2011 年度
修士論文

ホームネットワークシステムにおけるサービス環境競合問題の研究

神戸大学大学院工学研究科
情報知能学専攻

090T504T 池上 弘祐

指導教員	中村匡秀 准教授 まつ本真佑 助教
審査教員	主査 上原邦昭 教授 副査 羅志偉 教授 中村匡秀 准教授

2012 年 2 月 6 日

Formulation and Evaluation of Environment Feature Interactions in Home Network System

Kousuke Ikegami

Abstract

The integrated services of the home network system (HNS) orchestrate multiple networked home appliances to achieve value-added and comfortable services for home users. If multiple services are used at the same time, functional conflicts may occur among the services, which results in unexpected behaviors. This is known as feature interactions (FIs). We defined two kinds of FIs : appliance interactions and environmental feature interactions(EFIs). Due to lack of the degree of environmental impact and explicit consideration of requirements, the previous formalization tends to overestimate many acceptable cases as feature interactions.

To capture the EFIs more precisely, this paper introduces an *environment impact model*, describing how much impact is given to the environment by each appliance operation within a service. We also describe user requirements by environment properties evaluated within the model. Then, the EFIs is formalized as the unfulfilled requirement caused by the use of multiple services. we implement EFIs Detecting System with proposal method, and evaluate proposal method.

ホームネットワークシステムにおけるサービス環境競合問題の研究

池上 弘祐

概要

ホームネットワークシステム (HNS) を利用して，複数の家電を協調動作させる家電連携サービスの研究が進んでいる．単体では正常に動作する連携サービスを複数実行したとき，それらのサービスの持つ要求が干渉・衝突を起こし，ユーザの意図しない振る舞いが生じることがある．これをサービス競合と呼ぶ．我々は，先行研究にてサービス競合に「機器競合」と「環境競合」の 2 種類が存在することを述べた．このうち，機器競合についての定式化を提案し，検出解消法の実装を行ったが，環境競合については定式化および検出法についての十分な考察が行われてこなかった．

本稿では，HNS 機器が環境に与える影響をより正確に捉えるため，**環境インパクトモデル**を導入する．これは，連携サービスの各機器操作が，環境にどれだけのインパクトを与えるのかを記述するものである．加えて，サービスの環境的要求・制約を環境インパクトモデルを用いて記述し，環境競合を再定式化する．また，提案方式を用いたサービス競合検出システムを実装し，提案方式の評価を行う．

目次

図目次

表目次

第 1 章

はじめに

家庭内の家電機器やセンサをネットワークに接続し，付加価値サービスを実現する *HNS*（*Home Network System*）の研究，開発が盛んに行われている．HNS では，テレビ，照明，エアコン，カーテンといった家電機器や，温度計．照度計，騒音計といったセンサをネットワークに接続し，家庭内外から遠隔制御したり，連携制御することで，より便利で快適なサービスをユーザに提供する．

図 1.1 に HNS の構造を示す．各家電やセンサの持つ機能は，API (Application Program Interface) としてネットワーク上に公開され，利用者は様々な外部アプリケーションからネットワーク越しに各家電やセンサを連携・協調動作させて，付加価値の高い**家電連携サービス**を実現することができる．以下に連携サービスの例を示す．

- *TVTheaterService (TVT)*：テレビ・カーテン・室内照明を連携制御することで，映画館のような雰囲気で TV の視聴を行うことが出来るサービス．ユーザがサービスの実行を要求すると，照明が暗くなり，カーテンが閉まり，テレビがつく．
- *ComingHomeService (CH)*：ユーザの帰宅時に廊下照明・室内照明・アロマポットを連携制御し，ユーザを迎える．
- *BGMService (BGM)*：生活の背景に音楽プレーヤーを用いて小さい音量でバックグラウンドミュージックを流すサービス．
- *AirCleaningService (AC)*：空気清浄機を用いて空気清浄，脱臭を行うサービス

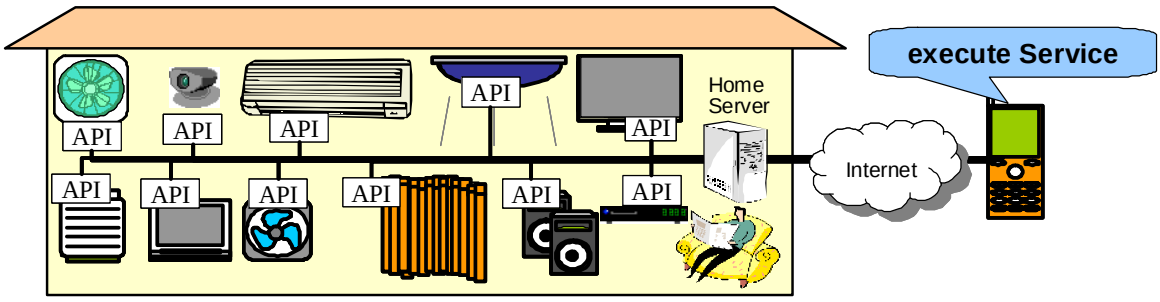

Fig 1.1. Home Network System

これら多様な連携サービスが提供されている環境においては，単体で正常に動作する連携サービスであっても，複数の連携サービスを同時に実行した場合，機器や環境に対するサービスの要求が干渉・衝突を起こし，ユーザの意図しない動作が発生することがある．これを**サービス競合**[1] [2] と呼ぶ．

以下にサービス競合の例を示す．

- 競合例 1 (CH vs TVT) ComingHome サービスと TVTheater サービスを同時に実行した場合，ComingHome サービスによって点灯していた室内照明が，TVTheater サービスの実行に伴って消されてしまう．
- 競合例 2 (CH vs TVT) CominHome サービスと TVTheater サービスを同時に実行した場合，ComingHome サービスによる廊下照明の点灯により室内が明るくなってしまい，TVTheater サービスの提供する映画館のような雰囲気での TV の視聴が阻害される．
- 競合例 3 (TVT vs BGM) TVTheater サービスと BGM サービスを同時に実行した場合，BGM サービスの提供する音楽によって，TVTheater サービスの提供する TV の視聴に適した環境が阻害される．
- 競合例 4 (CH vs AC) ComingHome サービスと AirCleaner サービスを同時に実行した場合，ComingHome サービスの提供するリラックスする香りが AirCleaner サービスにより脱臭され，効果が薄れてしまう．

我々は，先行研究 [3] において，**機器競合**と**環境競合**の 2 種類のサービス競合を定義した．

機器競合は，異なる連携サービスが同一の機器に対して，電源の on と off 等，同時には実行できない機器操作を要求した場合に発生する競合である．競

合例 1 は，室内照明において発生した機器競合である．

対して，環境競合は，異なる連携サービスが異なる機器に対してそれぞれ機器操作を要求したとき，それらの要求が HNS をとりまく環境を介して間接的に矛盾・干渉する場合に発生する競合である．競合例 2，3，4 は，明るさ，音楽コンテンツ，匂いといった環境を介してサービス同士が干渉・衝突を起こしている環境競合である．

我々は，先行研究において HNS をモデル化し，機器競合と環境競合の検出解消法を提案した．しかし，先行研究における環境競合の定義には幾つかの問題があり，環境競合を誤検出してしまうケースや，検出できないケースが多く存在した．本稿では，HNS における環境競合をより正確に検出するため，機器操作が環境に与える多様な影響を分類，モデル化し，環境競合を検出する新たな手法を提案する．また、提案法を導入したサービス競合検出システムを用いて、既存の連携サービス間の競合検出を行い、提案法の評価を行う．

第 2 章

準備

2.1 ホームネットワークシステム (HNS)

ホームネットワークシステム (*HNS*) は，家庭内のネットワークに接続された，複数の家電機器及びセンサから構成される．接続された家電及びセンサは，制御 API を備えており，ユーザや外部エージェントがネットワーク越しに機器を操作することを可能としている．我々は，田中らの提案する手法 [5] を元に，研究室内に実際の HNS(CS27-HNS と呼ぶ) を構築している．この手法に基づき，赤外線を利用した従来家電を，PC に接続可能な学習リモコンを用いて制御することで，本来機器ベンダに依存する赤外線レベルのプロトコルをカプセル化し，各機器の持つ機能をベンダ非依存の「サービス」という単位でまとめて，ネットワーク上に公開している．

CS27-HNS では，TV，Curtain，MusicPlayer，Light，AirCleaner，Aroma といった家電機器の機能を，ネットワークを介して利用することができる．CS27-HNS は，Apache Axis2 WebService を用いて実装しており，各機器操作は，SOAP もしくは REST 方式で利用することができる．

2.2 HNS モデル

先行研究 [3][9] において，我々は，HNS におけるサービス競合問題をオブジェクト指向に基づきモデル化し，サービス競合の検出・解消法の提案を行っている．また，松尾らは，環境への影響として新たに影響の向きを考慮した検出法の提案 [6] を行っている．図 2.1 に先行研究における機器モデル，環境モデル，連携サービスモデルを示す．機器モデルは，機器状態を表す機器プロパ

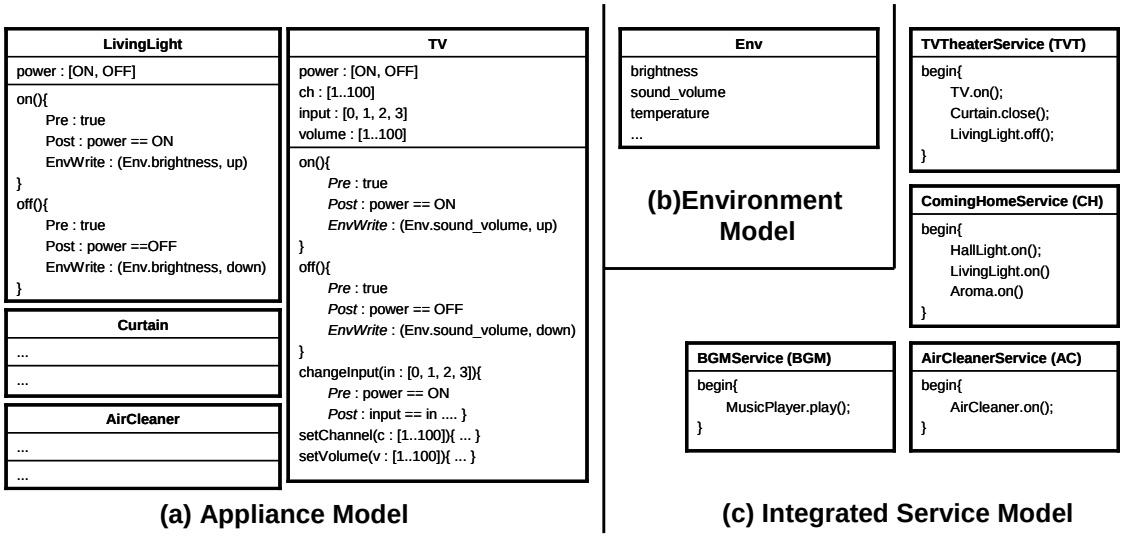

Fig 2.1. HNS Model in preceding work

ティと，機器操作を表すメソッドを持つオブジェクトであり，各メソッドは，以下の要素で定義される．

- Pre：事前条件．メソッドの実行に必要となる機器の状態に関する条件
- $Post$：事後条件．メソッドの実行後に成立する機器の状態に関する条件
- $EnvRead$：メソッドによって参照される環境プロパティの集合
- $EnvWrite$：メソッドによって更新される環境プロパティと影響の向きの対の集合

．図 2.1-(a) の TV の機器モデルを例にとると，TV は，power,ch,input,volume という機器プロパティを持ち，power は値として ON(電源 ON) か OFF(電源 OFF) をとると定義されている．また，機器メソッド on() は，メソッド実行後には，事後条件に基づいて Power が on であることが必要と定義されており，同時に環境プロパティの Env.sound_volume に対して up 方向への更新を行うと定義されている。

環境モデルは，HNS が存在する空間の環境属性 (**環境プロパティ**) を持つオブジェクトとしてモデル化される．代表的な環境プロパティには，室温 (Env.temperature)，音量 (Env.sound_volume)，明るさ (Env.brightness) 等が存在する．

連携サービスモデルは，サービスが実行するメソッドの系列によって表現される．図 2.1-(c) の TVT サービスは，TV を付ける，カーテンを閉める，室内照明を消す，という 3 つの機器操作の系列より構成されている．

2.3　機器競合

先行研究では, 上記モデルに基づいて機器競合を以下のように定義している．

定義 1 (機器競合)
s_1, s_2 を任意の連携サービス，m_1, m_2 をそれぞれ s_1, s_2 に含まれる任意の機器メソッドとする．このとき，「s_1 と s_2 が**機器競合を生じる**」とは，同機器内のメソッド m_1，m_2 が存在して以下の条件が成り立つときである．

- 条件 A1：$Post(m_1) \wedge Post(m_2) = \perp$ または,
- 条件 A2：$Post(m_1) \wedge Pre(m_2) = \perp$

直感的には，二つのサービスが同一機器内で目的の矛盾するメソッドを実行する (条件 A1) か，一方のメソッド実行により他方のメソッドが実行できなくなる (条件 A2) かで，機器競合を性質づけている．

1 章で例示した競合例 1(CH vs TVT) の機器競合は，「LivingLight.on()」の事後条件「Power = true」と，「LivingLight.off()」の事後条件「Power = false」が相容れないために発生する．

我々はサービス競合を解消するためのアプローチとして，サービス優先度を取り入れている．各サービスには予め優先度 $pri(s)$ が与えられており，s_1 と s_2 が競合した際に，もし $pri(s_1) > pri(s_2)$ ならば，s_1 を優先して実行し，$pri(s_2)$ を中断または終了させる．

[9] において我々は，機器競合をサービス実行時にオンラインで検出・解消する方法を提案している．オンライン検出・解消とは，サービスの新規実行時に，現在実行中のサービスと比較することで競合が発生するか調べ (検出)，可能であれば競合を回避する操作 (解消) を行うことである．この実現にあたり，我々はサービス期間，必須メソッド，サービスの中断・再開の概念を導入している．

2.3.1　サービス期間

サービス競合を実行時に厳密に検出するために，全ての連携サービスを，サービスの有効期間という観点から以下の 3 種類に分類する．

BeginEnd 型　：　ユーザがサービスを開始したのち，明示的にサービスを終了するまで有効であるサービス．

Timer 型　：　ユーザがサービスを開始したのち，一定時間で有効期間が切れるサービス．

Instant 型　：　ユーザがサービスを開始したのち，瞬時に有効期間が切れるサービス．

BeginEnd 型，Timer 型はサービスの有効期間中であれば，新規実行されるサービスとの競合検出の対象となるが，Instant 型は有効期間が一瞬しか存在しないため，競合検出の対象としない．

2.3.2　必須メソッド/通常メソッド

連携サービスを構成するメソッドのうち, そのメソッドを欠くことで, サービスの主目的が達成できなくなるメソッドを必須メソッドと定義する．それ以外のメソッドを通常メソッドと定義する．あるサービス s のメソッド m が実行可能であるための条件は,「m と競合し, 実行中であり, 且つ $pri(m) < pri(m_a)$ であるメソッド m_a が存在しない」ことであると定義する．そして，新規実行するサービス s_new が実行可能であるための条件は,「s_new に含まれる全ての必須メソッドが実行可能である」ことであると定義する．つまり，s_new に優先するサービス s_a が既に実行中であり，s_new に含まれる必須メソッドが一つでも実行できないとき，s_new は実行不可能となる．一方，s_new に含まれる通常メソッドのみが実行不可能であるとき，s_new のサービス機能を縮退して実行することができる．

2.3.3　サービスの中断/再開

実行中であるサービス及びメソッドが，他のサービスとの競合により終了されたとき，メソッドが再開可能なものであれば，終了はせずに中断状態となる．中断状態のメソッドは，再び自らが実行可能な環境になったとき，自動的に再開する．

2.3.4　機器競合検出・解消の流れ

機器競合検出・解消の流れは以下のものとなる．

開始条件　：新規に連携サービス s_new が実行される．
Step1　：s_new と実行中サービスとの機器競合を検出する．
Step2　：s_new が実行可能であるか検証する．
Step3　：s_new が実行可能であった場合，s_new の実行に伴う他メソッドの終了・中断・再開を行う．
Step4　：HNS 機器への操作制御を行う．

Step1 では，連携サービスモデルを利用し，条件の比較を行い，競合する全てのメソッドのペアを検出する．Step2 では，検出した結果を用いて，s_new の全ての必須メソッドが実行可能であるかの検査を行う．Step3 では，s_new が実行可能であった場合に，s_new との競合によって終了・中断するサービス・メソッドや，更にそれに伴って再開するサービス・メソッドの状態の更新を行う．Step4 では，s_new の実行および Step3 で状態を変更した機器について，HNS 機器に操作命令を発し，実際の機器の制御を行う．

第 3 章

環境競合

3.1 従来の環境競合

先行研究において, 環境モデルは, **環境プロパティ**を持つオブジェクトとしてモデル化されている．代表的な環境プロパティには, 室温 (Env.temperature), 明るさ (Env.brightness)，音量 (Env.sound_volume)，湿度 (Env.humidity) 等が存在する．

また，機器メソッドが環境に与える影響を表現するために，機器メソッドには，$EnvRead$：メソッドによって参照される環境プロパティの集合と，$EnvWrite$：メソッドによって更新される環境プロパティと影響の向きの対の集合とが定義されている．

環境競合は，先行研究モデルを用いて以下のように定義されている．

定義 2 (先行研究における環境競合)「s_1 と s_2 が**環境競合を生じる**」とは，異なる機器のメソッド m_1，m_2 が存在して以下の条件が成り立つときである..

- 条件 E1：$\exists(p_1, d_1) \in EnvWrite(m_1)$;
  $\exists(p_2, d_2) \in EnvWrite(m_2)[p_1 = p_2 \wedge d_1 \neq d_2]$ または
- 条件 E2：$\exists(p_1, d_1) \in EnvWrite(m_1)$;
  $\exists(p_2) \in EnvRead(m_2)[p_1 = p_2]$

直感的には，図 3.1 に示すよう，2 つのサービスが共通の環境プロパティを更新し，その際に影響の向きが異なる (条件 E1) か，一方が更新したものを他方が参照するとき (条件 E2) に，環境競合が生じ得るとしている．競合例 2 では,

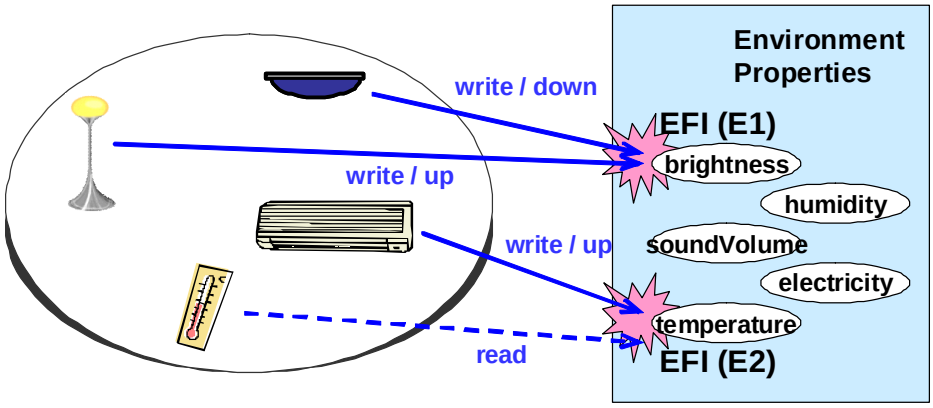

Fig 3.1. Environment Interactions in preceding work

TVT による室内照明の消灯メソッドと，CH による廊下照明の点灯メソッドが共に同一の環境プロパティ Env.brightness を更新し，両メソッドの影響の向きが異なるため，条件 E1 を満たし環境競合となる．また，温度センサ等の Env.temperature を参照する機器とは，冷房，暖房共に干渉が発生するため，条件 E2 を満たし環境競合となる．

3.2　先行研究における問題点

先行研究における環境競合の定義に関して，以下の問題が存在した．

- **問題 P1:** サービスの要求を考慮していない．
  環境競合をより正確に捉えるためには，サービスが環境に対してどのような状態を要求しているかを明示的に考慮すべきである．競合例 2 のように，TVT の明るさに対する競合は，先行研究モデルを用いると，TVT の求める室内の明るさの度合いは考慮されず，常に環境競合となる．しかしながら，ルームランプやディスプレイ等，明るさに微少な影響しか与えない機器であるならば，TVT と同時に利用することが可能であるかもしれない．
  先行研究の定義においては，明るさに対して多少の影響をもつ HNS 機器 a_1，a_2 について，a_1.on() と a_2.off() は明るさに対して異なる向きに作用し，環境競合となりえるため，ユーザの意図を超えて，環境競合が過大に検出される恐れがある．実行するサービスが求める明るさ，すなわち

サービス要求を明に考慮することで，環境競合とすべきかの判断材料にできる．

- **問題 P2:** 表現できる環境プロパティが少ない．
  先行研究における環境競合モデルは，環境に対する影響の向きのみを利用して競合を定義している．このため，冷暖房の設定温度のような，そもそも向きを持たない不可算的な影響や，調光ランプの調光操作など，影響の向きが前状態によって変化する作用を記述することができない．また，音楽コンテンツや匂いコンテンツなどの，同じ向きをした作用同士で衝突，競合が発生するような非数値的な環境プロパティを取り扱うことができない．
  例えば，競合例 3 における音楽コンテンツにおける競合を考えると，TV と MusicPlayer の音楽コンテンツに対する影響はどちらも「音楽コンテンツを提供する」という同じ向きの作用であると考えられ，先行研究における定義では環境競合と検出されない．

3.3　環境インパクトモデル

問題 P1，P2 を解決するため，機器が環境に与える影響 (環境インパクト) を詳細に表現する**環境インパクトモデル**を定義し，**サービス環境要求**に基づいた環境競合の再定式化を行う．

3.3.1　環境プロパティの分類

まず，問題 P2 における環境プロパティの表現能力を向上するため，全ての環境プロパティを，環境状態を数値として定量化することが可能かどうかという視点から，数値型と非数値型に分類する．

- 数値型 ： 値を数値で表現することのできるプロパティ．例：明るさ (Env.brightness), 音量 (Env.sound_volume), 消費電力 (Env.electricity), 湿度 (Env.humidity)，温度 (Env.temperature) など．
- 非数値型 ： 値を数値以外の要素 (文字列，ラベル等) で表現するプロパティ．例：映像コンテンツ (Env.movie_content)，音楽コンテンツ

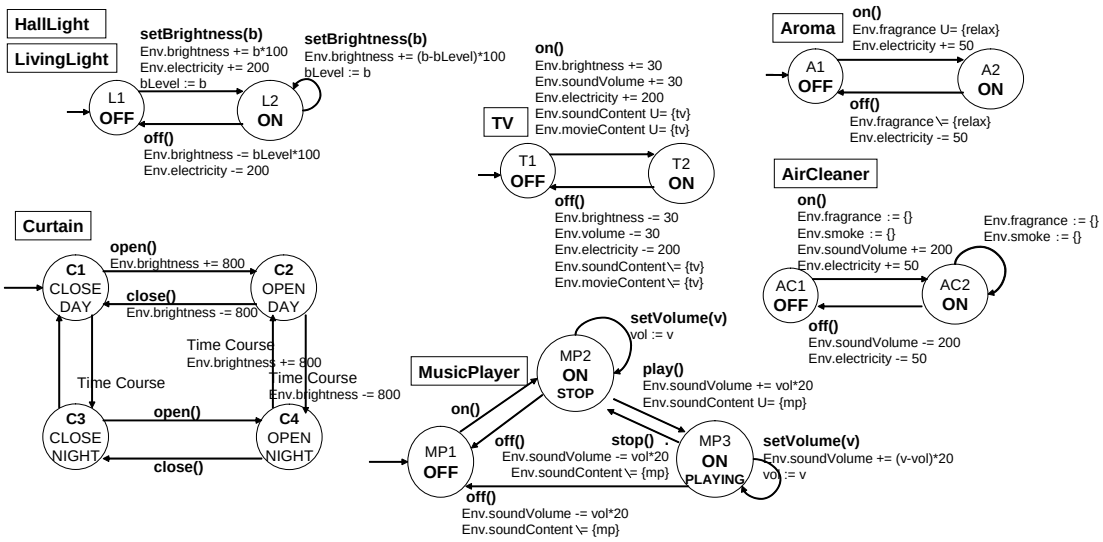

Fig 3.2. Environment Impact Model

(Env.sound_content)，匂い (Env.fragrance)，煙 (Env.smoke) など.

3.3.2　HNS 機器の環境インパクトモデル

次に，機器が環境に与える影響を詳細に表現するため，状態遷移機械 (FSM) による**環境インパクトモデル**を導入する.

定義 3 (環境インパクトモデル) E を環境プロパティの集合, d を HNS 機器としたとき，d の環境インパクトモデルを，以下の状態遷移機械 I_d で定義する.

$I_d = (S_d, M_d, T_d, \Delta, s_0)$

- S_d は d の状態の集合,
- M_d は d の持つ機器メソッドの集合,
- $T_d \subseteq S_d \times M_d \times S_d$ は状態遷移の集合,
- $\Delta : T_d \times E \rightarrow Z$ は環境インパクト関数,
- s_0 は初期状態.

環境インパクトモデルはメソッド実行による機器の状態遷移をモデル化したものである．各状態遷移が環境プロパティに対し，どれだけインパクトを持つかを表す．図 3.2 に，本稿で利用する HNS 機器の環境インパクトモデルを

示す.

例えば TV は OFF($T1$),ON($T2$) の 2 状態を持ち，メソッド on() の実行によって $T1$ から $T2$ への遷移が起こる．遷移に伴って生じる環境インパクトは以下の 5 つである，

- $Env.brightness \mathrel{+}= 30$
- $Env.sound\_volume \mathrel{+}= 30$
- $Env.electricity \mathrel{+}= 200$
- $Env.sound\_content \cup = \{tv\}$
- $Env.movie\_content \cup = \{tv\}$

ここで，+=,-=はそれぞれ加算，減算演算子，$\cup =$ は集合への要素追加演算子を表す．この環境インパクトは，メソッド on() の実行に伴って，明るさ (Env.brightness) に +30, 音量 (Env.sound_volume) に +30, 消費電力 (Env.electricity) に +200, 音楽コンテンツ (Env.sound_content)，映像コンテンツ (Env.Movie_content) に tv が追加される作用を表している．

数値プロパティには数値型のインパクトのみが与えられ，非数値プロパティには非数値型のインパクトのみが与えられる．

3.3.3　環境インパクトの分類

競合例 4 において，AC サービスの持つ機器操作 AirCleaner.clean() は，他の機器が提供する匂いを全て清浄し，空間環境をある一定の状態へと置き換える作用を持っている．このような作用は，これまでに述べた可算的なインパクトだけでは記述ができない．よって，影響が可算的であるか否かという視点より，環境インパクトを可算インパクトと不可算インパクトに分類する．

組み合わせにより, 全ての環境インパクトは以下に示す 4 種類に分類される．

- **数値型かつ可算的** : 数値型かつ可算的なインパクトは，任意の数値型環境プロパティに対して，一定値を加減算 (+=,-=) する作用を持つ．明るさ (Env.brightness) や，音量 (Env.sound_volume)，消費電力 (Env.electricity) に対する環境インパクトは，一般的に数値型かつ可算的である．これらのインパクトの値は，機器操作がそれぞれの環境に与え

る影響の大きさを，その意味を損なうことなく加減算ができるよう正規化した数値である．そのため，インパクトの値は，必ずしも明るさ，音量等の実際の変動値と一致するものではない．

- **数値型かつ不可算的**：数値型かつ不可算的なインパクトは，任意の数値型環境プロパティの値を一定値に置き換える (:=) 作用を持つ．冷房を付け，室温が設定温度の温度の 28 ℃となる作用を考えるとき，この作用は数値で表現することが出来るが，作用同士を足し合わせる事は不可能である．このような作用を，数値型かつ不可算的なインパクトとして，”$Env.temperature := 28$”と記述する．
- **非数値型かつ可算的**：非数値型かつ可算的なインパクトは，任意の非数値型環境プロパティに対して，コンテンツを put，remove($\cup =, \setminus =$) する作用を持つ．アロマポットによるリラックスする香りを提供する作用”$Env.fragrance \cup = \{relax\}$”が該当する．
- **非数値型かつ不可算的**：非数値型かつ不可算的なインパクトは，任意の非数値型環境プロパティの値を，一定の集合に置き換える (:=) 作用を持つ．空気清浄機による匂いの清浄化作用”$Env.fragrance := \{\}$”が該当する．

3.3.4　環境作用

連携サービス s によって複数の機器操作が実行されていくと，環境に対して様々なインパクトが加えられていく．ここで，環境プロパティごとに環境インパクトの累計を計算し，連携サービスの環境に対する作用 (**環境作用**, Environmental Effect) と定義する．

定義 4 (環境作用，排他環境作用) s を連携サービス，$[t_1, t_2, ..., t_x]$ を s の実行によって起こった環境インパクトモデルの遷移系列とする．この時，ある環境プロパティ $e \in E$ に対する s の環境作用 $EE(s, e)$ を以下のように定義する：

$EE(s, e) = \sum_i^x \Delta(e, t_i)$

また，$[t'_1, t'_2, ..., t'_m]$ を現在実行中の全ての連携サービスの実行によって起こった遷移系列とする．この時，特定のサービス s によらない環境作用 $EE(e)$

を以下のように定義する：

$EE(e) = \sum_i^m \Delta(e, t'_i)$

さらに，連携サービス s を除く連携サービスによる EE を排他環境作用と呼び，以下のように定義する．

$XEE(s, e) = EE(e) - EE(s, e)$

非数値プロパティの環境作用 EE は，文字列値の集合である．例えば，機器 Aroma による”$Env.fragrance\cup = \{relax\}$”のみが与えられているときの $EE(Env.fragrance)$ は，$\{relax\}$ となる．

同一の環境プロパティに，複数の不可算インパクトが与えられている状況を考える．実世界における環境状態は，影響を与えている機器の強弱関係に従って 1 状態に収束するため，一部の不可算インパクトが期待する環境状態と実世界における環境状態には乖離が生じる．しかしこの乖離は，全てがユーザの不利益や興味の対象となるわけではなく，直ちに環境競合であるとは言えない．

例えば, 異なる冷房機器が同一空間に存在し, それぞれが “$Env.temperature := 28''$ 及び “$Env.temperature := 26''$ という冷房の設定温度に関するインパクトを与えている場合を考える．このとき，これらのインパクトが同時に提供されているだけでは環境競合とは言えず，実行されているサービスの要求や制約と比較する必要がある．このとき，環境作用 EE(Env.temperature) の値は，“$28\ or\ 26''$ と記述する．

3.4　サービス環境要求

サービス s が環境に対して求める条件を，**サービス環境要求** *(EReq)* と呼び，環境作用を用いた論理式 $(EReq(s))$ として定義する．

図 3.3 に，提案法に基づいて記述した，1 章で例示した 4 つの連携サービスの機器操作 (begin) とサービス環境要求 (EReq) を示す．図 3.3 において，TVT サービスは「1.1.1:TV を付ける」,「1.1.2:カーテンを閉じる」,「1.1.3:リビングライトを暗くする」という 3 つの機器操作から構成され,「1.2.1:辺りが静かであってほしい」,「1.2.2:辺りが暗くあってほしい」,「1.2.3:同時に他の音楽コンテンツが提供されないでほしい」,「1.2.4:同時に他の映像コンテンツ

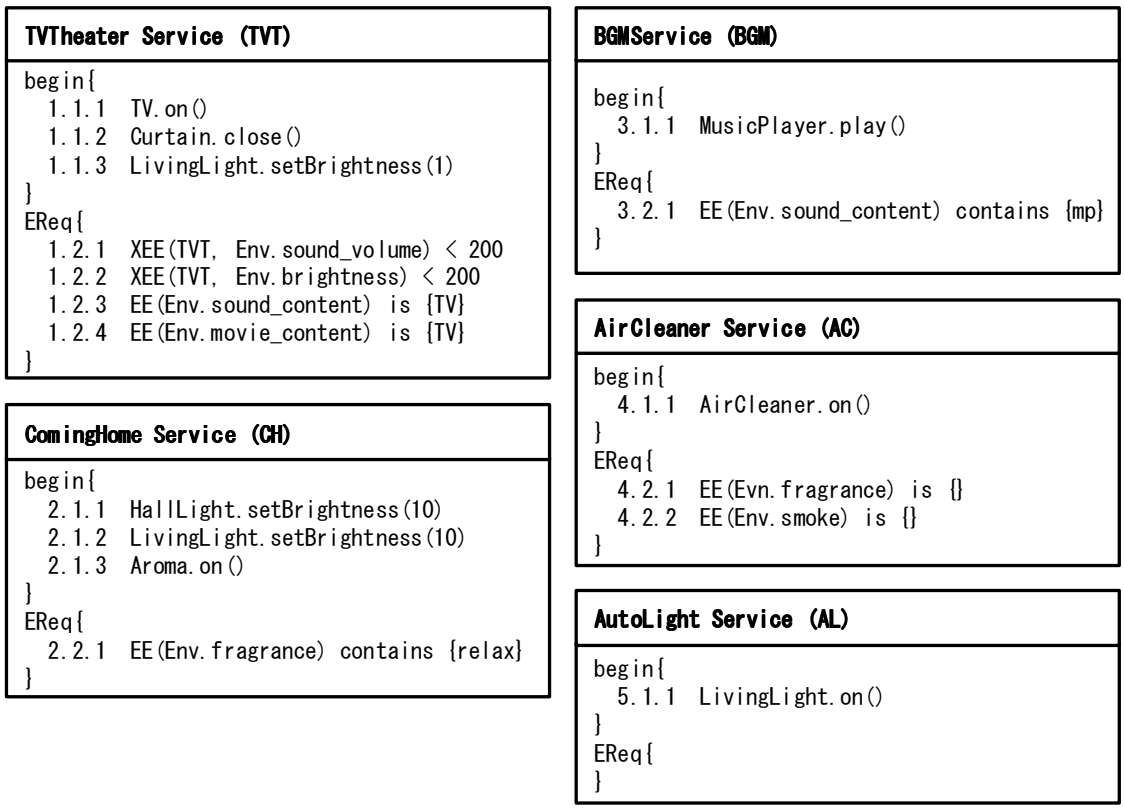

Fig 3.3. Integrated Service Model

が提供されないでほしい」という 4 つのサービス環境要求を持っている．

非数値プロパティに対するサービス環境要求は，要素の集合である環境作用 EE または XEE が，ある要素を含む集合である (contains) か，または任意の要素のみで構成される集合である (is) かという条件判定式で記述する．例えば，TVT サービスの「1.2.3:同時に他の音楽コンテンツが提供されないでほしい」というサービス環境要求は，is を用いて "$EE(Env.sound\_content)\ is\ \{TV\}$" と記述する．また，BGM サービスの「3.2.1:他サービスによる音楽コンテンツの提供の有無には関心がなく，自サービスの音楽コンテンツが提供できていればよい」という要求は，contains を用いて，"$EE(Env.sound\_content)\ contains\ mp$" と記述する．

3.5　環境制約

また，環境に求められる条件には，サービスが要求するもののほか，元来環境で定められる「使用ルール」や「電気容量」といった制約がある．このよ

うにサービスに依存せず，環境が元来持っている制約条件を，**環境制約**と呼び，環境作用を用いた論理式で定義する．例えばある家で，「同時使用電力が2000kW 未満でなければならない」，という使用ルールが決まっていたとする．この時の環境制約は，

$G = EE(Env.electricity) < 2000$

と記述する．連携サービス s の環境要求 $EReq(s)$ は通常 G を満たす範囲で与えなくてはならない．

複数の不可算インパクトが同一プロパティに与えられており，or で区切られた複数の値を持つ環境作用を含む環境要求及び環境制約を評価する場合は，区切られた値のいずれかが要求や制約を満たさないとき，要求や制約全体が満たされないものとする．

例えば前章に示した $EE(Env.temperature) = 28\ or\ 26$ に対して，“$EE(Env.temperature) < 27$″ という環境要求が与えられたとき，この環境要求は満たされず，環境競合が生じる．冷房の設定温度の場合のみを考えた場合には，より優先されるであろう設定温度値の下限値のみで比較する方法が考えられるが，他の環境プロパティを考えた場合，その性質によって，上限値や中間値での比較が適切となる場合や，文字列値同士の比較となる場合が考えられる．よって，適切な簡略化として，値全てでの比較を行い，環境競合を最大数検出し，以後の解消のステップでの解決を図る．

3.6　環境競合の再定義

サービス競合を定義するため，まず単体の連携サービスが，環境の面において正常に動作するという概念を定義する．

定義 5 (連携サービスの正常動作) 連携サービス s が正常動作するとは，環境インパクトモデルにしたがって実行された s が，環境要求 $EReq(s)$ と環境制約 G を共に満たすことと定義する．

またこのことを，$s \vdash EReq(s) \wedge G$ と書く．

次に，連携サービス s_1, s_2 を考える．s_1 と s_2 の環境競合は「それぞれは正常動作するが，両方実行すると正常動作しない」という条件になるから，以下

のように定義できる．

定義 6 (2 者間環境競合) s_1, s_2 を連携サービス，$[s_1; s_2]$ を両サービスの逐次実行 (s_1 の後 s_2 の順) とする．この時，s_1 **と** s_2 **が環境競合を起こす**とは，以下の条件が全て満たされることである．

条件 B1： $s_i \vdash EReq(s_i) \wedge G \ (i = 1, 2)$
条件 B2： $[s_1; s_2] \nvdash EReq(s_1) \wedge EReq(s_2) \wedge G$

s_1 と s_2 が環境競合を起こすとき，$EFI(s_1, s_2)$ と書く．

条件 B1 は s_1 と s_2 がそれぞれ単独で正常動作することを表す．条件 B2 は両者を実行すると，s_1 または s_2 の環境要求が満たされなくなるか，環境制約 G を満たせなくなることを表している．

環境競合は複数のサービスからの影響を表す環境作用と，環境制約，サービス環境要求との間の競合である．よって，環境競合は 2 つのサービス間に留まらず，n 個のサービス間で発生しうる．n 者間に拡張した環境競合の定義を以下に示す．

定義 7 (n 者間環境競合) $[s_1; s_2; ...; s_{n-1}]$ を現在実行中の連携サービスの系列とし，s_n を新たに実行する連携サービスとする．このとき，**実行中のサービス** $\{s_1, s_2, ..., s_{n-1}\}$ **と** s_n **が環境競合を生じる** ($EFI(\{s_1, ..., s_{n-1}\}, s_n)$ と書く) とは，以下の条件が全て満たされることである．

条件 C1： $\neg EFI(\{s_1, ..., s_{i-1}\}, s_i) \ (i = 2, 3, ..., n-1)$
条件 C2： $[s_1; s_2; ...; s_n] \nvdash EReq(s_1) \wedge ... \wedge EReq(s_n) \wedge G$

条件 C1 は 2 者間環境競合 (定義 6) を基底とする再帰的な条件となっている．$n-1$ 個のサービス間では環境競合が起こらなかったことを表している．この条件の下，条件 C2 で n 個目のサービスを実行すると環境要求または環境制約が満たせなくなってしまうということを表している．

Table 3.1. Environment Impact: Example2(CH vs TVT)

Example:2	ComingHome	TVTheater	EE
Env.brightness	+1000 (LivingLight) +1000 (HallLight)	-900 (LivingLight) +30 (TV)	1130
Env.sound_volume		+30 (TV)	30
Env.electricity	+200 (LivingLight) +200 (HallLight) +50 (Aroma)	 +200 (TV)	650
Env.sound_content		U={TV} (TV)	{TV}
Env.movie_content		U={TV} (TV)	{TV}
Env.fragrance	U={relax} (Aroma)		{relax}

3.7　ケーススタディ

1 章で例示した環境競合例について，提案した環境インパクトモデルを用いて競合検出を行うケーススタディを紹介する．

- **競合例 2 (CH vs TVT)**
  CH サービスを実行中に TVT サービスを実行する例である．表 3.1 に，CH サービスと TVT サービスの環境インパクトと，環境作用 EE の値を示す．
  CH サービスによって実行されたメソッド「HallLight.setBrightness(10)」と他メソッドの環境インパクトにより，明るさに関する環境作用は，”EE(Env.brightness) = 1130” となっている．この結果，TVT サービスの「周りが暗くあってほしい」という環境要求 “$XEE(Env.brightness, TVTheater) < 200$″ が満たされなくなり，環境競合として検出される．
  先行研究においては，サービスの要求を考慮しないために，HallLight の明るさと TVTheater サービスの求める明るさの強弱に関わらず，その全てを環境競合としており，過剰に環境競合を検出していた．
- **競合例 3 (TVT vs BGM)**

Table 3.2. Environment Impact: Example3(TVT vs BGM)

Example:3	TVTheater	BGM	EE
Env.brightness	+100 (LivingLight) +30 (TV)		130
Env.sound_volume	+30 (TV)	+500 (MusicPlayer)	530
Env.electricity	+200 (TV)	+100 (MusicPlayer)	300
Env.sound_content	U={TV} (TV)	U={mp} (MusicPlayer)	{TV, mp}
Env.movie_content	U={TV} (TV)		{TV}

Table 3.3. Environment Impact: Example4(CH vs AC)

Example:4	ComingHome	AirCleaner	EE
Env.brightness	+1000 (LivingLight) +1000 (HallLight)		2000
Env.electricity	+200 (LivingLight) +200 (HallLight) +50 (Aroma)	+50 (AirCleaner)	500
Env.fragrance	U={relax} (Aroma)	:={} (AirCleaner)	{relax} or [}
Env.smoke		:={} (AirCleaner)	{}

TVT サービスを実行中に BGM サービスを実行する例である．表 3.2 に，TVT サービスと BGM サービスの環境インパクトと，環境作用 EE の値を示す．

TVT サービスは，非数値プロパティである音楽コンテンツ (Env.sound_content) を自分だけが提供したいという環境要求 “$EE(Env.sound\_content)\ is\ \{TV\}''$ を持つ．しかし，BGM サービスが機器操作「MusicPlayer.play()」を実行することで，MusicPlayer による音楽コンテンツの提供が始まる (“$Env.sound\_content \cup = \{mp\}''$)．そのため，TVT サービスの環境要求が満たされなくなり，環境競合が検出される．

先行研究の定義においては，同じ向きの環境に対する作用は環境競合として検出することができなかった．

- **競合例 4 (CH vs AC)**

  CH サービスを実行中に AC サービスを実行する例である．表 3.3 に，CH サービスと AC サービスの環境インパクトと，環境作用 EE の値を示す．

  CH サービスは，機器操作「Aroma.on()」を実行し，リラックスする香りのインパクト “$Env.fragrance \cup = \{relax\}''$ を提供する．一方で AC サービスは，機器操作「AirCleaner.on()」を実行し，匂いを除去するインパクト “$Env.fragrance := \{\}''$ を提供する．このとき，匂いに関する環境作用は “$EE(Env.fragrance) = \{relax\} or \{\}''$ となり，CH サービスの「リラックスする香りを提供されている」という環境要求 “$EE(Env.fragrance)\ contains\ \{relax\}''$ および，「辺りが無臭である」という AC サービスの環境要求 “$EE(Env.fragrance)\ is\ \{\}''$ を満たさなくなるため，環境競合が検出される．

第 4 章

サービス競合検出システム

提案する環境競合検出法を評価するため，提案法を用いて**サービス競合検出システム**を実装した．サービス競合検出システムは，連携サービスの実行要求を受けると，機器競合の検出，解消と環境競合の検出を行ったのち，CS27-HNS にて連携サービスを実行する．システムの全体像を図 4.1 に示す．サービス競合検出システムは，先行研究 [8] である機器競合検出サービス，機器競合解消サービス，実行管理サービスと，環境状態を保持する**状態遷移機械サービス**，環境要求の評価を行う**環境競合検出サービス**より構成される．

システムを用いて連携サービス S_{new} を実行する流れをシーケンス図 4.2 に示す．

システムは，以下の流れで処理を行う．

Step1: コントローラが，S_{new} の実行命令を受け，機器競合検出サービスを利用して S_{new} の機器競合検出を行う．

Step2: コントローラが，機器競合解消サービスを利用して，S_{new} の機器競合解消を行う．

Step3: 実行管理サービスにより，CS27-HNS に対して実際の機器操作が命令される．状態遷移機械サービスの持つ環境モデルに，機器操作の環境的影響が反映される．

Step4: コントローラが，S_{new} と既に実行されている連携サービスの全てのサービス環境要求を，環境競合検出サービスと状態遷移機械サービスを用いて検査し，環境競合の検出を行う．

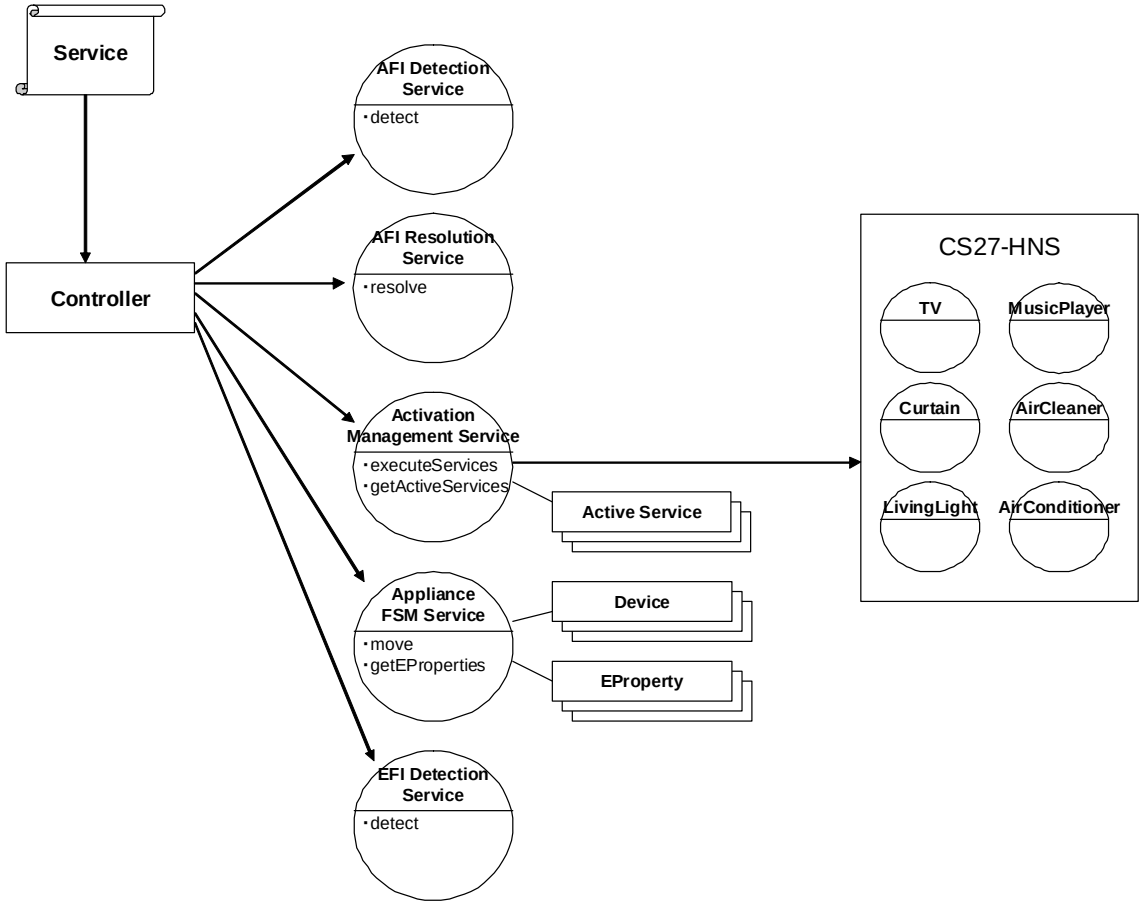

Fig 4.1. EFI Detection System

次節以降では，各サービスの詳細について説明する．

4.1　機器競合検出サービス

機器競合検出サービスは，新規に実行する連携サービス S_{new} と，現在実行中の全ての連携サービス間の機器競合を検出する．このサービスが公開するサービスメソッドは以下の通り．

detect(ServiceInfo S_{new})：ServiceInfo

引数と戻り値はともに ServiceInfo である．ServiceInfo は連携サービスに関する情報を保持しており，我々の提案するサービス競合検出システムにおける全てのサービスが共通して利用する．ServiceInfo の属性は以下の通り．

ServiceInfo

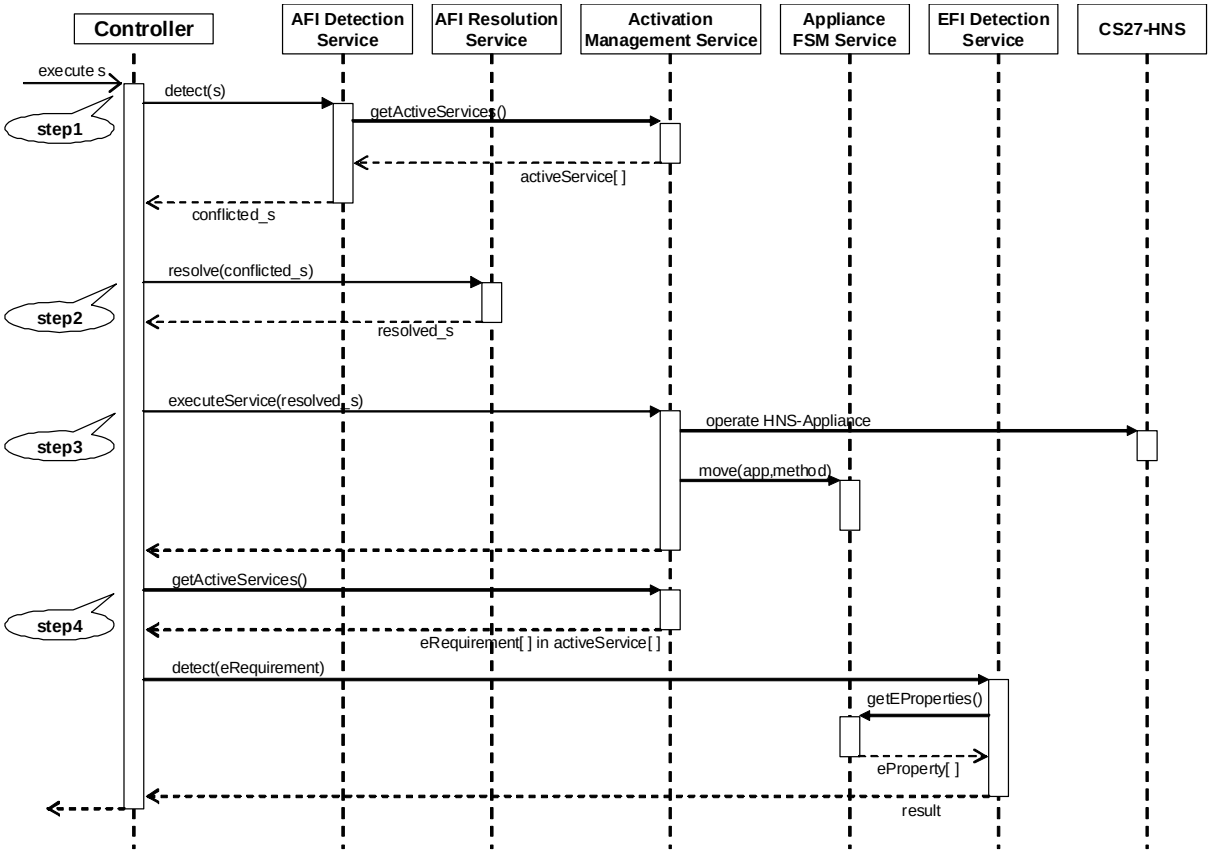

Fig 4.2. Sequence diagram : Execution integrated service

- scenario(String)：連携サービスシナリオ名.
- id(long): 連携サービスを個別に識別するための ID.
- beginMethods(MethodInfo[])：連携サービス開始時に実行される機器メソッド群.
- endMethods(MethodInfo[])：連携サービス終了時に実行される機器メソッド群.
- resumable(boolean)：再開可能性．再開可能時に再開するメソッドか否かを表す.
- eRequirements(String[])：サービス環境要求．サービス実行中に成立している必要がある，環境作用に関する条件.

detect メソッドは，実行管理サービスより現在実行中である全ての機器メソッドを取得し，S_{new} との機器競合を検査する．機器競合が検出された場合，S_{new} の $beginMethods$ に競合メソッド情報を付与し，S_{new} を戻

り値として返す.

4.2　機器競合解消サービス

機器競合解消サービスは，機器競合検出サービスによって検出された競合情報にもとづいて，機器競合解消を行うサービスである.

このサービスが公開するサービスメソッドは以下の通り.

resolve(ServiceInfo $conflictedS_{new}$**)：ServiceInfo**

引数と戻り値はともに ServiceInfo 型である．機器競合解消サービスは，競合メソッド情報を利用して競合解消を行う.

resolve メソッドは，S_{new} の持つ競合メソッド情報について，サービス優先度を利用して S_{new} と競合相手サービスのどちらを優先するべきかを決定する．解消結果に従って $beginMethods$ の状態を更新し，S_{new} を戻り値として返す.

4.3　実行管理サービス

実行管理サービスは，実行中サービス情報の管理を行う．また，競合解消結果に従って機器操作要求を CS27-HNS と状態遷移機械サービスに伝達する．このサービスが公開するサービスメソッドは以下の通り.

executeService(ServiceInfo $resolvedS_{new}$**)：ServiceInfo**

引数と戻り値はともに ServiceInfo 型である．引数の ServiceInfo は，$resolvedS_{new}$ が持つ機器メソッドに関する機器競合解消結果を保持している.

executeService メソッドは，機器競合が解消された S_{new} を受け取り，関連したメソッドの再開，中断を行い，最終的に実行可能であると判断した機器メソッドの実行要求を CS27-HNS に送る．また，状態遷移機械サービスの move(String $applianceName$, String $methodName$) メソッドを用いて，状態遷移機械サービス中の機器状態を更新する.

getActiveServices()：ServiceInfo［］ services

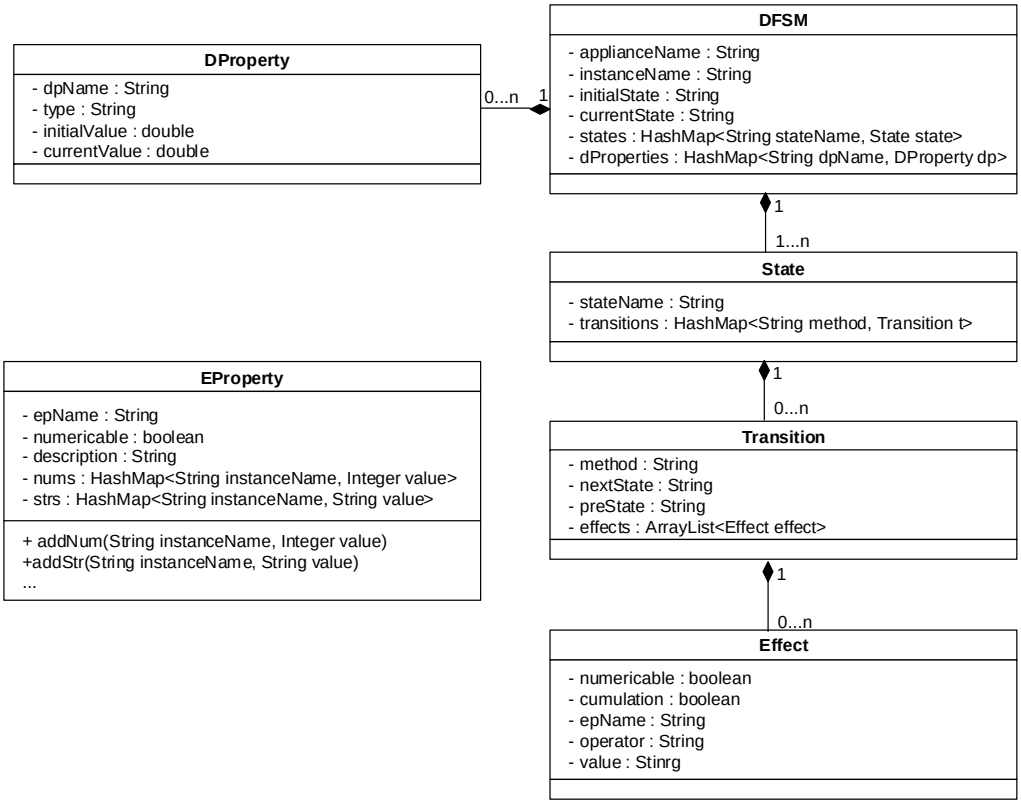

Fig 4.3. Class diagram : DFSM and EProperty

引数はなし．戻り値は現在実行中のサービスの ServiceInfo の集合である．

4.4　状態遷移機械サービス

状態遷移機械サービスは，CS27-HNS の機器状態と環境状態を保持し，機器メソッドの実行に際して，機器が環境に与える影響を算出するサービスである．

状態遷移機械サービスは，CS27-HNS に存在する機器の環境インパクトモデルオブジェクト $DFSM[\,]$ と，環境プロパティモデルオブジェクト $EProperty[\,]$ を持つ．$DFSM$ と $EProperty$ のクラス図を図 4.3 に示す．$DFSM$ と $EProperty$ の役割，及び属性は以下の通りである．

DFSM ：単一の機器の環境インパクトモデルを表現するクラス．

- applianceName(String) ： 機器クラス名．LivingLight であれば，

“Light”.

- instanceName(String)：機器を個別に識別するための名前．LivingLight であれば，“LivingLight”.
- initialState(String)：環境インパクトモデルの初期 State 名.
- instanceName(String)：機器の現在 State 名.
- states(HashMap<String stateName, State state>)：機器の全 State を格納する HashMap．key は State 名，value は State インスタンス.
- Dproperties(HashMap<String dpName, DProperty dp>)：機器プロパティを格納する HashMap．key は機器プロパティ名，value は機器プロパティインスタンス.

DProperty ：機器プロパティを表現するクラス．機器内部に記録される，TV，MusicPlayer の設定音量や，冷暖房機器の設定温度等が機器プロパティにあたる.

- dpName(String)：機器プロパティ名.
- type(String)：機器プロパティの種別.
- initialValue(double)：機器プロパティの初期値.
- currentValue(double)：機器プロパティの現在値.

State ：環境プロパティモデルの 1 状態を表現するクラス．TV を例にとると，“on”．“off” の 2 状態が存在する.

- stateName(String)：状態名.
- transitions(HashMap<String method, Transition t>)：この状態から発生しえる遷移の HashMap．key は遷移を起こす機器操作名，value は遷移インスタンス.

Transition ：環境プロパティモデルの 1 遷移を表現するクラス．TV を例にとると，“on()” メソッドによって発生する “on” 状態から “off” 状態への 1 遷移を表現する.

- method(String)：遷移を引き起こす機器操作.
- nextState(String)：遷移後の状態名.
- preState(String)：遷移前の状態名.

- effects(ArrayList<Effect e>)：機器状態の遷移によって発生する環境的影響の集合.

Effect　：環境への影響を表現するクラス.

- numericable(boolean)：影響の数値性．影響が数値で表現できるか否か.
- cumulation(boolean)：影響の可算性．影響を足し合わせできるか否か.
- epName(String)：影響先の環境プロパティ名
- operator(String)：演算子.
- value(String)：影響値.

EProperty　：単一の環境プロパティを表現するクラス.

- epName(String)：環境プロパティ名.
- numericable(boolean)：影響の数値性．影響が数値で表現できるか否か.
- description(String)：環境プロパティに関する説明.
- nums(HashMap<String instanceName, Integer value>)：数値的な影響を足し合わせ，保持する HashMap．key は影響を発生させた機器インスタンス名，value は影響値.
- strs(HashMap<String instanceName, ArrayList<String value>>)：非数値的な影響を足し合わせ，保持する HashMap．key は影響を発生させた機器インスタンス名，value は影響値.

このサービスが公開するサービスメソッドは以下の通り.

move(String $applianceName$**, String** $methodName$**)：boolean**

move メソッドは，引数 $applianceName$，$methodName$ を受け取ると，機器 $applianceName$ のメソッド $methodName$ の実行による環境への影響を，環境インパクトモデル $DFSM[\ ]$ を用いて算出する．算出された環境インパクトは，可算的であればそれまでのインパクトと足し合わされ，不可算的であればそれまでのインパクトとは別に，環境プロパティモデル $EProperty[\ ]$ に保存される.

getEProperties()：EProperty[　eProperty]

引数はなし．戻り値は，現在の環境作用値を示す $EProperty$ の集合で

ある.

4.5　環境競合検出サービス

環境競合検出サービスは，現在の環境作用に基づいてサービス環境要求の評価を行い，環境競合を検出するサービスである.

このサービスが公開するサービスメソッドは以下の通り.

detect(String *eRequirement*)：**boolean**

detect メソッドは，サービス環境要求 *eRequirement* を受け取ると，状態遷移機械サービスより，*eRequirement* に関連した環境プロパティの環境作用値を取得する．*eRequirement* と環境作用値を比較し，*eRequirement* が満たされ，環境競合が存在しなければ true, *eRequirement* が満たされず，環境競合が検出された場合は false を戻り値として返す.

4.6　実装

サービス競合検出システムに含まれる全てのサービスは以下の環境で開発・公開した.

- サーバー:950MB RAM 2.00GHz WinXP Pro
- Tomcat 5.5
- Apache Axis2
- Java JDK5

システムの総コード行数は約 6300 行であり，加えて，HNS の構成モデルや機器モデル，環境モデルが約 1500 行であった.

第 5 章

評価

5.1 環境競合検出試験

CS27-HNS が提供する 5 種類の連携サービスに対して，開発した環境競合検出システムを用いて，1 対 1 の環境競合検出試験を行った．検出を行ったサービスは，以下の通りである．

TVTheater サービス: サービスを実行すると，TV が付き，LivingLight が暗く調光され，カーテンが閉まり，映画館のような雰囲気で TV が視聴できる．

ComingHome サービス: ユーザの帰宅時を想定し，HallLight と LivingLight を明るく点灯させ，アロマポットでリラックスする香りを出す．

BGM サービス: MusicPlayer で生活空間に BGM を流す．

AirCleaning サービス: AirCleaner で空気を清浄，脱臭する．

AutoLight サービス: 室内の人センサをトリガとし，LivingLight を明るく点灯させる．

各サービスが持つ機器メソッド，およびサービス環境要求を図 3.3 に示す．

表 5.1 に，検出試験の結果を示す．検出試験は，各サービスの組み合わせにおいて，横の行のサービスを実行したのち縦の列のサービスを実行し，実行時に発生した機器競合，環境競合と，2 サービスの競合を検査するのに掛かった実行時間をみた．表 5.1 では，機器競合 (AFI) が発生した場合は衝突が発生した機器名を，環境競合 (EFI) が発生した場合は衝突が発生した環境プロパティと，満たされなかったサービス環境要求の ID(図 3.3 と対応) を記す．

Table 5.1. FI Detection Examination

	TV_Theater	ComingHome	BGM	AirCleaning	AutoLight
TVTheater (priority:2)		AFI (LivingLight) EFI (Env.brightness, 1.2.2)	EFI (Env.sound_volume, 1.2.1) EFI (Env.sound_content, 1.2.3)		AFI (LivingLight)
		343[ms]	336[ms]	360[ms]	343[ms]
ComingHome (priority:4)	AFI (LivingLight) EFI (Env.brightness, 1.2.2)			EFI (Env.fragrance, 4.2.1) EFI (Env.fragrance, 2.2.1)	
	352[ms]		328[ms]	343[ms]	344[ms]
BGM (priority:10)	EFI (Env.sound_volume, 1.2.1) EFI (Env.sound_content, 1.2.3)				
	336[ms]	328[ms]		336[ms]	320[ms]
AirCleaning (priority:6)		EFI (Env.fragrance, 2.2.1) EFI (Env.fragrance, 4.2.1)			
	336[ms]	328[ms]	336[ms]		320[ms]
AutoLight (priority:8)	AFI (LivingLight)				
	344[ms]	328[ms]	320[ms]	328[ms]	

下線が引かれた環境競合は，先行研究における定義では検出できなかった環境競合である．また，表中に示した優先度 (priority) は，数字の低いサービスが，より優先されて実行される．

以下，特徴的な検出例について説明する．

検出例 1: TVTheater vs ComingHome　TVTheater サービスを実行中に，ComingHome サービスが実行された例である．ComingHome サービスを実行すると，まず現在実行中である TVTheater サービスとの機器競合が検査され，TVTheater サービスのメソッド「LivingLight.setBrightness(1)」と，ComingHome サービスのメソッド「LivingLight.setBrightness(10)」間で機器競合が検出される．サービス優先度に従って解消が行われ，優先度の低い ComingHome サービスは機能が縮退した形で，「LivingLight.setBrightness(10)」を除いた機器操作のみが実行される．このとき実行される「HallLight.setBrightness(10)」により，部屋内の明るさが増し，TVTheater サービスの「部屋が暗くあってほしい」という環境要求 “$XEE(TVT, Env.brightness) < 200''$ が満たされなくなるため，環境競合が検出される．

先行研究においては，サービスの要求を考慮することなく，過剰な環境競合の検出が行われていたが，サービス環境要求を導入することで，各サービスの要求に沿った適切なレベルで，環境競合の検出が行えるようになった．

検出例 2: TVTheater vs AutoLight　TVTheater サービスを実行中に，AutoLight サービスが実行された例である．検出例 1 と同様に，TVTheater サービスの「LivingLight.setBrightness(1)」と，AutoLight サービスの「LivingLight.setBrightness(10)」間で機器競合が検出され，優先度の低い AutoLight サービスは実行されない．

仮に機器競合の検出，解消を行わずに両サービスを実行した場合，AutoLight サービスは優先度を無視して実行され，部屋が明るくなることで TVTheater サービスの環境要求 “$XEE(TVT, Env.brightness) < 200''$ が満たされなくなり環境競合が誤検出されてしまう．環境競合の検出には，事前の機器競合の検出，解消が必須であることがわかる．

検出例 3: ComingHome vs AirCleaner　ComingHome サービスを実行中に，AirCleaner サービスが実行された例である．ComingHome サービスの実行によって，状態遷移機械サービスが保持する環境プロパティ $fragrance$ に，匂いコンテンツ $relax$ が加算され，$Env.fragrance$ の環境作用は $\{relax\}$ となっている．このとき，AirCleaner サービスが実行されると，AirCleaner.on() の不可算的な環境インパクト “Env.fragrance == {}” により，$Env.fragrance$ の環境作用は，$\{relax\} or \{\}$ となる．環境競合の検出フェイズにおいて，ComingHome サービスの環境要求 “EE(Env.fragrance) contains {relax}” と，AirCleaner サービスの環境要求 “EE(Evn.fragrance) is {}” が満たされないため，環境競合が検出される．

提案法を利用することで，以前は検出できなかった，ラベルや文字列等を値にとる非数値的な環境プロパティや，向きを持たない不可算的な環境インパクトに関する環境競合を検出できることがわかる．

いずれの試行でも，実行時間は 360〜320[ms] であり，実用に適することが確認された．また，提案法を利用することで，先行研究において検出できな

かった環境競合を検出し，ユーザやサービスの意に沿わない環境競合の過剰検出を防止できることが確認された．

5.2　限界

提案法は，サービスの開発者，もしくは利用するユーザが，サービス実行中に成立しておいて欲しい環境的条件，サービス環境要求を決定する必要がある．しかし，機器が環境に与える影響は多様であり，サービスを利用するユーザ一人一人にとって，許容，我慢できる環境的閾値は異なる．このため，事前にサービス環境要求を完全な形で決定するのは困難であると考えられる．条件付けを行う環境プロパティの指針を用意するとともに，サービスを利用する特定のユーザに対応したサービス環境要求を提供する枠組みがあれば，より高付加価値な連携サービスを提供することができると考える．

また，検出した環境競合は，何らかの形で解消する必要がある．先行研究においては，環境競合は機器競合と同列に扱われ，同じステップで解消が行われていた．しかし，機器競合や環境競合が検出された場合，どちらの機器操作を優先するかを必ず選択しならない機器競合とは異なり，環境競合は，解消案の一つとして両操作をそのまま実行することができる (電力容量制限など一部の環境競合を除く)．

そのため，本提案では，機器競合と環境競合の検出，解消のステップを分割し，将来的な環境競合の解消法が取り得る自由度を増した．環境競合の解消法について，本提案では述べていないが，サービス優先度やユーザ優先度による自動解消や，ユーザとの対話型応答による解消，また，競合の種別によっては何もしない等の解消策が考えられる．適切な競合解消法の選択については，今後の課題とし，検討を行う．

第 6 章

おわりに

本稿では，機器による環境への影響を詳細に記述できる環境インパクトモデルを用いて環境競合の再定式化を行った．また，ケーススタデイおよびサービス競合検出システムにより，提案の有効性を確認した．提案法は，先行研究においてみられた環境競合の過検出を防ぎ，既存の環境プロパティや，コンテンツに関する環境競合の記述能力を大きく向上させるものとなった．

現在は一部屋内の環境のみをモデル化しているが，今後は家全体の環境モデルを構築し，複数の場にまたがった環境競合や，外部の影響を考慮した競合検出，および検出した環境競合の解消法を課題としている．

謝辞

本研究を進めるにあたり，多くの方々に御指導，御協力をいただきました．ここに感謝の意を表したいと思います．

本論文をまとめるに際して有益な御助言を賜りました上原邦昭教授ならびに羅志偉教授に厚く御礼申し上げます．

神戸大学大学院工学研究科 中村匡秀 准教授には，本研究の指導教員を担当していただき，研究内容のみならず，研究室内での活動全般において多大なるご指導を頂きました．心より深く感謝申し上げます．

神戸大学大学院工学研究科 まつ本真佑 助教には，研究の進め方や，論文の書き方など詳細にご指導いただきました．厚く御礼申し上げます．

最後に，CS27 講座の皆様には，日頃の研究生活においてさまざまな御助言を頂き，心地よい研究環境を与えて頂きました．心より深く感謝申し上げます．

参考文献

[1] M. Calder, E. Magill, M. Kolberg, and S. Reiff-Marganiec, “Feature Interaction: A Critical Review and Considered Forecast”, *Computer Networks*, Volume 41/1, pp.115-141, North-Holland, January 2003.

[2] M. Kolberg, E. H. Magill, and M. Wilson, “Compatibility Issues Between Services Supporting Networked Appliances”, *IEEE Communications Magazine*, vol. 41, no. 11, Nov 2003 pp. 136-147

[3] M. Nakamura, H. Igaki, K. Matsumoto. “Feature Interactions in Integrated Services of Networked Home Appliances -An Object-Oriented Approach-,” *Int'l. Conf. on Feature Interactions in Telecommunication Networks and Distributed Systems (ICFI'05)*, pp.236-251, July 2005.

[4] M. Wilson, M. Kolberg, and E. H. Magill, “ Considering side effects in service interactions in home automation - an online approach, ” *Feature Interactions in Software and Communication Systems IX (L. du Bousquet and J.-L. Richier, eds.)*, pp. 172-187, IOS Press, Amsterdam, 2007.

[5] M. Nakamura, A. Tanaka, H. Igaki, H. Tamada, and K. ichi Matsumoto. “Constructing home network systems and integrated services using legacy home appliances and web services.” *International Journal of Web Services Research*, 5(1):82–98, January 2008.

[6] 松尾尚文, パッタラ リーラープルット, 土屋達弘, 菊野亨, “ホームネットワークシステムにおける競合問題の検証,” 情報処理学会論文誌，Vol.49, No.6, pp.2129–2143, June 2008.

[7] 吉村悠平，井垣宏，中村匡秀，” ホームネットワークシステムにおける

サービス競合の動的検出・解消システムの設計と実装 ”，電子情報通信学会技術研究報告，Vol.108, No.136, pp.35-40，July 2008.

[8] 吉村 悠平, 稲田 卓也, 井垣 宏, 中村 匡秀, “SOA にもとづくホームネットワーク サービス競合検出・解消基盤の提案,” 情報処理学会研究報告, vol.2009-SE-166, No.4, November 2009.

[9] M. Nakamura, H. Igaki, Y. Yoshimura, and K. Ikegami. Considering Online Feature Interaction Detection and Resolution for Integrated Services in Home Network System. *10th International Conference on Feature Interactions in Telecommunications and Software Systems (ICFI2009)*, 191–206, June 2009.

付録 A

関連発表論文

- 池上弘祐, 吉村悠平, 井垣宏, 中村匡秀, “サービス期間を考慮したホームネットワークサービス競合検出・解消システムの実装,” 電子情報通信学会 OIS 研究会, Vol.IEICE-108, No.IEICE-OIS-462, pp.007-012, March 2009.
- 池上 弘祐, 井垣 宏, 中村 匡秀, “ホームネットワークシステムにおけるサービス要求を考慮した環境競合検出法,” 電子情報通信学会技術報告, Vol.109, No.327, pp.053-058, December 2009.
- Kousuke Ikegami, Shinsuke Matsumoto, and Masahide Nakamura, “New Definition of Environment Feature Interactions in Home Network System,” In Workshop on Dependability of Network Software Applications 2010 (DNSA 2010), November 2010.
- 池上 弘祐, 稲田 卓也, まつ本　真佑, 中村 匡秀, “ホームネットワークシステムにおける環境インパクトの性質を考慮した環境競合の再定式化,” 電子情報通信学会技術研究報告, Vol.111, No.255, pp.67-72, October 2011.

付録 B

サービス競合検出システム実行結果

(TVTheater vs BGM):

//////////<begin:TVTheaterService/BeginEnd>////////

TVTheater **サービスの実行を試みます**

【機器競合検出解消フェイズ】

機器競合は検出されませんでした

【サービス実行フェイズ】

★ WEB **サービス実行★** TVTheater,TV,on(*)

★ WEB **サービス実行★** TVTheater,LivingLight,setBrightness(1)

★ WEB **サービス実行★** TVTheater,Curtain,close(*)

【環境競合検出フェイズ】

環境競合は検出されませんでした

TVTheater **サービスの実行処理が終了しました**

//////////<begin:BGMService/BeginEnd>////////

BGM **サービスの実行を試みます**

【機器競合検出解消フェイズ】
機器競合は検出されませんでした

【サービス実行フェイズ】
★ WEB サービス実行★ BGM,MusicPlayer,play(*)

【環境競合検出フェイズ】
[▲環境競合検出]XEE(sound_volume)<200 が満たされていません.
[▲環境競合検出]EE(sound_content)is{TV}が満たされていません.

BGM サービスの実行処理が終了しました