社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

センサ駆動連携サービスのための サービス競合検出手法に関する検討

稲田　卓也† 　池上　弘祐† 　まつ本真佑† 　中村　匡秀† 　井垣　宏††

† 神戸大学　〒 657–8501 神戸市灘区六甲台町 1–1
†† 東京工科大学　〒 192–0982 東京都八王子市片倉町 1404-1

あらまし　ホームネットワークシステム (HNS) のアプリケーションの一つとして，複数の家電を連携制御する家電連携サービスの研究が進んでいる．我々はこれまで，単体では正常に動作するが複数の連携サービスを同時に実行すると発生する連携サービス間の衝突を**連携サービスにおけるサービス競合**と定義し，検出・解消を行う手法についての研究を進めてきた．本稿では，**センサ駆動連携サービス** (センササービス) を対象としたサービス競合についての検討を行う．センササービスは連携サービスの中でも特にセンサによるイベント検知を契機として動作するサービスのことを指している．このような連携サービスでは，別の連携サービスの実行結果が与えた環境の変動をイベントとして検知してしまう**連携サービスの連鎖**による不具合が発生することがある．我々はセンササービスにおけるサービス競合を検討するにあたり，この連携サービスの連鎖を検出するための HNS モデルおよび検出手法の提案を行う．
キーワード　HNS, 家電連携サービス, センサ駆動, サービス競合.

A Study for Detecting Feature Interactions between Sensor Driven Integrated Services

Takuya INADA†, Kosuke IKEGAMI†, Shinsuke MATSUMOTO†, Masahide NAKAMURA†, and Hiroshi IGAKI††

† Kobe University　Rokkoudaityou 1–1, Nada-ku, Kobe, Hyogo, 657–8501 Japan
†† Tokyo University of Technology

Abstract　As one of the major Home Network System, the integrated service of networked home appliances are widely developed. When such integrated services are executed simultaneously, functional conflicts may occur between the functions of each appliance. Such conflicts are called Feature Interaction between HNS integrated services. In our previous research, we proposed a feature interaction detection and resolution mechanism. Sensor-driven services included in the HNS integrated services are triggered by some events detected by sensors. Therefore some sensor-driven services might be triggered by other service execution. We define such ripple effect between services as a chain of services. In this paper, we formalized the chain of services with using our extended HNS models and sensor-driven service scenario models.
Key words　HNS, Integration Services, Sensor-Driven Service，Feature Interaction.

1. はじめに

家電機器を家庭内のネットワーク (ホームネットワーク) に接続し，宅外からの制御や複数機器間の連携を実現する**ホームネットワークシステム (HNS)** の普及・導入が進んでいる [1] [2]. HNS では，テレビや DVD レコーダ，エアコン，照明，扇風機といった複数の家電や人感センサ，照度センサといったセンサを連携制御することで，家電単体で利用する場合に比べてより付加価値の高い**家電機器連携サービス**（以下，連携サービス）を実現することができる．また，連携サービスの中でも特にセンサによるイベント検知を契機として動作する**センサ駆動連携サービス** (センササービス) は，照明機器と人感センサ，自動ドアやセキュリティアラーム等実際に数多くの製品が開発・販売されている [3] [4] [5] .

以下に連携サービス例を紹介する．

SS1：DVD シアターサービス　：DVD レコーダ，テレビ，

スピーカー，照明を連携し，映画館の雰囲気でユーザが DVD を視聴できるサービス．ユーザがサービス開始ボタンを押下すると，自動的に DVD レコーダとテレビが再生モードで ON になり，照明が暗くなる．

SS2：照明制御サービス　：ユーザ在室時に，明るさを保持するサービス．ユーザ在室中の部屋で，明るさが 100lx よりも暗くなったときに，照明を点ける．

これらの連携サービスは，HNS に適用されることでユーザの利便性を向上させることができる反面，**サービス競合**と呼ばれる新たな問題を起こす可能性がある [6] [7]．サービス競合とは，単独で正常に動作する連携サービスが複数同時に実行されることで，互いに干渉・衝突を起こし，ユーザの意図した通りに動作しなくなる現象である．HNS におけるサービス競合はユーザの快適性・利便性を損ない，HNS の品質を低下させる要因となるため，HNS による**検出・解消**が求められている．

我々は先行研究において，センササービスを含まない連携サービス間に発生する競合検出・解消の仕組みを提案してきた [14]．本稿では先行研究の内容に加えて，センササービスを含んだより多様な連携サービスを含む HNS におけるサービス競合問題の検出手法についての検討を行う．

センササービスを含む連携サービスではこれまでのサービス競合問題に加え，ある連携サービスの実行結果が他の連携サービスの開始/終了に影響を与えることによる不具合を考慮する必要がある．これは連携サービスの実行により，室内の環境が変化し，別サービスに含まれるセンサがイベントを検知してしまうことによる問題である．例えば上記の連携サービス SS1 をユーザが実行することで部屋の明るさは暗くなる．SS2 は照度センサによって部屋の暗さをイベントとして検知するため，SS1 が実行されることで連鎖的に SS2 が実行されて部屋が明るくなってしまうことがある．

我々はこのような連携サービスの実行結果が他のサービスのイベントに波及することを**サービスの連鎖**と呼び，センササービスを含む連携サービス間競合問題の主たる要因であると捉える．本稿ではこの連鎖を検出する手法と検出に必要な HNS および連携サービスのモデル化について詳述する．

2. 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，宅内のネットワークに接続された複数のネットワーク家電から構成されている．各ネットワーク家電は，ユーザや外部エージェントがネットワーク越しに制御できるように，制御 API を備えている．これらのネットワーク家電 API を組み合わせることで，宅外からの制御や連携サービスといった各種アプリケーションが実際に開発・提案されている．

我々の研究室では，実際の HNS(CS27-HNS と呼ぶ) [8] を構築しており，エアコン，照明，TV といった多様な家電の機能が Web サービス [15] として公開されている．Web サービスとして構築された各機器の API は SOAP または REST 形式で呼び出すことが可能であり，複雑な連携サービスでも容易に開発することが可能となっている．

2.2 先行研究：連携サービス間サービス競合

我々は先行研究において，HNS における連携サービスのためのサービス競合を検出・解消する手法 [6], [11] を提案した．これらの手法では，各家電を状態 (**機器プロパティ**) と操作 (**機器メソッド**) を持つオブジェクトとしてみなすことで **HNS のモデル化**を行った．ここで機器メソッドは，機器プロパティを参照・更新するものとしてモデル化されている．各メソッドは以下の 2 種類の条件 (**事前条件**，**事後条件**) をもつ．

- メソッドの実行に必要な機器に関する事前条件
- メソッド実行後に成立する機器に関する事後条件

各条件はプロパティを含む条件式として構成される．例えば TV の場合，機器プロパティとして power， channel， volume を持ち，power は true(電源 ON) か false(電源 OFF) である boolean 型，channel と volume は int 型で与えられる．表 1 に機器プロパティ例を示す．ここで，TV の機器メソッドとして，音量を調節する vol(int volume) があるとする．このメソッドは事前条件として power プロパティが true でなければならず，実行後 (事後条件) に volume プロパティの値が vol メソッドの引数で与えられた値で更新されていなければならない．

連携サービスは任意の機器メソッドを組み合わせたものであり，機器メソッドの系列を**連携サービスシナリオ**と呼ぶ．例えば 1. 章で述べた SS1:DVD シアターサービスは DVDRecorder.on(), TV.on() 等のメソッドから構成されている．また，連携サービスは通常実行時に呼び出される**開始シナリオ**と終了時に実行される**終了シナリオ**を含んでいる．

我々の既存の競合検出手法では，サービス競合を「あるサービス S に含まれるメソッド m と別のサービス S' に含まれるメソッド m' について，m の事後条件と m' の事前条件あるいは m' の事後条件を同時に満たすことができない場合に発生する」と定義している．

これらのモデルおよび定義を利用することで，我々は先行研究 [13] において，HNS および競合検出のモデルを利用し，競合検出・解消を行う手法およびシステムを提案した．

2.3 センサ駆動連携サービス

センサによって人の出入りや室温などの環境の変化をイベントとして検知し，イベントに応じた振る舞いを提供するサービスをここでは**センサ駆動連携サービス** (センササービス) と呼ぶ．例えば，「室温が 28 ℃以上になる」というイベントを温度

表 1　Device Property

ApplianceName	AppliancePropertyName	PropertyType
TV	power	boolean
	channel	int
	volume	int
Light	power	boolean
	brightness	int
AirConditioner	power	boolean
	temperature	int
	mode	enum

センサが検知し,「エアコンの冷房運転を開始する」といったような機器連携が行われる．

以下にセンササービスの例にあげる．

SS3：おかえりサービス　：ユーザが帰宅した際に，照明を明るくするサービス．ユーザの帰宅を感知すると，照明を明るく制御する．

SS4：冷房サービス　：エアコンを使用し，室内の温度を快適に保つサービス．室内の温度が 24 ℃以上で且つ在人数が 1 人以上の場合，冷房を使用する．

SS5：暖房サービス　：ヒータを使用し，室内の温度を快適に保つサービス．室内の温度が 10 ℃以下で且つ在人数が 1 人以上の場合，暖房を使用する．

SS6：湿度調節サービス　：加湿器を使用し，室内の湿度を快適に保つサービス．室内の湿度が 20%以下になると，加湿器を ON にする．

通常，センサを利用してイベントを検知し，実行されるサービスは ECA ルール [9] と呼ばれるモデルにもとづいて定義される．ECA はそれぞれ Event，Condition，Action を指している．Event はサービスを実行する契機となるトリガやきっかけを表す．Condition はイベントが発生した際に評価される論理式であり，これが True であるときにのみサービスの振る舞いである Action が実行される．

先行研究では，センササービスを除いた連携サービスのみをサービス競合の対象としていた．すなわち，ECA ルールでいうところの Action に相当するもののみを連携サービスシナリオとして扱っていた．そのため，イベントやコンディションに起因する連携サービス実行時の不具合が考慮されていなかった．以降では，センササービスを対象として発生する新たなサービス競合問題とその検出手法に関する検討内容について述べる．

3. センサ駆動連携サービスにおけるサービス競合

センササービスでは，室温や湿度，明るさの変動といった環境の変化をセンサによってイベントとして検知し，対応する機器制御などを行う．そのため，あるサービスの機器制御を実行した結果が他のサービスのイベントを誘発することがある．本稿ではこのような連携サービスにおけるイベントの誘発を**サービスの連鎖**と呼ぶ．連鎖によってユーザの意図しないサービスが実行されてしまうことがある．前節であげた連携サービスを例に，連鎖が発生する 2 つのシナリオを紹介する．

シナリオ S1： ユーザが SS1:DVD シアターサービスの開始ボタンを押下し，DVD シアターサービスが実行されたとする．このとき照明が暗くなることで部屋の照度が低くなる．そのため，サービス実行後の環境状態が，SS2:照明制御サービスの開始条件である 照度 $<$ 100 lx という条件を満たしてしまう可能性がある．すなわち，SS1 の実行結果が SS2 の実行を誘発する連鎖が発生してしまう．

シナリオ S2： 室内の温度が 10 ℃以下になり，SS5:暖房サービスが実行されたとする．このとき，サービスによって暖房が使用され，室内の温度は上昇する．その結果，室内の湿度が下がり，SS6:湿度調節サービスの開始条件である 湿度 $<$ 20 %という条件を満たしてしまう可能性がある．この場合も，SS5 の実行結果と SS6 が連鎖する可能性があるといえる．

シナリオ S1 の場合，DVD シアターサービスの部屋を暗くし「映画館の雰囲気でユーザが DVD を視聴する」という要求にも関わらず，照明制御サービスによって照明が明るくなってしまう．その結果，DVD シアターサービスの目的を果たせないという状況に陥ってしまう．

シナリオ S2 の場合，暖房サービスの「温度を快適に保つ」という要求と湿度調節サービスの「湿度を快適に保つ」という要求の両方が満足されている．そのため，シナリオ S2 は許容される連鎖であるといえる．

これらの事例からも分かるように，センササービスは，他の連携サービス実行の影響を受けることが多い．我々は連鎖によって連携サービスの実行結果がユーザの意図しないものになってしまうことを**センササービスにおけるサービス競合**と呼ぶ．上記のシナリオでは S1 がサービス競合にあたると考えられる．他の不具合の可能性としては，2 つの連携サービス間だけでなく，複数の連携サービスが順に連鎖し，結果として連鎖系列が閉路に陥ることもありうる．

従来の連携サービス間競合検出は，単独で正常に動作する連携サービスが複数同時に実行される際に発生する不具合を識別することを目的としている．そのため，連鎖にもとづく不具合の検出は考慮していない．そこで本稿では，センササービスにおけるサービス競合検出を目的として，まず連携サービス間の連鎖を検出する手法を提案する．

4. センサ駆動連携サービスにおける連鎖検出手法

センササービスの連鎖は連携サービスの実行結果が他のセンササービスが検知するイベントに影響を与える場合に発生する．そのため連鎖を検出するためには，連携サービスの実行結果が環境に与える影響を分析し，センササービスに定義されているイベントと比較する必要がある．そこで我々はまず，2.2 節で述べた HNS モデルに，センササービスにおけるイベント記述と連携サービスに含まれる各機器メソッドが環境に与える影響の内容を記述するためのモデル定義を追加する．以降では，ECA ルールを考慮したセンササービスを記述するための HNS モデルについて紹介した後，そのモデルを利用して連鎖を検出する手法について述べる．

4.1 ECA ルールにもとづくセンササービス記述

本稿で我々が定義するセンササービス記述は ***S_event***, ***S_condition***, ***S_action*** の 3 つの要素を持つ．それぞれの詳細は以下に示すとおりである．

- ***S_event*** :センササービスを実行する契機となるイベントを機器や環境のプロパティから構成される条件式として記述する．ここで機器プロパティは 2.2 節で述べた各機器オブジェクトがもつ内部状態を示す．環境プロパティはセンサが取得する環境の状態を示す．ここでは，env.Brightness が室内の照度を env.Temperature が室温といったように，”env.”

ではじまる形式で記述する．例えば,「室温が 28 ℃を超える」というイベントをトリガにしてサービスが起動される場合は，$env.Temperature > 28$(℃) と記述される．

- ***S_condition*** :S_event が検知された際に，S_action の実行可否を判断するための条件式として記述する．条件式の構成要素は S_event と等しい．例えば,「室温が 28 ℃を超える」というイベントをトリガにして起動されるサービスが,「室内に人が居たら」という条件が成立するときにのみ S_action が実行されるとする．この場合は $S\_condition$ として，$env.PeopleNum > 0$ が定義される．
- ***S_action*** :従来の連携サービス記述における連携サービスシナリオ (すなわち機器メソッドの系列) を記述する．

S_action の持つ各機器メソッドは従来のメソッドモデルにおける事前条件 ***M_pre***，事後条件を ***M_post*** に加え，本稿で提案する以下の ***ME_future*** を持つ．

- ***ME_future*** :ある状態にある機器の機器メソッドを実行し続けたときに特定の環境プロパティに与える最終的な影響の大きさと方向を示す．

これらを用いた新しいモデルにもとづくセンササービスのシナリオを図 1 に示す．

2. 2 節で述べたように，全ての機器は機器状態と機器メソッドを持つ．そのため，機器メソッドによる機器状態の変化を状態遷移機械を用いたモデルとして表現することができる．ここで d を機器，M_d を d の持つ機器メソッドの集合とし，S_d を機器状態の集合，s_0 は初期状態であるものとする．機器ごとの状態遷移機械 I_d は $I_d = (S_d, M_d, g_d, s_0)$ と定義される．g_d は状態遷移関数であり，$S_d \times M_d \to S_d$ を表す．

このとき $m_d(\in M_d)$ における ME_future の集合 F_{m_d} は，全ての状態遷移の集合 T_d ($\subseteqq S_d \times M_d \times S_d$) および環境プロパティ集合 E の直積を F_{m_d} に写像する関数 Δ によって $\Delta : T_d \times E \to F_{m_d}$ 表される．

表 2〜5 に各機器の ME_future を示した．例えば，表 4 において，TV は S1:(power=on) と S2:(power=off) の 2 種類の状態を持つ．状態が S2 のとき機器メソッド on を実行すると，S1 に遷移する．このとき，ME_future は

- $\triangle(env.Brightness, (S2, on(), S1))$
  $= (env.Brightness, +200)$

と定義されており，明るさ (env.Brightness) へ 200 の正の変化を及ぼす．なお表では，$\triangle(e, t) = +v$ (または $-v$) を遷移 t のラベルとして $\triangle e = +v$(または $= -v$) と簡略表記にし，環境プロパティは”env.”以降の文字の頭文字で略記している．

また，env.Brightness 等の明るさに関する影響は状態遷移の直後にその変動が収束するが，温度，湿度などは必ずしもすぐに変動が終了するものではなく時間変化を伴う．ここで ME_future は対応する機器メソッドを実行し続けた結果の収束値であるとして定義しており，例えば表 2 に示すようなエアコンの状態遷移では温度 (env.Temperature) はエアコンの設定温度 (temperature) に収束するものとして定義されている．このように，ME_future の取る値は環境プロパティの絶対値あるいは機器メソッド実行前の環境プロパティの値と比較した相対値のどちらかで指定される．

以降では，これらのモデルを利用した連携サービス間の連鎖を検出する手法について詳しく説明する．

4. 2　連鎖検出手法

ここでは，先に実行される連携サービスを SS_{old}，また SS_{old} 実行の結果，連鎖が引き起こされる可能性のある連携サービスを SS_{new} とし，2 者間に連鎖の関係があるかどうか，またどのような機器状態および環境状態のときに連鎖が発生するかを検出する手法を示す．

検出は以下の手順で行う．

Step1 : SS_{new} からイベント SS_{new_event} を取得する．

Step2 : SS_{new_event} に含まれている全てのプロパティの集合 P を抽出する．

Step3 : SS_{old} の SS_{old_action} に含まれる全ての機器メソッドのうち，P に含まれるプロパティを一つ以上含む ME_future を持つ機器メソッド集合 $M_{P_{SS_{old}}}$ を抽出する．

Step4 : $M_{P_{SS_{old}}}$ に含まれる機器メソッド $m1$, $m2$,.., mn それぞれがもつ ME_future 集合 (F_{m1},F_{m2} ,... ,F_{mn}) からプロパティ P を含むもの $F_{P_{mi}}(1 \leqq i \leqq n)$ をメソッド毎に抽出する．

Step5 : $F_{P_{mi}}(1 \leqq i \leqq n)$ における全ての ME_future の組み合わせ集合 FC_{ME_future} を算出する．

Step6 : 全ての $fc_{ME\_future}(\in FC_{ME\_future})$ において，プロパティ $p(\in P)$ 毎の ME_future を加算した合成 ME_future, $afc_{ME\_future}(\in AFC_{ME\_future})$ を算出する (加算方法については後で詳述する)．

Step7 : 全てのプロパティ $p(\in P)$ の初期状態が SS_{new_event} の補集合に含まれているという仮定の下で，SS_{new_event} の補集合から SS_{new_event} の状態に遷移させる可能性がある全ての afc_{ME_future} を AFC_{ME_future} から導出する．ここで afc_{ME_future} が 1 つでも導出されれば，SS_{new} は SS_{old} に連鎖する．

Step8 : Step7 で導出された afc_{ME_future} をもとに，連鎖を起こす可能性がある初期の環境状態および機器状態を導出する．

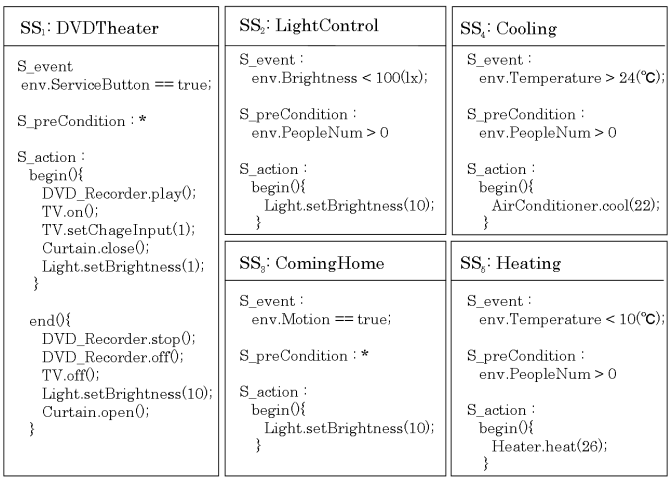

図 1　センサ駆動連携サービスシナリオ

表 2　AirConditioner の ME_future

		Method				
		cool(20)	cool(22)	cool(24)	cool(26)	off
State	S1:on (temperature = 20)	-	env.temp = 22 / S2	env.temp = 24 / S3	env.temp = 26 / S4	env.temp = * / S5
	S2:on (temperature = 22)	env.temp = 20 / S1	-	env.temp = 24 / S3	env.temp = 26 / S4	env.temp = * / S5
	S3:on (temperature = 24)	env.temp = 20 / S1	env.temp = 22 / S2	-	env.temp = 26 / S4	env.temp = * / S5
	S4:on (temperature = 26)	env.temp = 20 / S1	env.temp = 22 / S2	env.temp = 24 / S3	-	env.temp = * / S5
	S5:off	env.temp = 20 / S1	env.temp = 22 / S2	env.temp = 24 / S3	env.temp = 26 / S4	-

表 3　Heater の ME_future

		Method				
		heat(20)	heat(22)	heat(24)	heat(26)	off
State	S1:on (temperature = 20)	-	env.temp = 22 / S2	env.temp = 24 / S3	env.temp = 26 / S4	env.temp = * / S5
	S2:on (temperature = 22)	env.temp = 20 / S1	-	env.temp = 24 / S3	env.temp = 26 / S4	env.temp = * / S5
	S3:on (temperature = 24)	env.temp = 20 / S1	env.temp = 22 / S2	-	env.temp = 26 / S4	env.temp = * / S5
	S4:on (temperature = 26)	env.temp = 20 / S1	env.temp = 22 / S2	env.temp = 24 / S3	-	env.temp = * / S5
	S5:off	env.temp = 20 / S1	env.temp = 22 / S2	env.temp = 24 / S3	env.temp = 26 / S4	-

表 4　TV の ME_future

		Method	
		on	off
State	S1:on	-	Δ B = -200 / S2
	S2:off	Δ B = +200 / S1	-

表 5　Light の ME_future

		Method			
		setBrightness(1)	setBrightness(5)	setBrightness(10)	off
State	S1:on (brightness = 1)	-	Δ B = +400 / S2	Δ B = +900 / S3	Δ B = -100 / S4
	S2:on (brightness = 5)	Δ B = -400 / S1	-	Δ B = +500 / S3	Δ B = -500 / S4
	S3:on (brightness = 10)	Δ B = -900 / S1	Δ B = -500 / S2	-	Δ B = -1000 / S4
	S4:off	Δ B = +100 / S1	Δ B = +500 / S2	Δ B = +1000 / S3	-

上記の一連のプロセスでは，まず SS_{new} からイベントを取得し，Step2 においてイベントに含まれているプロパティを取得する．Step3 では，取得したプロパティに対して影響を与える可能性がある機器メソッド集合を獲得し，機器メソッドごとのプロパティに与える影響の度合い (ME_future) を Step4 で取得する．連携サービス内の一連の機器メソッドが各プロパティに総合的にどのような影響を与えるかを評価するため，Step5 において，関連するプロパティに影響を与える複数の機器メソッドが持つ ME_future の全組み合わせを抽出する．

Step6 では，Step5 で抽出した ME_future の組み合わせをもとに，一連の機器メソッドを実行したときにプロパティに与える影響の内容を算出する．ここでは，特定のプロパティに影響を与える複数の機器メソッドが存在した際に，それらの影響 (ME_future) を足しあわせることで，総合的な影響の大きさを算出する．ME_future は環境プロパティとプロパティの変動する値もしくは最終的に到達する絶対値のどちらかを持つ．そのため，単純に足し合わせるだけでは問題があることがある．そこで我々は，機器メソッドと環境プロパティの組ごとに加算ルールを設定しておき，そのルールにもとづいて ME_future の加算を行う．加算ルールは以下の 3 種類があるものとする．

A1:　単純な足し合わせを行う加算ルール．例：照明機器の明るさ調節機能による照度プロパティに対する影響．

A2:　最も大きい値を選択する．例：複数の空調機器があるようなケースで，暖房機能による室温プロパティに対する影響．表 2 に示すように，空調機器の温度調節機能は実行し続けることで設定された室温に近づけることを目的としている．そのため複数の空調機器がある場合の ME_future は最も影響度合いの大きいものになる．

A3:　最も小さい値を選択する．例：A2 と同様のケースで，冷房機能による室温プロパティに対する影響．

ここで，複数の ME_future に異なる加算ルールを同時に適用するケースは想定しない．

以上のような連鎖検出プロセスによって，連携サービス間の連鎖と連鎖が発生する状況を検出することが可能となった．以降では，具体的な連携サービスを対象に連鎖を検出する事例を紹介し，考察を行う．

5. ケーススタディ

図 1 で示した 6 種類のセンササービスに対し，4. で示した手法を用いて，実際に連鎖検出を行った．

連鎖 (DVD シアター&照明制御)

この事例では，SS_{old} は SS1:DVD シアターサービスで，SS_{new} は SS2:照明制御サービスを対象として連鎖を検出する．

Step1 で SS2 から，$env.Brightness < 100$ がイベントとして抽出される．Step2 で env.Brightness というプロパティが求められる．Step3 で env.Brightness に対して影響を与えるメソッドの抽出を SS1 を対象に行う．この事例では，TV.on(), Light.setBrightness(1) という 2 つのメソッドが抽出される．Step4 でメソッド毎に env.Brightness を含む ME_future を抽出する．TV.on()(= $m2$) に関して，

- $f1_{P_{m2}} = \triangle(env.Brightness, (S1, on(), S1)) = (-, -)$
- $f2_{P_{m2}} = \triangle(env.Brightness, (S2, on(), S1)) = (env.Brightness, +200)$

Light.setBrightness(1)(= $m5$) に関して，

- $f1_{P_{m5}} = \triangle(env.Brightness, (S1, on(), S1)) = (-, -)$
- $f2_{P_{m5}} = \triangle(env.Brightness, (S2, on(), S1)) = (env.Brightness, -400)$
- $f3_{P_{m5}} = \triangle(env.Brightness, (S3, on(), S1)) = (env.Brightness, -900)$
- $f4_{P_{m5}} = \triangle(env.Brightness, (S4, on(), S1)) = (env.Brightness, +100)$

の計 6 つの ME_future が導かれる．Step5 で ME_future の組み合わせを求める．この事例では，8 組の組み合わせが作成される．ここでは例として以下の 2 つを対象として説明を行う．

- $fc_1 = f1_{P_{m2}} + f3_{P_{m5}}$
- $fc_2 = f2_{P_{m2}} + f4_{P_{m5}}$

これら 2 つの ME_future は加算ルールが A1 の単純加算になるため，以下のように合成される．

- $afc_1 : \triangle B = -900$
- $afc_2 : \triangle B = +300$

Step7 で照明制御サービスにおける $SS2\_event$ の補集合は $env.Brightness \geqq 100$ になるので， $env.Brightness \geqq 100$ の状態から，$env.Brightness < 100$ に遷移せしめる可能性のある afc を抽出すると，afc_1 をはじめとした 4 つの afc が算出される．最後に Step8 において，afc_1 は $100(lx) \leqq env.Brightnss < 800$ の環境下で，TV が S1(power==on)，Light が S3(brightness==10) の機器状態のとき SS1 を実行すれば，SS2 が連鎖することがわかる．

連鎖 (暖房&冷房)

SS4:暖房サービスと SS5:冷房サービスの間の連鎖を検出する例を紹介する．Step1,2,3 で SS5 から $env.Temperature > 24$ がイベントとして抽出され，env.Temperature に影響を及ぼす Heater.heat(26) が SS4 から抽出される．Heater.heat(26) において，ME_future は全て同じであるので，

- $F_{P_{m1}} = \triangle(env.Temperature, (*, heat(26), S4)) = (env.Temperature, 26)$

が抽出される．Step6 における環境への影響度の合成については，A2(最大値) にもとづいて行われ，

- $afc_1 : env.Temperature = 26$

が算出される．Step7 で冷房サービスの補集合 $env.Temperature \leqq 24(℃)$ から $env.Temperature > 24$ へと afc_1 が遷移せしめることが可能であるため，連鎖が発生すると判断できる．さらに，Step8 で $env.Temperature > 24(℃)$ の環境下で暖房サービスを実行すると，冷房サービスに連鎖する可能性があると検出できた．

これらの事例からも分かるように，本稿で我々が提案した手法によって，連携サービス間の連鎖がどのような状況でどのサービス間に発生するのかを具体的に検出することが可能となった．全ての連携サービス間の連鎖とその連鎖が発生する状況を我々の手法によって検出することにより，2 つ以上の連携サービス間におけるループなどの連鎖に起因する不具合を容易に発見することが可能となると考えている．今後の課題としては，カーテンの開け閉めによる明るさの変動のような，そのときの環境状態によって ME_future の値が変わるような機器メソッドをどのように扱うか，また連鎖発生時の不具合分類によるセンササービスにおけるサービス競合の定義を行うことを考えている．

6. おわりに

本稿では，センササービスを含む連携サービス間に発生するサービスの連鎖検出手法とそのためのセンササービスモデルおよび HNS モデルを提案した．先行研究では考慮していなかったセンサのイベント記述を追加し，連携サービスに含まれている一連の機器メソッド実行が環境に与える影響を ME_future として定義することで，連携サービス間の連鎖を検出することが可能となることを示した．

今後は，検出された連鎖の情報をもとに，連鎖に起因する連携サービス間競合の定式化と検出・解消モデルの構築を進めていくことを考えている．

謝辞 この研究の一部は，科学技術研究費 (若手研究 B 20700027,21700077)，およびパナソニック電工株式会社の助成を受けて行われている．

文　献

[1] パナソニック電工株式会社,“ ライフィニティ ”, http://denko.panasonic.biz/Ebox/kahs/
[2] 東芝,“ 東芝ネットワーク家電 Feminity ”, http://www3.toshiba.co.jp/feminity/about/index.html
[3] Matsushita Electric Works Ltd. Katteni switch, http://biz.national.jp/Ebox/katte_sw/.
[4] Nabtesco Corp. Automatic entrance system. http://nabco.nabtesco.com/english/door_index.asp.
[5] Daikin Industries Ltd. Air conditioner, http://www.daikin.com/global_ac/.
[6] M. Nakamura, H. Igaki, and K. Matsumoto, ”Feature Interactions in Integrated Services of Networked Home Appliances -An Object-Oriented Approach-,” In Proc. of Int’l. Conf. on Feature Interactions in Telecommunication Networks and Distributed Systems (ICFI’05), pp.236-251,2005.
[7] M. Wilson, M. Kolberg, and E. H. Magill. Considering side effects in service interactions in home automation - an online approach. In *Proc. Int’l. Conf. on Feature Interactions in Software and Communication Systems (ICFI’07)*, pages 172–187, 2007.
[8] 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.
[9] Chongqing Zhang, Minglu Li and Qunhua Pan, ”An ECA Rules Based Middleware Architecture for Wireless Sensor Networks ”, IEEE, ISBN:0-7695-2405-2 , PDCAT’05, Pages: 586 - 588, 05-December 08,
[10] 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.
[11] 吉村 悠平, 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家電連携サービスのための競合解消方式の考察 ”, 電子情報通信学会技術研究報告, Vol.108, No.458, pp.439-444, March 2009.
[12] 坂本 寛幸, 井垣 宏, 中村 匡秀, “コンテキストアウェアアプリケーションの開発を容易化するセンササービス基盤,” 電子情報通信学会技術研究報告, vol.108, no.458, pp.381-386, March 2009.
[13] 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.
[14] Masahide Nakamura, Hiroshi Igaki, Yuhei Yoshimura, and Kousuke Ikegami, ”Considering Online Feature Interaction Detection and Resolution for Integrated Services in Home Network System, ”In 10th International Conference on Feature Interactions in Telecommunications and Software Systems (ICFI2009), pp.191-206, June 2009.
[15] web サービス, http://www.w3.org/2002/ws;