxy,

Event Binder，Binding BUI，Binding Client，Commanderからなる．それぞれについて以下で説明する．

Remote Event Object

イベントオブジェクトは，イベントが発生と同時に生成され内部に，イベントID番号（Event ID），イベントの発生源クラス，イベントの発生順序を表すシーケンスナンバ（Event Sequence Number），および付加情報(Marshaled Object)を保持する．図4.3に概念図を示す．付加情報は，制御データ（Control Data）と，メディアデータ（Media Data）の両方を保持できる．Event Sequence Numberを用いる事で，ネットワークの遅延，イベントの欠損，イベントの通知遅れなどを検出できる．そのため，分散環境に適してた機能有する．

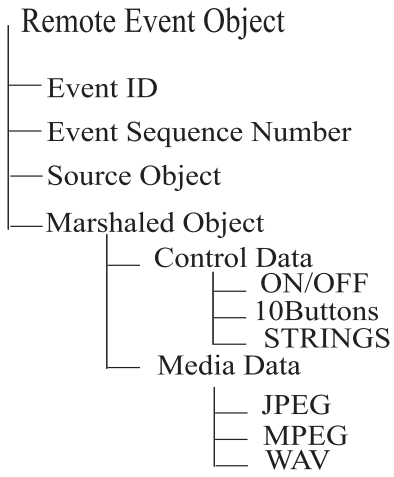

図 4.3: Remote Event Objectの保持する内容

Event Producer Service

2.1.1項で定義したイベント消費サービスである．イベント消費サービスにイベントの通知を行うサービス．洗濯が終了した事を通知するサービス人が通過した事を関知するサービスなどである．本サービスは，イベントをどういったタイミングで発生させるかを規定し，本ツールキットを用い情報家電開発者により実装されるモジュール部分である．

Producer Proxy

Event Producer ServiceをLUSに登録し，LUSに対し一定周期で登録を繰り返すサービス．また，Consumer Proxyからのイベント登録を受付その後通知を行う．そ

のためサービスの移動を検出をでき,「サービスの動的発見機能」の一部をはたす．またEvent Consumer Serviceが検索されるため，Event Binderに「イベント消費サービス検索機能」の一部を果たす．Event Binderについては，以下で説明する．さらに，Consumer Proxyからのイベント登録を受け付けるため「分散サービスイベント配送機能」の一部を果たしているといえる．

Event Consumer Service

章2.1.1で定義したイベント消費サービスである．イベントの通知を受け，アクションを起こすサービス．テレビに文字情報を表示するサービス，電灯の電源をONにするサービスなどである．本サービスは，アクションをどのように起こすかを規定し，本ツールキットを用い情報家電開発者により実装される．

Consumer Proxy

Event Producer Serviceから通知されたアクション命令をEvent Consumer Serviceに引き渡すプログラム．さらに，Event Binder Serviceの検出を行い，Event Producer Serviceに対してイベント通知登録を行なう．Consumer Proxyは，本システムのライブラリパッケージとして提供する．常にLookup Service上のイベント生産サービスの変化（消滅，新規登録，属性の変化など）を見張っている．そのためイベント生産サービスがいなくなってしまった場合や移動した場合に対応できる.「サービスの動的発見機能」の一部を果たしている．また，Event Binderからの通知先変更命令を受け付け，登録先を変更する機能を有する．そのため,「イベント通知先動的変更機能」の一部を果たしている．さらにProducer Serviceに対しイベント登録を行う機能を備えるため,「分散サービスイベント配送機能」の一部を果たしている．現在，一つイベント消費サービスに対して複数のイベント生産サービスがイベント通知を行うモデルは実現しない．複数のイベント生産サービスからの一つの消費サービスに対するイベント通知が同時にあった場合，イベント消費サービスがアクション中が，ONやOFFなどのアクションを起こす場合，イベント消費サービスに矛盾を起こしてしまうからである．

Event Binder

サービス間のイベントの組み合わせを行なうサービス．Jini Lookup Serviceから現在入手可能なサービスを検索し，次に説明するEvent Binding GUIに通知する.またConsumer Proxyにイベントの通知先変更する命令を通知する．Event Binderは，本システムのライブラリパッケージとして提供される．Event Binderは，Lookup Service上のイベント生産サービスの変化（消滅，新規登録，属性の変化など）を検知している．そのためイベント生産サービスがいなくなってしまった場合や移動した場合に対応でき,「サービスの動的発見機能」の一部を果たしている．動的変更命令をConsumer Proxyに送るそのため「イベント通知先動的変更機能」の一部を果たしている．また，イベント消費サービスからのイベント登録を受けると同時にイベント消費サービスの

属性情報を獲得しているイベント消費サービスの検索が可能になる．上記のように「イベント消費サービス検索機能」の一部を果たしている．

Binding GUI

ユーザがイベント通知先を組み合わせる段階で利用するGUI．ユーザにはアイコンを結びつけることにより，ネットワークや機器間の相互利用を容易に設定できる環境が提供される．Binding GUIは，ダウンロードして遠隔利用を提供しているためセグメントを越えて（マルティキャストの通過しないルータをこえて）利用できえる．そのため,「遠隔サービス利用機能」の一部を果たしている．また容易に通知先変更を設定できるインターフェースを備えるため「簡易設定機能」を果たしている．

Binding Client

Event Binderを発見し，ユーザがイベント通知先を組み合わせる段階で利用するBinding GUIをEvent Binderよりダウンロードしてくるプログラムである．アドレス指定により及びマルティキャストによりLUSを指定し，Event Binderを発見し，Event Binderが持つBinding GUIをダウンロードしてくるそのため，インターネット全体で利用でき,「遠隔サービス利用機能」の一部を果たしている．

Commander(通知先変更命令)

Commanderは，イベントの通知先変更にかんするイベント生産サービスとイベント消費サービスとの関係を表4.2の様に対応づける．サービスのID，Nmame，Vendor，Model Name及びLUSに登録されたサービスそのものをさすUpped Serviceを用いて表す．欄が空欄（Null）の場合は，ワイルドカードとして扱う．例えば，表4.3の様にCommanderを作成した場合，名前がWasher Serviceのすべてのサービスの通知をディスプレイ機能を持ったすべてサービスに通知すると規定した意味になる

現在，イベント消費サービスは，複数のイベント生産サービスの通知による機器の競合を避けるため，イベント消費サービス一つに対して，一つのイベント生産サービスしか登録できない.（一つのイベント生産サービスには，複数のイベント消費サービスが登録できる）イベント消費サービスを一意に指定する必要があり，サービスの一意な識別子であるService IDが必須になっている．

4.5 動作手順

本節では，kmsf-DRAGON/TKの主な構成要素であるProducer Proxy，Consumer Proxy，Event Binder の動作手順を述べる．

表 4.2: 通知先変更命令

Commander	
イベント生産サービス	**イベント消費サービス**
ID	ID
Name	Name
Vendor	Vendor
Model Name	Model Name
Upped Service	Upped Service

表 4.3: 通知先変更命令の一例

Commander	
イベント生産サービス	**イベント消費サービス**
null	null
Washer Service	null
null	null
null	null
null	display service

Producer Proxyの動作手順

本システムでのProducer Proxyの動作手順を図4.4に示す.

1. Producer Proxyは，Event Producer ServiceをLUSに登録する.
2. LUSに登録されたEvent Producer Serviceは，Event Binderに検索される.
3. LUSに登録されたEvent Producer Serviceは，Consumer Proxyに検索される.
4. Consumer Proxyは，検索したEvent Producer Serviceから，Producer Proxyにイベント登録をおこなう.

Consumer Proxyの動作手順

本システムでのConsumer Proxyの動作手順を図4.5に示す.

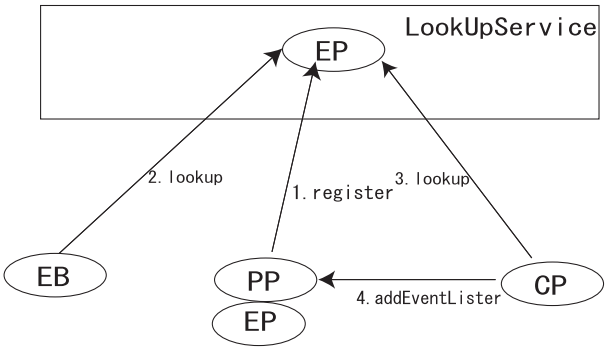

図 4.4: Producer Proxy の詳細な動作手順

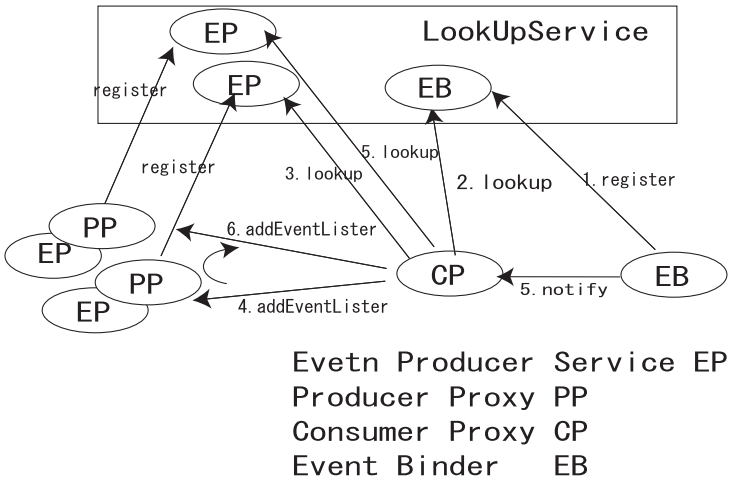

図 4.5: Consumer Proxy Service の動作手順

1. Event Binder は，LUS に登録される．
2. Consumer Proxy は，Event Producer Service を検索する．
3. Consumer Proxy は，さらに Event Producer Service を検索する．
4. Consumer Proxy は，発見した Event Producer Service の Producer Proxy にイベント登録する．
5. Event Binder は，変更通知命令を Consumer Proxy に送る．
6. Consumer Proxy は，それに従い，新たな Event Producer Service を検索する．
7. Consumer Proxy は，発見した Event Producer Service の Producer Proxy にイベント登録する．

Event Binderの動作手順

本システムにおけるEvent Binder動作手順を図4.6に示す.

1. Event Binderは，Lookup Service上にあるEvent Producer Service を検索する.
2. Event Binderは，LUSに自分自身を登録する.
3. Event Binderは，Consumer Proxyからの登録を受け付ける.
4. ユーザはBinding Clientを起動し，Event Binderを検索する.
5. Event Binderから，Binding GUIをダウンロードする.
6. Binding GUIは，ユーザにサービスの情報を表示し,イベント通知先書き換え命令を受け付ける.
7. Binding GUIは，通信によりEvent Binderに情報を送る.
8. Event Binderは，ユーザから得た命令をConsumer Proxyに通知する.

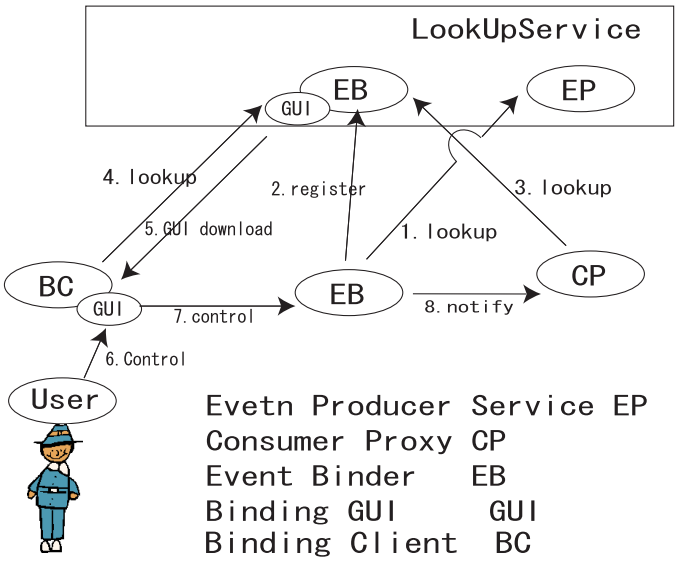

図 4.6: Event Binder の動作手順

4.6 本章のまとめ

kmsf-DRAGON/EMを普及させ，本論文の目標とする情報家電機器を開発するコストを軽減し，ユーザが設定する複雑さを軽減するために，kmsf-DRAGON/EMに基づいたシステムを構築を支援するツールキットを提案した．本章の後半ではツールキットの設計方針として，遠隔サービス利用機能，分散サービス間イベント配送機能，イベント消費サービス検索機能，柔軟なサービスの検索機能，サービスの動的発見機能，イベント通知先動的変更機能，簡易設定機能を述べた．さらに本システムの各モジュール部の設計について述べた．

第 5 章

実装

本章では，Dynamically RedirectionAble Graphical Operating eNvironment/Tool Kit の各モジュールごとに詳しい実装方針と提供機能を述べる．さらに，kmsf-DRAGON/TK を用いたイベント生産サービスおよびイベント消費サービスの実装例であるチャットシステムを示す．実装例により本ツールキットが，動的分散イベントモデルに適したシステムの構築が容易に行えることを示す．

5.1 実装環境

kmsf-DRAGON/TKの実装はJava言語を用いて行なった．実装言語としてJavaを選択し分散環境としてJiniを用いた．Jiniを利用した理由を以下に示す．

開発言語

Javaは，オブジェクト指向と，Garbage Collectionと言うメモリ管理によりプログラムの生産性をあげ，カプセル化機能により安全性を与えた．JavaVMが様々な情報家電機器に組み込まれている．以上の理由により，開発言語をJavaを採用し実装を行った．

分散環境基盤技術

Jiniとは，分散環境構築技術の中で，応用範囲が広い研究の一つである．Jiniには，Lookup Serviceによる位置透過性，Javaによるプラットフォーム透過性，Leaseの概念による耐障害性などのメリットがある．そのため，分散環境構築技術としてJiniを利用した．

5.2 kmsf-DRAGON/TK： モジュール部

図??で示す様に，kmsf-DRAGON/TKパッケージは，comsumer，producer，bainderという３つのディレクトリからなる．**jp.ac.keio.sfc.ht.dragon.comsumer**パッケージは主に，イベント消費サービスを開発する際に利用するクラス群である．**jp.ac.keio.sfc.ht.dragon.prod**パッケージは主に，イベント生産サービスを開発する際に利用するクラス群である．**jp.ac.keio.sfc.ht.dragon.binder**パッケージは主に，イベントの切り替えを行うのに必要なクラス群である．

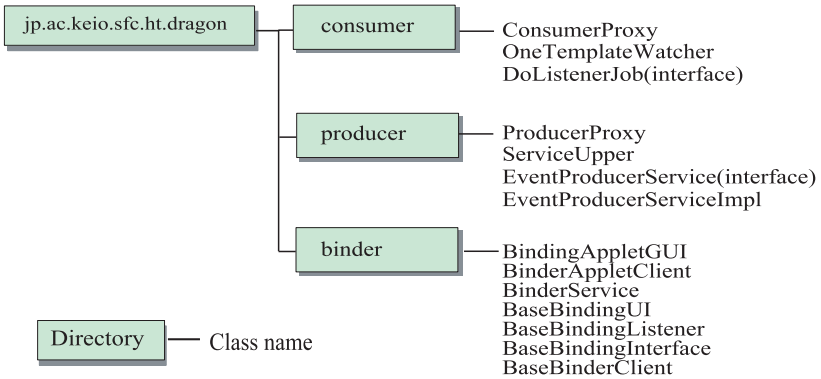

図 5.1: パッケージDragonTKの構成

5.2.1 OneTemplateWatcherクラス

OneTemplateWatcherクラスは，**jp.ac.keio.sfc.ht.dragon.comsumer**中のクラスである．イベント生産サービスに対して，イベント登録を行うクラスである．図5.2にクラスの状態遷移を示す．OneTemplateWatcherクラスの主な機能を以下に示す．

- マルチキャスト及び明示的なアドレスの指定によりLUSの検索を行う．
- LUSの上の要求されたイベント生産サービスの検索を行う．
- 発見したイベント生産サービスへのイベント登録を行う．
- LUSの上の要求されたイベント生産サービスが登場，消滅の検知する．
- 要求したイベント生産サービスが登場をした場合，そのサービスに対しイベント登録を行う．

OneTemplateWatcherは，LUSの検索をマルティキャス及び，下記に示すlookupLocaterの配列により検索できる．そのため，同一セグメントか，ある複数の固定的なアドレス配列によってLUSを検索できる．

```
lookupLocater[]={
                new lookupLocater(‘‘jini://dali.ht.sfc.keio.ac.jp’’),
                new lookupLocater(‘‘jini://wanwan.sfc.wide.ad.jp’’),
                new lookupLocater(‘‘jini://zz023.sfc.keio.ac.jp’’),
}
```

5.2.2 ServiceUpperクラス

ServiceUpperクラスは，**jp.ac.keio.sfc.ht.dragon.producer**中のクラスである．図5.3にクラスの状態遷移を示す．ServiceUpperクラスの主な機能を以下に示す．

- マルチキャスト及び明示的なアドレスの指定によりLUSの検索を行う．
- LUSの上へイベント生産サービスを登録を行う．
- LUSの上のイベント生産サービスを常に一定周期で再登録を繰り返す．

5.2.3 EventProducerImplクラス

EventProducerImplクラスは，**jp.ac.keio.sfc.ht.dragon.producer**中のクラスである．イベント通知先のイベント消費サービスをデータベースで管理し，実際にイベント通知を行う．ProducerProxyクラス及び，EventBinderクラスで使用される．EventProducerImplクラスの主な機能を以下に示す．

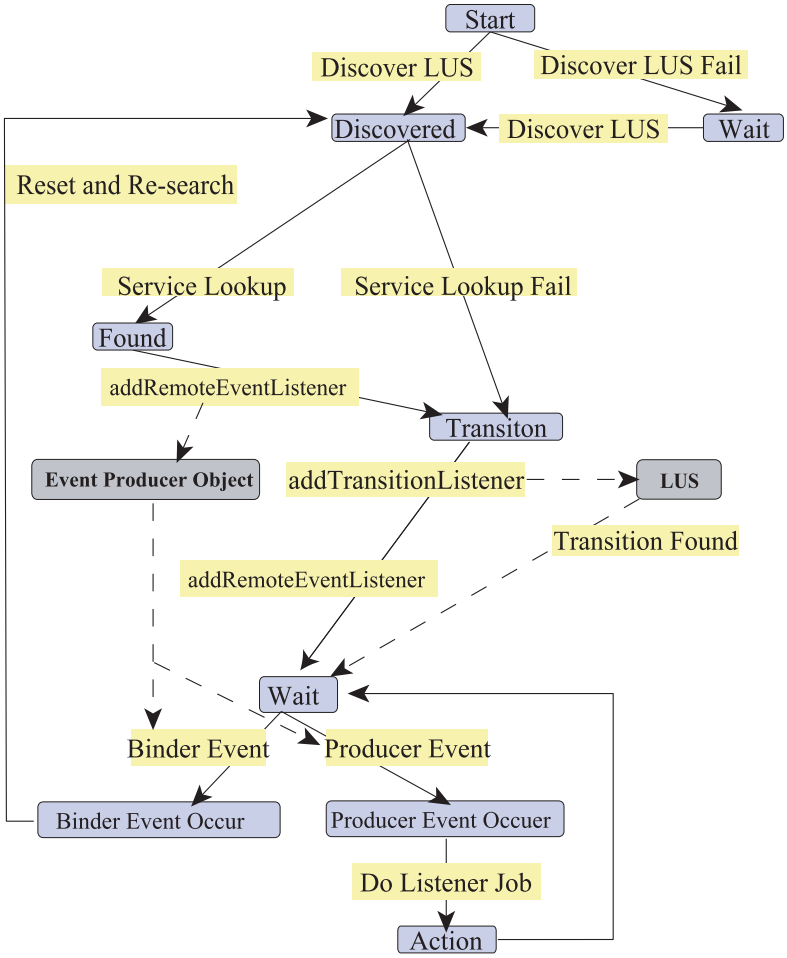

図 5.2: OneTemplateWatcher クラスの状態遷移

- イベント登録を行うサービスへの参照を保持する．
- イベント登録のキャンセルを行うサービスへの参照を破棄する．
- 登録されているサービスが持つnotify メソッドを呼び出し，引数としてRemote Event Objctを渡す．

5.2.4 BaseBinderUI クラス

BaseBinderUI クラスは**jp.ac.keio.sfc.ht.dragon.binder** 中のクラスである．ユーザに設定の促す際に用いるUIを構築する際の基本機能を持ち，GUI及びUIを作成する際に必ず継承される．BaseBinderUI クラスの主な機能を以下に示す．

- EventBinder クラスが保持しているイベント消費サービスの情報を獲得する．

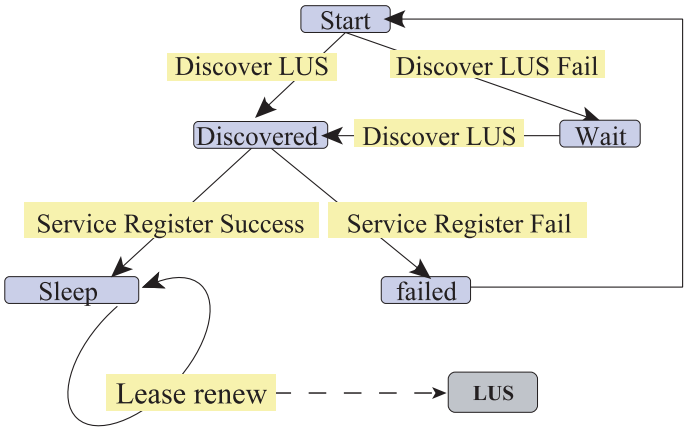

図 5.3: ServiceUpper クラスの状態遷移

- EventBinder クラス保持しているイベント生産サービスの情報を獲得する．
- EventBinder クラスが保持しているイベント消費サービスとイベント生産サービスの関連テーブルを獲得する．
- ユーザが指定した設定に基づいたて通知先変更命令の作成する．
- EventBinder クラスに通知先変更命令を送信する．

5.2.5 BaseBinderClient クラス

BaseBinderClient クラスは，**jp.ac.keio.sfc.ht.dragon.binder** 中のクラスである．上記の BaseBinderUI クラスを拡張させた UI クラス検索するクラスである．BaseBinderClient クラスの主な機能を以下に示す．

- マルチキャスト及び明示的なアドレスの指定により LUS の検索を行う．
- LUS より，EventBinder クラスを検索する．
- EventBinder クラスとともに LUS に登録されたクラスを検索する．

5.2.6 ProducerProxy クラス

ProducerProxy クラスは，**jp.ac.keio.sfc.ht.dragon.producer** 中のクラスであり，ServiceUpper クラス及び，EventProducerService クラスで実装する．本クラスを用い開発者はイベント生産サービスを実装する．実装例を図 5.9示す．また，図 5.4にクラス構成図を示す．

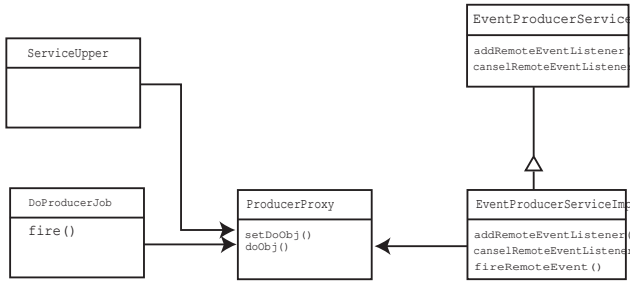

図 5.4: ProducerProxy のクラス構成図

5.2.7 ConsumerProxy クラス

ConsumerProxy クラスは，**jp.ac.keio.sfc.ht.dragon.consumer** 中のクラスであり，OneTemplateWatcher からなる．本クラスを用い開発者はイベント消費サービスを実装する．実装例を図**??**示す．また，図 5.5にクラス構成図を示す．

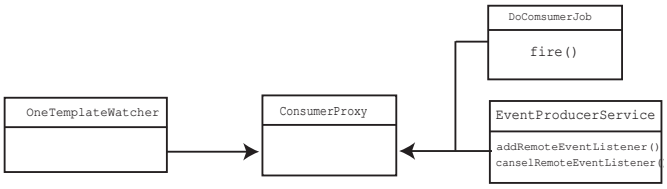

図 5.5: ConsumerProxy のクラス構成図

5.2.8 EventBinder クラス

EventBinder クラスは，**jp.ac.keio.sfc.ht.dragon.binder** 中のクラスであり，ServiceUpper クラス，EventBinderImpl クラス，OneTemplateWatcher クラスからなる．本システムの重要な構成要素であり，ユーザにより情報家電ネットワークで起動される．図 5.6にクラス構成図を示す．

5.2.9 BindingAppletGUI クラス

BindingAppletGUI クラスは，**jp.ac.keio.sfc.ht.dragon.binder** 中のクラスであり，BaseBinderUI クラスを用い実装される．本クラスは，ユーザによりダウンロードされ，イベント通知先の変更の操作される．図 5.7にクラス構成図を示す．

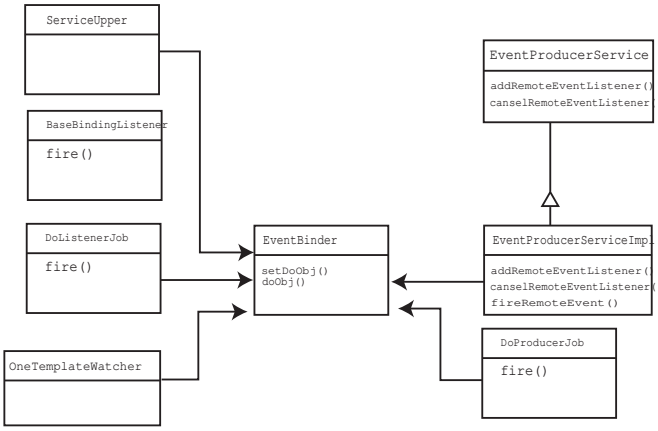

図 5.6: EventBinderのクラス構成図

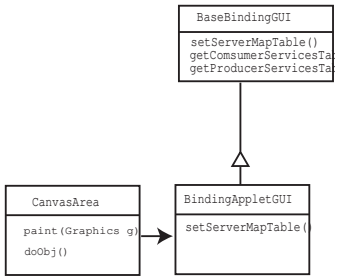

図 5.7: BindingAppletGUIのクラス構成図

5.2.10 BindingAppletClientクラス

BindingAppletClientクラスは，**jp.ac.keio.sfc.ht.dragon.binder**中にある．本クラスは，ユーザがBindingAppletGUIクラスをダウンロードする際に利用する．図5.8にクラス構成図を示す．

5.3 パッケージを用いたシステムの実装例

kmsf-DRAGON/TKを用いたシステムの構築例として，チャットシステムを挙げる．イベント生産サービスとして，KeyInputEventProducerクラスを実装する．KeyInputEventProducerクラスは，あるオペレータからの情報をキー入力をイベントとして発信するサービスである．イベント消費サービスとして，KeyInputEventConsumerクラスの実装を実装する．KeyInputEventConsumerクラスは，イベントを受け取るとそのイベントの付加情報を表示するクラスである．この実装例は，キーを入力する動作を，このまま情報家電機器から発生するイベントと置き換え，KeyInputEventConsumer

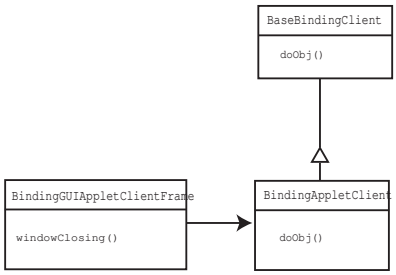

図 5.8: BindingAppletClinet のクラス構成図

クラスが表示取る部分を情報家電機器のアクションと置き換えると，この例をそのまま，情報家電ネットワークのシステムとして応用できる．本実装例は，イベント生産サービス，イベント消費サービスともマルチキャストによってLUSを探す実装であり，明示的なアドレスの指定は行わない．

KeyInputEventProducer クラスの実装

KeyInputEventProducer クラスのソースを図 5.9示す．KeyInputEventProducer クラスは，20 行目から 24 行目まで ProducerProxy のインスタンスを作成し，イベント作成する段階でインスタンスの 34 行目で fire メソッドを呼び出す．それによって，もしイベント登録をしているイベント消費サービスがあれば通知を行う．

KeyInputEventConsumer クラスの実装

KeyInputEventConsuemer クラスのソースを図 5.10示す．KeyInputEventConsumer クラスは，20 行目から 23 行目までで ConsumerProxy のインスタンスを作成し，その引数に 27 行目から 37 行目で作成した DoListenerJobImpl を引数にとる．DoListenerJobImpl は，32，33 行目で，通知されたイベントの付加情報の表示を行っている．

動作様子

本チャットシステムが起動している動作の様子を図 5.11に示す．左に並んでいるのが KeyInputEventProducer であり，右に表示されているのが KeyInputEventConsumer である．サービスはすべて異なるホストで実行されている．ユーザが任意のイベント生産サービスとイベント消費サービスを選び動的に結びつけを変更できる．インターフェースは，マウスによってドラッグアンドドロップを行うだけである．

```
 1 import jp.ac.keio.sfc.ht.dragon.producer.ProducerProxy;
 2
 3 public class KeyInputEventProducer implements Runnable{
 4    public static ProducerProxy pproxy = null;
 5    public KeyInputEventProducer() throws RemoteException {
 6           String[] groups = {"tailor"};//LUS Group
 7           Entry[] entries =
 8           {
 9                  new Name("KeyInputEventProducer"),
10                  new ServiceInfo("KeyInputEventProducer",
11                         //Name of this Service
12                         "TokudaMuraiKusumotoNakamuraLab",
13           //The name of the manufacturer or author of this service
14                         "SFC",
15                         //The name of the vendor of this service
16                         "0.1b",
17                         //The version of this service
18                         ""),
19           };
20                 pproxy = new ProducerProxy(
21                                  groups ,//set LUS Groups
22                                  entries ,//set Service Atters to LUS
23                                  60*5 ,//更新頻度（5分ごとにLUSに登録更新）
24                                  12*24*365*1);//継続回数（約1年間）
25    }
26
27    public void run(){ //Threads method
28         int idint; String stringl;
29           while(true)
30           {
31                 System.out.print("input: ");
32                 System.out.flush();
33                 stringl = in.readLine(); //read prompt
34                 pproxy.fire(idint++,new MarshalledObject(stringl));
35                 //Event Occered
36                 System.out.flush();
37           }
38 }
```

図 5.9: KeyInputEventProducerの実装

5.4 本章のまとめ

本章では，Dynamically RedirectionAble Graphical Operating eNvironment/Tool Kitの各モジュールごとに詳しい実装方針と提供機能を述べた．さらに，kmsf-DRAGON/TKを用いたイベント生産サービスおよびイベント消費サービスの実装例であるチャットシステムを示した．本ツールキットを用いることで分散環境において動的分散イベント通知モデルに基いたシステムを構築が容易になった．

```
 1 import jp.ac.keio.sfc.ht.dragon.comsumer.ConsumerProxy;
 2 import jp.ac.keio.sfc.ht.dragon.comsumer.DoListenerJob;
 3
 4 public class KeyInputEventConsumer{
 5
 6      KeyInputEventConsumer() {}
 7      public static void main(String[] args) throws Exception{
 8           String[] groups = {"tailor"};//LUS Group
 9           try {
10                ServiceInfo sinfo =
11                new ServiceInfo("KeyInputEventConsumer",
12                          //Name of this Service
13                                    "TokudaNuraiKusumotoNakamuraLab",
14           //The name of the manufacturer or author of this service
15                                    "SFC",
16                          //The name of the vendor of this service
17                                    "0.1b",
18                          //The version of this service
19                                    "","");
20                new ConsumerProxy(groups,//set LUS Groups
21                                    new DoListenerJobImpl(),
22                                    //set the Event action
23                                    sinfo);//set Consumer Info to binder
24           } catch (Exception ex) {}
25      }
26
27      public class DoListenerJobImpl implements DoListenerJob
28      {
29           public DoListenerJobImpl(){}
30           public void  fire(RemoteEvent event) throws RemoteException{
31                try{
32                  System.out.println(">>"
33                   +event.getRegistrationObject().get()+":");}
34                catch(IOException e){}
35                catch(ClassNotFoundException e){}
36            }
37      }
38 }
```

図 5.10: KeyInputEventConsumerの実装

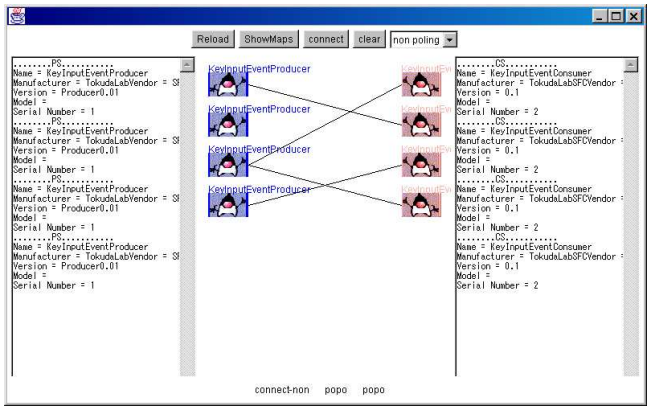

図 5.11: BindingAppletGUIの画面

第 6 章

評価

本章の前半では，kmsf-DRAGON/TKを用いて構築されたアプリケーションのイベント伝搬能力及びイベント通知先切り替え能力について定量的評価を行ない，本ツールキットの性能が情報家電ネットワークにおいて利用できる範疇にあることを示す．また本章の後半では，機能比較し，kmsf-DRAGON/TKが関連するイベントモデルに比べて機能的に優れていることを示す.

6.1 定量的評価

イベント伝搬にかかる時間の測定及び，イベント通知先切り替え能力を測定する．

6.1.1 イベント伝搬にかかる時間の測定

一台のマシン上で起動しているイベント生産サービスが複数のイベント消費サービスにイベント通知を行うまでのイベント伝搬に関わる時間を測定する．本システムが機能的に速度の面で問題ないことを示す．

測定方法

測定用アプリケーションには，一秒ごとに1024byteイベント付加情報を保持たイベントの伝搬を行うイベント生産サービスを作成し，イベント消費サービスを1台，3台，7台のマシンで実行しイベント通知をおこなった．すべてのイベント消費サービスは，イベント生産サービスにイベント登録を行う．イベント生産サービスは，すべてのイベント消費サービスに送信する．その送信始めから送信終了までの時間を，100回測定した．

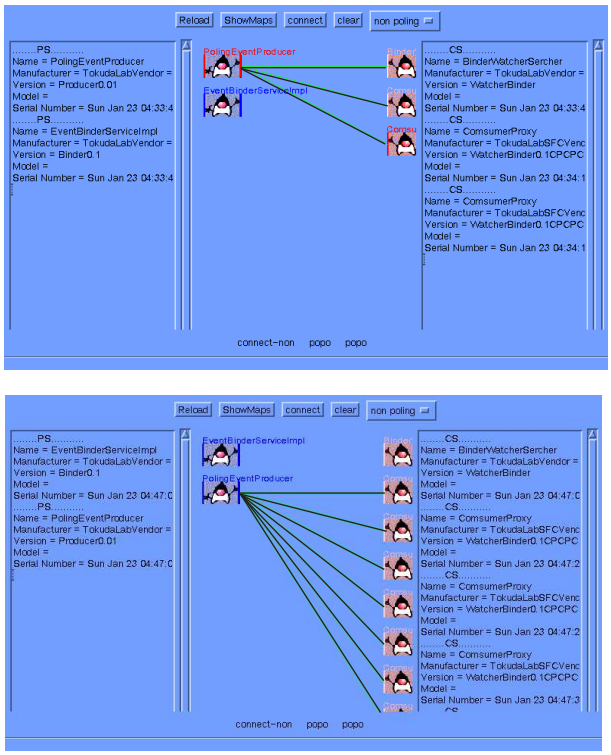

図 6.1: 複数のイベント消費サービスにイベント通知中の画面

測定環境

測定には100Base-Tで相互接続された，Sun Microsystems社のワークステーションであるUltra 30をイベント生産サービスの実行に1台使用した．表6.1にUltra 30の性能を示す．またイベント消費サービスにそれぞれ，1台，3台，7台のUltra 30を使用した．

表 6.1: 測定環境

CPU	UltraSPARC-II 248MHz × 1
メモリ	128MB
OS	Solaris2.6
JDK version	Solaris_JDK_1.2.1_03_pre-release
Jini version	1.0.1

測定結果

測定結果を表6.2に示す．また，横軸にイベント消費サービスの数を，縦軸に通知にかかる総時間を示したグラフ6.2を表示する．

結果考察

また一つのイベント生産サービスにイベント登録を行うイベント消費サービスが爆発して増える事はないと想定している．情報家電ネットワークは，ホームエリアの使用を目的をしているそのため，想定している情報家電ネットワークでは，目覚まし時計のアラームをならすなどの1秒単位で，イベントの発生を行う情報家電機器が多く存在する．1秒を情報家電ネットワークの基準単位とし、上記のグラフの傾きを5．7とすると毎秒173個の消費者サービスに同時に通知できる計算である．この値は，家庭内での情報家電ネットワーク中で一つのイベント生産サービスが通知を行う数としては大きいと言える．

表 6.2: 通知時間測定結果

イベント消費サービス	**測定値 (ms)**
1	12.0
3	23.9
7	46.3

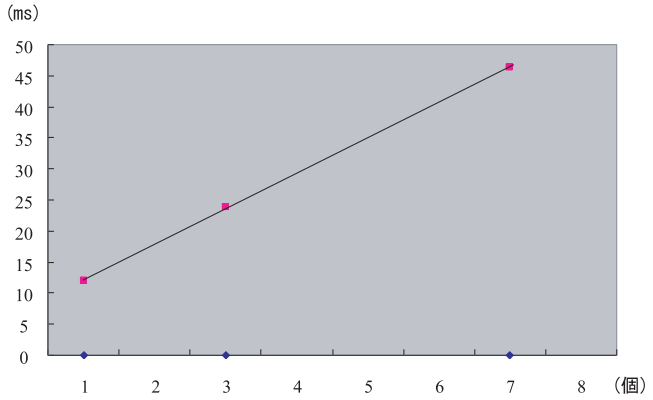

図 6.2: 通知時間測定結果

6.1.2 イベント通知先切り替えにかかる時間の測定

ユーザが GUIにより操作を繰り返してから切り替えがイベント生産サービスに反映されるまでの時間を測定した．イベント通知先切り替えにかかる時間は，比較的に短い事を示す．また各モジュールのオーバヘッドを測定した．通知先変更命令は，図6.3の様に逐次伝搬される．各モジュールは，以下に示す区間である．

- 区間AB: BindingGUI クラスに対する入力からの描画処理
- 区間BC: アプレット通信にかかった時間
- 区間CD: EventBinder クラスで消費された時間
- 区間DE: ConsumerProxy クラスに通知先変更命令が通知される時間
- 区間EF: ConsumerProxy クラスで消費された時間
- 区間FG: ConsumerProxy クラスで発生したイベント登録が ProducerProxyに完全に登録されるまでの時間

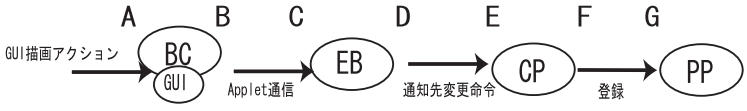

図 6.3: 各区間の説明

表 6.3: 切り替え時間測定結果

AB	BC	CD	DE	EF	FG	TOTAL（ms）
1.6	31.6	0.1	35.6	99.3	48.4	216.6

測定方法

測定用アプリケーションには，一秒ごとにイベント伝搬を行うイベント生産サービスを作成し，イベント消費サービスを 1 台イベント通知命令を記述してからその命令が反映されるまでの時間を測定する．イベント登録と登録のキャンセルを 5 秒ごとに繰り返し，GUIに接続を示すラインを描画してから，通知先命令が反映されるまでの時間を，100 回測定した．

測定環境

測定には 100Base-T で相互接続された，Sun Microsystems 社のワークステーションである Ultra 30 を 1 台使用した．同マシン上で，バインダアプレット，バインダサービス，イベント消費サービス，イベント生産サービスを起動した．

測定結果

測定結果を表 6.3に示す．また，各モジュールに生じたオーバヘッドをグラフ 6.4に表示する．

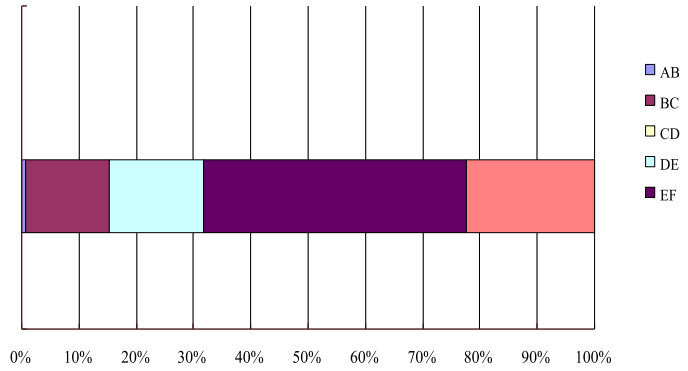

図 6.4: 切り替え時間測定結果

結果考察

ユーザがイベント切り替え命令を描画させてから切り替え反映されるまでの時間は，約 216 ミリ秒かかっている．このうち本システムでのオーバヘッドは AB+CD+EF＝101ms

を必要とする．そのために，基準時間内にイベント通知先変更機能が果たされる．

6.2 定性的評価

他の研究機関等で開発されている関連システムのイベントモデルおよび本ツールキットが，情報家電ネットワークの必要機能を満たすかどうかを評価する．評価項目を以下に示す．

分散機能: ネットワーク上に分散した個々のサービス間で自律的に制御機構を構築できる．さらに家庭内の情報家電機器へ，インターネットを利用して，外部から操作を可能とする．

検索可能機能： イベント生産サービス，イベント消費サービス双方から検索，様々な検索条件を用いた自由度の高い検索に対応する．

耐障害機能 ：サービスの停止や，サービスの発生に対応する．

動的変更機能 ：イベントの通知先を動的にユーザによって変更可能にする．

簡易設定機能 ：イベントの通知先をユーザに容易変更可能にする．

分散機能は，遠隔サービス利用機能と分散サービス間イベント配送機能に機能分類される．検索可能機能は，イベント消費サービス検索機能，柔軟なサービスの検索機能に機能分類される．耐障害機能は，サービスの動的発見機能の一部である．動的変更機能は，イベント通知先動的変更機能と同値である．表の様にkmsf-DRAGON/TKが情報家電ネットワークに必要な機能をすべて満たしたモデルであると言える．

6.3 本章のまとめ

本章では，kmsf-DRAGON/TKを用いたシステムの定量的評価を行ない，kmsf-DRAGON/TKのイベントバインディングの速度及び，イベント配送の速度は想定している情報家電ネットワークにおいて機能を果たせる速度であることをしめした．本章の後半では，kmsf-DRAGON/TKを関連するイベントモデルと機能比較し，定性的的評価をした．kmsf-DRAGON/TKは，他のイベントモデルと比較しても機能的に優れているており，情報家電ネットワークに必要な機能をすべて満たしていることを示した．特に，Jiniとの比較においてイベント消費サービス検索機能，簡易設定機能，イベント通知先動的変更機能などの機能が充実したことを示した．

表 6.4: 関連イベントモデルとの機能比較

評価項目	Beans	ECJ	Jini	Dragon
分散機能				
遠隔サービス利用機能	×	○	○	○
分散サービス間イベント配送機能	×	○	○	○
耐故障機能				
サービスの動的発見機能	×	×	△	○
サービス障害の検知機能	×	×		
検索可能機能				
イベント消費サービス検索機能	○	×	×	○
柔軟なサービス検索機能	×	×	○	○
動的変更機能	△	○	×	○
簡易設定機能	○	×	×	○

第 7 章

結論

本章では，kmsf-DRAGON/TKを今後普及させるにあたっての課題を述べる．kmsf-DRAGON/TKを拡張させ以下の各機能を持たせる．音声インタフェースへの対応，Lookup Service（LUS）障害時の復旧機能の実装，競合条件下での機器制御の実現，地理的情報を基にしたサービス検索の実現，EventGeneratorの実装，イベントの通知条件の実現について説明する．最後に，まとめを行い本論文を締めくくる．

7.1 今後の課題

現在のkmsf-DRAGON/TKはごく基本的な機能しか提供していない．将来は，本研究を情報家電機器に組み込まれるプログラムとして広く使用されることを目標にしている．そのため，今後，以下のような拡張機能について予定している．

音声インタフェースへの対応

音声によりユーザからの要求を認識し，その認識に基づき情報家電機器を結びつけるプログラム拡張を予定している．5.2.4項で述べたBaseBindingUIクラスは，GUI問わずあらゆるUIに対応できるよう，抽象度を上げて設計してある．BaseBindingUIクラスを拡張させViaVoice SDK, Java Technology Edition[1] を用いてVoiceBindingUIクラスを実装する予定である．

Lookup Service（LUS）障害時の復旧機能の実装

LUSの障害は，情報家電ネットワークでは，他のサービスと同じ頻度で発生すると考えられる．本システムでは，検索部分でLUSを利用しているだけであり，一旦イベント登録されたイベント消費サービスとイベント生産サービスは，LUSが障害にあっても不具合が生じる事はない．しかしLUS障害時には，新しいサービスの登場に対処できなくなるため，障害を関知した場合自動的にLUSを起動する枠組みが必要になる．5.2.2節で述べたServiceUppperクラスに，この機能を持たせる予定である．

競合条件下での機器制御の実現

kmsf-DRAGON/TKは，4.4節で述べた様に競合の観点から，一つイベント消費サービスに対して複数のイベント生産サービスがイベント通知を行うモデルは実現できていない，しかし，一つの消費サービスに対して複数のイベント生産サービスからイベント通知が同時にあった場合，イベント消費サービスがアクション中は，ロックを用いてイベント通知を排他制御する．それによって，矛盾が発生することなく上記のモデルが実現できる．

地理的情報を基にしたサービス検索の実現

現在，ServiceUpperクラス及び，OneTemplateWatcherクラス，BaseBinderClientクラスは，5.2.1節で述べた様にLUSの検索をマルティキャスト及び，下記のlookupLocaterの配列により検索できる．そのため，同一セグメントか，ある複数の固定的なアドレス配列によってLUSを検索できる．

```
lookupLocater[]={
                new lookupLocater(‘‘jini://dali.ht.sfc.keio.ac.jp’’),
```

```
                new lookupLocater(‘‘jini://wanwan.sfc.wide.ad.jp’’),
                new lookupLocater(‘‘jini://zz023.sfc.keio.ac.jp’’),
}
```

しかし，この方式は，固定的なホスト名を指定する必要があり，情報家電ネットワークには向かない．情報家電ネットワークは,「この家の 2 階にあるイベント生産サービスの発見」など，地理的状況からサービスを保持している LUS の検知を行う必要がある．そのため，慶應義塾大学徳田研究室の楠本晶彦氏の FLUSH システム [24] と統合する予定である．

イベントジェネレータサービスの実装

情報家電ネットワークでは，イベント生産サービスであり，かつイベント消費サービスであるサービスが存在する事がある．例えばエアコンは，部屋の温度を取得するイベント生産サービスでもあり，他のサービスからのイベント通知によって ON，OFF といった操作を行うイベント消費サービスでもある．このように二つの性質を備えたサービスをイベントジェネレータサービス（Event Generator Service）と呼ぶ．イベントジェネレータサービスは，図 7.1の様に，イベントを生産する機能を OutPut モジュール部，イベントを消費する機能を InPut モジュール部とし GUI 上で表示する．マウスで任意の OutPut モジュールから任意の InPut モジュールへラインを引くことにより，サービス間の連続したイベントの流れを実現する．また，各モジュール部をユーザがクリックすると，サービスに関するより詳細な設定を可能とする GUI を表示する予定である．GUIによる設定予定している項目を以下に示す．

OutPut モジュール イベント発生の一時停止

OutPut モジュール イベント通知先の最大数

InPut モジュール イベント消費の一時停止

InPut モジュール イベント通知元の最大数

イベントの通知条件の実現

通常，イベント通知はイベント生産サービスからイベント消費サービスへ直接通信を行われる．その際，イベント通知を第三者のサービスであるイベント配送サービスに行い，そのサービスからイベント消費サービスへイベント通知を行う事によって，様々な条件に応じたイベント配送を実現できる．例えば，以下のようなイベント通知が実現できる．

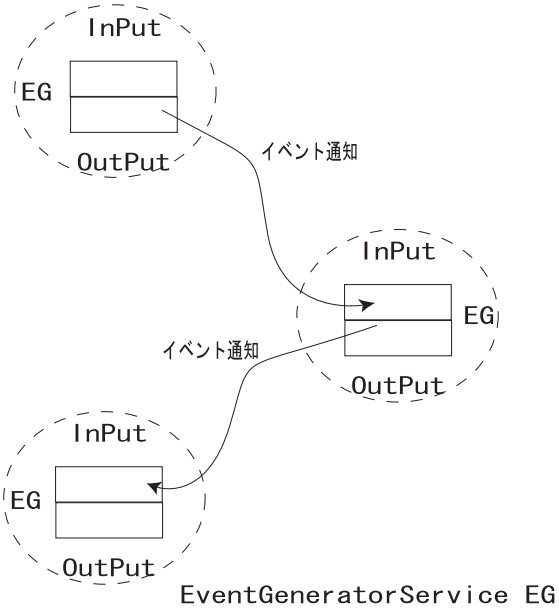

図 7.1: Event Generator Service の概要

タイマ型イベント通知

設定された時刻にしたがってイベント生産サービスから受信したイベントをイベント消費者に配送するサービスを構築した場合,「ユーザが部屋を出たのを検出したら 60 秒後にライトを消す.」といった管理の仕方が可能になる.

メールボックス

イベントメールボックスとは，ある一定期間イベントを保持し，イベント消費サービスの要求 (request) に応じて，イベントの配送を行う仕組みである．この仕組みにより，図 7.2の様に，イベントの通知をイベント消費サービスの request があるまで待機できる．これにより，イベント消費サービスの機器操作に対する排他制御，待ち行列の管理，負荷分散が行える．MailBox に複数のイベント消費サービスの request を登録することにより，イベント生産サービスは，不特定な複数のイベント消費者サービスグループに対してイベント通知を行うことができる.

イベントフィルタボックスによるフィルタリング機能の実現

イベントフィルタボックスとは，図 7.3の様に，イベントに対してフィルタリング機能をもったフィルタモジュールにより，イベントの通過，通知先などを制御する仕組みである．たとえば，情報家電ネットワーク外部からのイベント伝搬を完全にシャットアウトしたい場合,「外部からのイベントの侵入を禁止する」と規定されたポリシを保持したサービスを FilterBox に登録事ができる．また，図 7.3の様にイベント通知先

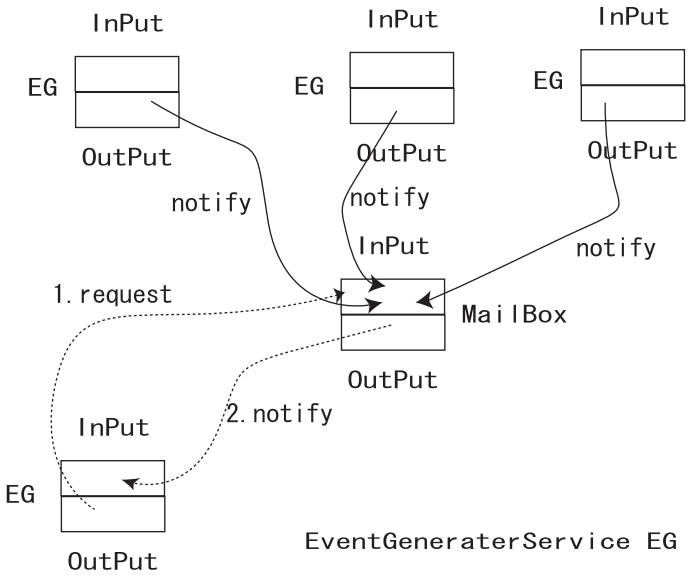

図 7.2: イベントメールボックスの概要

をある条件によって，切り替える事も可能にする．例えば,「風呂の温度がある摂氏 45 度を越た場合イベントを機器 A に配送する，摂氏 35 度を下回るとイベント機器 B に配送する」といった場合に有効になる．

メディア変換フィルタの実装

イベントには，4.4節で述べた様に様々な，メディアデータをいっしょに伝搬させる事ができる．そのため，HTML，GIF，MPEG，AU，WAV など様々なメディアを伝搬させる可能性がある．しかし，Event を消費する側は，その様なメディアにすべてが，対応しているとは限らない，そこで将来，慶應義塾大学徳田研究室の由良淳一氏が開発している tranService[23] 内のメディアデータ変換サービスを将来本システムのサービスに適応させ，イベントに含まれるメディアデータをイベント消費サービスが要求するメディアに変換する機構を構想中である．

7.2 まとめ

本論文では,『情報家電ネットワークに適したサービス間のイベント通知機構の構築』について述べた．既存の情報家電機器の制御には PC を中心とした集中管理によって，一方的にユーザが機器に対して行っていた．そのため，情報家電機器同士による自律的な制御は実現されていなかった．将来，急速な発展が見込まれる情報家電ネットワークは，情報家電機器と情報家電機器が状態の変化（イベント）を人間を介さずやり取りする場面が想定される．そのために必要となる機能は，分散機能，検索可能機

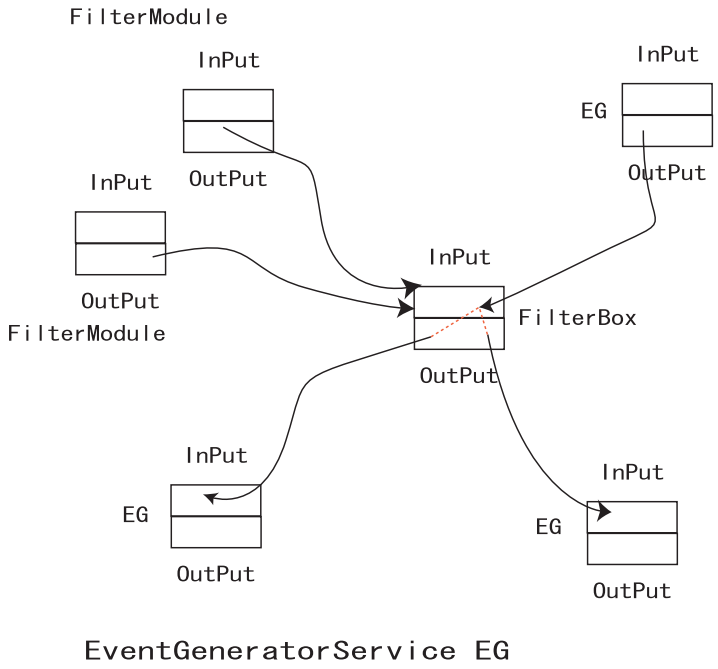

図 7.3: イベントフィルタボックスの概要

能，耐障害機能，動的変更機能である．本研究ではこれらの問題点に対して有効な動的分散イベント通知モデル (kmsf-DRAGON/EM) を提案した．また kmsf-DRAGON/EM を基にした，システムの構築を支援するツールキット (kmsf-DRAGON/TK) を構築した．さらに，同ツールキットの設計，実装，評価を述べた．本研究によって今までユーザの手を介して行なってきた情報家電機器制御が情報家電機器間で相互に行なわれるようになり，その設定も容易にユーザが行うことができる．情報家電機器をユーザが自由に組み合わせる環境が構築可能になり，ユーザが思い描く新しい役割を情報家電機器間に動的に割り振る事も可能になる．

謝辞

本研究を進めるにあたり，絶えず懇切丁寧な御指導を賜わりました，慶應義塾大学環境情報学部教授徳田英幸博士に深く感謝いたします．

慶應義塾大学徳田・村井・楠本・中村研究会の諸先輩方には，お忙しい中，貴重な示唆や御助言、御指導を頂きました．特に慶應義塾大学政策・メディア研究科修士2年の中澤仁氏，大越匡氏，修士1年の藤村浩士氏，牧野聡氏には，本論文執筆にあたって絶えざる励ましと御指導を賜りました．また、Keio Media Space Family-Architecture(KMSF)研究グループの学部4年楠本 晶彦氏，由良 淳一氏，原嶋 章介氏，中村 勇樹氏および学部2，3年のメンバーには，本研究に関する様々な議論，助言をして頂きました．また，Mobile Communication Next Generation(MCng)研究グループの学部4年荒井 仁氏，Engineering approach to Computer Networks(ECN)研究グループの学部4年西野秀明氏は，本論文執筆中の行き詰まりを打開する対戦相手としてなっていただきました．さらに，修士2年の坪井　孝弘氏は，卒論に打ち込める優れたマシンを提供していただきました．みなさまに，深い感謝の意を表します．

2000年 1月 31日
岩井 将行

参考文献

[1] IBM . the viavoice sdk, java technology edition v1., November 1998. http://www-4.ibm.com/software/speech/.

[2] Hewlett-Packerd Company. The chai appliance platform, 2000. http://www.internetsolutions.enterprise.hp.com/chai/index.html.

[3] 3Com Corporation. Palm iiie, 2000. http://www.palm.com/.

[4] S. DEERING. Host extensions for ip multicast routing, 1989. RFC-1112.

[5] R. Fielding, J. Gettys, J. Mogul, H. Frysty, and T. Berners-Lee. , 1997. RFC-2068.

[6] IEEE. Standard for a high performace serial bus, 1999.

[7] Sun MicroSystems Inc. Javabeans specification, 1997. http://www.java.sun.com/products/embeddedjava/.

[8] Sun Microsystems Inc. Thek virtual machine, 1998. http://www.java.sun.com/products/kvm/wp/.

[9] Sun MicroSystems inc. Embeddedjava application environment, 2000. http://www.java.sun.com/products/embeddedjava/.

[10] J.Gosling, B.Joy, and G.Steele. The java language specification, 1999.

[11] Tim Lindholm and Frank Yellin. . Sun Microsystems Inc., 1996.

[12] Microsoft, Corp. Universal plug and play device architecture reference specification, 1999.

[13] NTT. Asdl, 2000. http://www.ntt-east.co.jp/senyo/ADSL/index.html.

[14] NTT. Isdn, 2000. http://www.ntt-east.co.jp/ISDN/.

[15] Sony, Matsushita, Philips, Thomson, Hitachi, Toshiba, Sharp, and Grundig. Specification of the Home Audio/Video Interoperability (HAVi) Architecture, May 1998. http://www.havi.org/home.html.

[16] Sun Microsystems Inc. Remote method invocation specification, 1996.

[17] Sun Microsystems, Inc. Jini Architecture Specification, November 1998. http://www.javasoft.com/products/jini/specs/jini-spec.pdf.

[18] The Home Phoneline Networking Alliance. Simple, high-speed ethernet technology for the home, June 1998.

[19] Universal Plug and Play Forum. Universal Plug and Play (UPnP), 1999. http://www.upnp.org.

[20] World Wide Web Consortium. Extensible markup language(xml) 1.0, February 1999.

[21] 横河電機株式会社. Duonus, 2000. http://www.yokogawa.co.jp/DUONUS.

[22] 経済企画庁調査局景気統計調査課. 消費動向調査年報，四半期報, 1999.

[23] 由良淳一, 中澤仁, 大越匡, 徳田英幸. 端末およびサービス透過的な情報共有支援システムの構築. 全国大会. 情報処理学会, September 1999.

[24] 楠本大越 匡, 中澤仁, 徳田英幸. 位置情報に基づいたサービスのグループ化と透過的利用環境の構築. 全国大会. 情報処理学会, September 1999.

[25] 東芝研究開発センタコンピュータネットワークラボラトリ. Event centric for java, 1999. http://www2.toshiba.co.jp/ecj/.