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.

時間駆動型Webサービス呼び出しフレームワーク WS-Schedule Managerの提案と実装

大西　洋司† 　前島　弘敬† 　西澤　茂隆† 　田中秀一郎† 　中村　匡秀†
松本　健一†

† 奈良先端科学技術大学院大学 情報科学研究科　〒 630–0192 奈良県生駒市高山町 8916–5
E-mail: †{yoji-o,hirotaka-m,shigetaka-n,shuichiro-t,masa-n,matumoto}@is.naist.jp

あらまし　近年 Web サービスを何らかのイベントをきっかけとして呼び出す，イベント駆動型アーキテクチャ(EDA)の技術が研究・開発されてきている．多くの EDA のフレームワークは，あらゆる種類のイベントに対応できるよう設計されているため，その仕様が複雑・膨大になりがちである．したがって，特定のイベントのみを扱う Web サービスシステムに適用する際にも多大な労力がかかる．本稿では，数ある種類のイベントのうち「時間イベント」に特化した，Web サービス呼び出しフレームワーク WS-Schedule Manager を提案する．ユーザが呼び出す Web サービスとパラメタ諸元およびその時間スケジュールを登録すると，WS-Schedule Manager が指定した時間に Web サービスを呼び出す．Web サービスの時間駆動呼び出しという目的に特化しているため，仕様がコンパクトで動作が軽量であり，さまざまな Web サービスアプリケーションに組み入れて使用することができる．また本稿では，ホームネットワークシステムの目覚ましサービスなど，いくつかの適用事例について考察を行う．

キーワード　Web サービス，連携 Web サービス，サービス指向アーキテクチャ，イベント駆動型アーキテクチャ，時間駆動，スケジュール

Time-Driven Approach of Web Service Invocation Framework “WS-Schedule Manager”

Yoji ONISHI†, Hirotaka MAESHIMA†, Shigetaka NISHIZAWA†, Shuichiro TANAKA†, Masahide NAKAMURA†, and Ken-ichi MATSUMOTO†

† Graduate School of Information Science, Nara Institute of Science and Technology
8916–5, Takayama, Ikoma, Nara 630–0192 Japan
E-mail: †{yoji-o,hirotaka-m,shigetaka-n,shuichiro-t,masa-n,matumoto}@is.naist.jp

Abstract　Recently, some researches and developments of Event Driven Architecture (EDA) are in progress. This architecture is distinctive by invoking Web services by the specific event defined by users. The specification of many EDA frameworks can be complex and huge scale sometimes because it is designed to correspond to every kind of events. Therefore, there is large amount of cost when applying this architecture to the Web service which handles only specific events. In this paper, we propose a Web services invocation framework “WS-Schedule Manager” which is specified with “Time event.” The framework invokes a Web service at the specified time after a user set the Web service, parameters and their time schedules. The specification of this framework is compact and lightweight to execute because it is specialized to handle the time driven invocation of Web service, so this framework can be used by embedding various Web service applications. We also consider about some case studies such as wakeup service on Home Network System in this paper.

Key words　web services, composite web services, service oriented architecture, event driven architecture, time driven, schedule

1. はじめに

サービス指向アーキテクチャ(Service Oriented Architecture, SOA) [12] が注目を集めている．SOA とは，分散するソフトウェアシステムの機能群を，ビジネスとしての価値を形成しうる「サービス」という単位でネットワークに公開し，それらを組み合わせてさらに高度なサービスを構成するシステムアーキテクチャである． SOA を実現する標準技術として **Web サービス** [10] が知られている．Web の性質上，Web サービスの呼び出し方法は，基本的にクライアント側が必要なときに呼び出す**要求/応答型**に基づいている．したがって，クライアント側の要求ではなく，何らかのイベントが発生したときに Web サービスを自動的に呼び出すことは，元来前提とされていない．

近年，何らかのイベントを契機としてサービスを駆動する**イベント駆動型アーキテクチャ**(Event Driven Architecture, 以降 EDA) の研究・開発が盛んであり，SOA や Web サービスに対するフレームワークも提案されてきている．

EDA におけるイベントは，顧客の注文，天候の変化，トラブル発生，時間超過など，システムと非同期的に発生するものであり，システム設計者はイベントの種類に応じて適切なアクション（＝サービス呼び出し）を事前に定義しておく．EDA の考え方を適用することで，元来の Web サービス技術のみでは実現できなかった Web サービスの非同期呼び出しや Publish/Subscribe 通信 (あらかじめ情報を求めている購読者に対して，イベントを契機に情報を発行する仕組み) が可能となる．

現在，EDA を Web サービスに適用するための様々なフレームワークが提案されている [1], [2], [5]．しかしながら，これらのフレームワークは，あらゆる種類のイベントを汎用的に扱える反面，その仕様が複雑・膨大になりがちである．したがって，特定種類のイベントのみを扱う Web サービスシステムに適用する際に多大な労力がかかる．

本稿では，数ある種類のイベントのうち**時間イベント**に特化した，Web サービス呼び出しフレームワーク WS-Schedule Manager を提案する．あらかじめ指定した時間やスケジュールに「到達した」という状態をイベントとして捉え，Web サービスを呼び出す．時間イベントは，様々なシステムにおいても普遍的・汎用的に利用できるイベントであり，EDA において，もっとも頻繁に用いられるイベントである．WS-Schedule Manager は，時間イベントのみを対象とした EDA を実現するコンパクトで軽量なフレームワークである．利用者は様々な Web サービスアプリケーションに組み入れて，容易に時間駆動型 Web サービス呼び出しを実現できる．

ユーザが，呼び出したい Web サービスとパラメタ諸元およびその時間スケジュールを登録すると，WS-Schedule Manager は設定されたスケジュールどおりに Web サービスを呼び出す．

本稿では，WS-Schedule Manager を実装し，いくつかの時間駆動型サービスへの適用事例の考察を行った．また，Web サービス制御のホームネットワークシステム (NAIST-HNS) における目覚ましサービスを実際に実装し，提案フレームワークの有効性を確認した．

2. 準　　備

2.1 サービス指向アーキテクチャ(SOA) と Web サービス

サービス指向アーキテクチャ(SOA) [8] は，ネットワーク上に分散する複数のシステムを柔軟に連携・統合するためのシステムアーキテクチャであり，主に企業システムを対象に研究・開発が進められている．SOA では，各システムの機能をサービスという単位でくくりだし，これらの組み合わせでより高度なサービスを組み上げていく．SOA におけるサービスは，それぞれが自己完結でオープンなインターフェースを持つため，サービス同士の独立性が高い．Web サービスは，その SOA を実現するための標準的な技術である．

Web サービスが持つ機能を利用するためのインターフェースは，WSDL(Web Services Description Language) [11] で定義される．WSDL は Web サービスごとに用意され，Web サービスが持つメソッドや引数に関して記述されている．WSDL を参照することで，その Web サービスが持つ機能をすべて把握することができる．

2.2 イベント駆動型アーキテクチャ(EDA)

何らかのイベントをきっかけとして，あらかじめ定義しておいた操作を呼び出すという設計手法を，イベント駆動型アーキテクチャ(EDA) という．イベントの種類としては，システム利用者による操作や，顧客の注文，センサーの反応，特定の時間になったなど，自由に定義することができる．このようなイベントの発生を監視しておき，そのイベントが発生したときに，あらかじめ定義しておいた操作を行う．Web サービスの技術だけではこのようなイベントに対応した動作はできないが，Web サービスと EDA の考え方を併用することで，多様なサービスの実行形態をもつことができる．

EDA の考え方を用いた Web サービスの仕様の 1 つに WS-Notification [7] がある．これは，Web サービスを用いて Publish/Subscribe 通信を規定した仕様である．あらかじめ，通知の受け手である NotificationConsumer が，通知側である NotificationProducer に対して，発生したときに知らせて欲しい種類のイベント (Topic) に対して購読と呼ばれる登録作業を行っておく．NotificationProducer では，そのサーバーでイベントを監視し，あらかじめ定義しておいたイベントが発生したときに，購読している NotificationConsumer に対して情報が発行される．

3. 提案するフレームワーク

時間駆動に特化した Web サービス呼び出しフレームワークである “WS-Schedule Manager” について述べる．

3.1 要 件 定 義

Web サービスを時間駆動で呼び出すため，本稿では，以下の要求を満たすフレームワークを提案することを目的とする．なお本稿では，Web サービスを時間駆動で呼び出すためのルール定義のことをスケジュールと呼ぶ．

要求 R1　Web サービスを指定した時間に呼び出すスケジュールを登録できること．

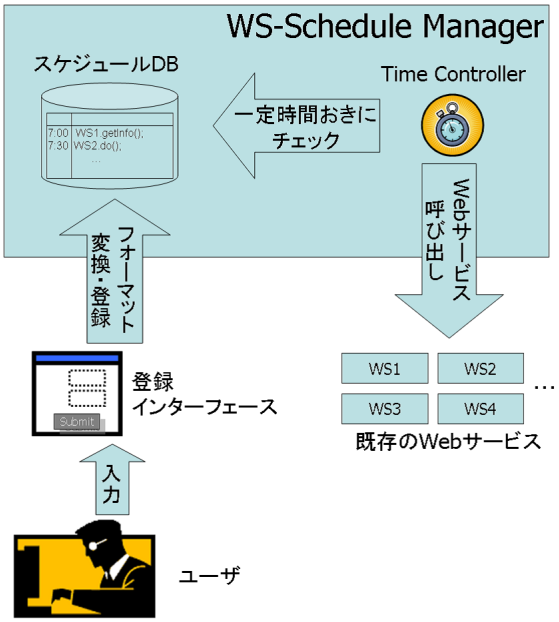

図 1　WS-Schedule Manager フレームワークの構成図

要求 R2　定期的に Web サービスを呼び出すスケジュールを登録できること．

要求 R3　Web サービスの呼び出し結果を保存し，参照できること．

時間駆動を考えたとき，登録されている日時に 1 回限りで実行する**単発スケジュール**と，毎日朝 7 時や毎週火曜日朝 9 時などのように，一定時間おきに定期的に実行する**定期スケジュール**の 2 種類のスケジュールが考えられる．これらのスケジュールを本フレームワークで取り扱うことができるようにするため，要求 R1，R2 とした．また，Web サービスを呼び出す事で戻り値を得るため，それを保存，参照できるようにする必要がある．これらの要件を満たすように，フレームワークを設計した．図 1 にその全体像を示す．

3.2　フレームワークの動作

本フレームワークでは，特定の時間に特定の Web サービスを実行したいというスケジュールを，図 1 におけるスケジュールデータベース (以下，スケジュール DB) に登録する．このスケジュール DB に新たにスケジュールを追加することで，指定時間に呼び出したい Web サービスを登録することができる．

ユーザーは，スケジュールの入力を登録インターフェースを通じて入力する．この登録インターフェースは用途に合わせてカスタマイズするため，本フレームワークでは考慮しない．スケジュール DB に登録する前に統一したフォーマットに変換し，いずれのインターフェースを用いても同じフォーマットになるよう変換を行う．

スケジュールの指定時間になったかどうかの確認は，図 1 における Time Controller が行う．Time Controller は，スケジュール DB を一定時間おきにチェックし，実行すべき Web サービスがないかを確認する．もし，実行すべきスケジュールが登録されていれば，新しいスレッドで Web サービスを呼び出す．Web サービスの実行が終われば，Web サービス呼び出しの戻り値をスケジュール DB に格納する．このことにより，要求 R3 が満たされる．

3.3　スケジュールのフォーマット

本フレームワークで扱うスケジュールは，Outlook などの市販のスケジュール管理ソフトで扱うような一般的なスケジュールと直感的に同等に扱うことができる．そのため，そのようなスケジューラでよく用いられている iCalendar(RFC2445) [4] の仕様をベースとして作成した．

スケジュール DB に格納されるスケジュールの項目は次のとおりである．

- スケジュール ID
- 登録したコンピュータの識別子
- 登録日時
- スケジュールの開始日時
- スケジュールの終了日時
- 定期イベントの頻度
- Web サービス呼び出しコマンド
- 重要度
- 実行結果
- スケジュールの有効/無効

それぞれのスケジュールには，ユニークなスケジュール ID が自動的に振られる．スケジュール登録後の修正，削除は，このスケジュール ID をもとにして行われる．

スケジュールが実行すべき時間になれば，Web サービス呼び出しコマンドの項目に入力されているプログラムが起動される．WSDL とメソッド名，パラメータを渡すことで Web サービスを呼び出すプログラムを作成したことにより，Web サービスを呼び出すことができるようになっている．

スケジュールの有効/無効の項目は，そのスケジュールが該当の時間になったときに，その Web サービスを実際に呼び出すかどうかを設定することができる．スケジュール DB に登録されているスケジュールを一時的に無効にする場合に使うことができる．

スケジュールの開始日時，終了日時，定期イベントの頻度については，単発スケジュールの場合と定期スケジュールの場合で意味が異なる．

- 単発スケジュールの場合
  スケジュールの開始日時は，そのスケジュールが実行される日時である．また，スケジュールが，1 度だけ実行されるスケジュールかどうかは，定期イベントの頻度の値を見ることで判別できる．定期イベントの頻度の項目が空欄であれば，定期的に実行しないということを表しているため，単発スケジュールであると判断できる．その際，スケジュールの終了日時の値は無視される．
- 定期スケジュールの場合
  定期的に実行されるスケジュールであるかどうかは，定期イベントの頻度の値を見ることで分かる．定期スケジュールは，スケジュールの開始日時からスケジュー

ルの終了日時の間の範囲で，定期イベントの頻度秒ごとに実行されることを表している．すなわち，定期スケジュールは，(スケジュールの開始日時 + 定期イベントの頻度 $\times\ i$ 秒)$(i = 0, 1, 2, \ldots)$ の時刻に実行され，スケジュールの終了日時がくるまで繰り返される．

3.4 実　　装

提案したフレームワークの実装を行った．それぞれのモジュールは，次の言語・ソフトウェアを用いて構築した．

スケジュール DB　MySQL

Time Controller　Java

Web サービス呼び出しモジュール　Perl

本フレームワークでは，スケジュール DB に対して 10 秒ごとにスケジュールをチェックするようにした．この時間は，システムへの負荷や登録するスケジュールの種類に応じて変更できる．

Web サービス呼び出しモジュールは，登録したスケジュールに書かれている Web サービスを実際に呼び出すためのスクリプトである．

スケジュールの登録方法

スケジュールを登録するには，一般的に登録インターフェースを用意し，ユーザーはそのインターフェースから登録を行う．WS-Schedule Manager では，Perl を用いてスケジュール DB にスケジュールを登録するスクリプトを作成し，Web 上から登録できるようなインターフェースを作成した．Web インターフェースでは，スケジュール開始日時と定期イベントの実行頻度，呼び出したい Web サービスの呼び出しコマンド (本フレームワークでは，Web サービス呼び出しモジュールとその引数) を入力できるようにした．

4. 応 用 事 例

本章では，提案フレームワークを用いるときの事例，また，提案フレームワークならではの利用例を示す．

4.1 快適めざましサービス

4.1.1 Web サービスに基づく家電によるホームネットワークシステム [3], [6], [9]

家電機器をネットワークで接続し，ネットワーク経由でそれぞれの家電をアプリケーションにより制御することのできるネットワーク家電がある．そのようなネットワーク家電と同様に，ネットワーク家電以外のネットワーク対応していない従来の家電機器に対して，Web サービスのインターフェースを実装し，赤外線リモコンを用いてネットワーク家電と同様に操作できるようにした**ホームネットワークシステム (Home Network System，以下 NAIST-HNS)** が提案されている [3], [6], [9]．NAIST-HNS では，家電それぞれを 1 つの大きな機能を持ったサービスと捉え，それらを Web サービス化している．そのため，そのような Web サービスを呼び出すことにより操作を行うことができる．そして，それらを連携することで新たな連携サービスを作成することができる．

NAIST-HNS における家電の 1 つとして，部屋の電気がある．この電気のスイッチを ON にしたい場合，LIGHT という Web サービスの on() メソッドを呼び出すことで，電気をつけることができる．現状では，このような Web サービスを呼び出すために，テレビ画面から専用リモコンで操作する方法と，携帯の画面から操作する方法の 2 種類がある．

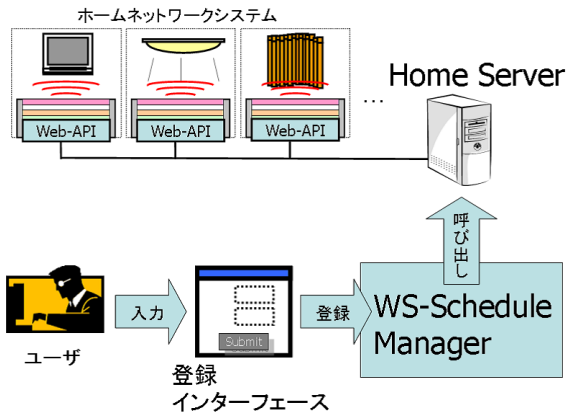

図 2　ホームネットワークシステムにおける本フレームワークの構成図

4.1.2 快適めざましサービスの概要

今回はこの NAIST-HNS において，Web サービスを用いて操作できる家電を時間駆動で動作させる．構成図を図 2 に示す．前節で述べた，部屋の電気をつける場合，部屋の電気をつける Web サービスの呼び出しを，呼び出したい時間とともに WS-Schedule Manager に登録することで，部屋の電気の動作を時間駆動で行うことができる．

また家電を個別に操作するだけではなく，例えば，朝 6 時半に電気がつき，朝 6 時 45 分にテレビの電源が入り，朝 7 時まで 5 分おきに少しずつボリュームが上がっていく，というスケジュールを登録し，これらのスケジュールを，起床したい時間に調整して登録しておく．このような，スケジュールを複数連動させることにより，快適な目覚めを支援する連携サービスを提供できる．

4.1.3 登録インターフェース

NAIST-HNS のような一般家庭において本フレームワークを用いる場合，スケジュールを詳細に入力できるようにすることも重要だが，それ以上に入力のしやすさを考慮しなければならない．そのため，そのような環境で用いるための簡易なインターフェースを作成した (図 3)．このインターフェースでは，Web サービスを実行する時間と，実行する曜日，呼び出す複数の Web サービス (ここでは家電の種類) をチェックボックスを用いて入力できるようにした．定期イベントの頻度は，入力された時間と曜日の情報，現在時刻を用いて自動的に計算する．このことで，入力が煩わしい携帯インターフェースや家電リモコンなどからでも容易にスケジュールを入力できるようになった．

また，登録されたスケジュール一覧も，携帯インターフェースから確認できるようにした (図 4)．この画面から，次のようなことが分かる．

図 3　ホームネットワークシステムにおけるスケジュール登録インターフェース

図 4　携帯インターフェースにおけるスケジュール一覧表示

1. 月曜日の朝 6 時にテレビの電源を ON にする Web サービスを呼び出す
2. 火曜日の朝 7 時に照明を，朝 7 時半にカーテンを開けるという Web サービスを呼び出す
3. 金曜日の朝 9 時半に照明を ON にする Web サービスを呼び出す
4. 1. のスケジュールは，頭に×がついていることから，一時的に無効に設定されている

4.1.4　本アーキテクチャを用いる利点

時間駆動である本フレームワークを用いることで，従来の Web サービスだけでは実現できなかった，このような細かな時間単位で複数の Web サービス実行を行うことができる．それにより，快適に起床するための新しい家電連携サービスを提供することができるようになった．また，タイマー機能やスケジュール機能のない家電に対して，それらの家電を操作するスケジュールを登録することで，それらの機能を付加することもできる．さらに，登録したそれぞれのスケジュールは他のスケジュールと独立しているため，例えば朝 6 時半に電気をつけるという Web サービス呼び出しが失敗した場合でも，次に呼び出されるテレビの電源を入れる Web サービス呼び出しには影響が出ないという特徴がある．

4.2　Yahoo!オークション 出品状況確認サービス

ポータルサイトの Yahoo!では，そのサイト内で行っている Yahoo!オークションの情報を Web サービスで提供している [13]．このサービスを WS-Schedule Manager とともに用いることで，オークションの情報を定期的に得ることができる．

まず，市販品ではなくオークションで買いたい商品ががあったとする．しかし，タイミングよくいつも欲しい商品がオークションで出品されているとは限らない．そこで，Yahoo!オークションにその商品が出品されていないかを一日おきに検索して探すための Web サービス呼び出しを WS-Schedule Manager に登録しておく．WS-Schedule Manager は，登録された時間おきにその Web サービスを呼び出すので，もし欲しい商品がオークションに出品されたら，Yahoo!オークションのサイトに行かなくても出品状況を知ることができる．

このように，WS-Schedule Manager では既存の Web サービスを変更することなく，一定時間おきに Web サービスを呼び出すタイマー機能を利用できる．また，一定時間おきに情報を得ることで，対象 Web サービスから得られた情報の変化なども見ることができる．

4.3　次発列車表示サービス

駅やビルなどの公共施設に設置されている，次発列車の時刻表示パネルを考える．次発列車表示パネルでは，現在の時刻に応じて，次発列車に関する情報 (列車の種類，列車番号，行き先，発車時刻) を表示する．このような機能を持ったソフトウェアを作成しようとした場合，定期的に現在表示されている列車が発車時刻に達しているかどうかを確認するプログラムを作成しなければならない．

このようなシステム構築において提案フレームワークを用いることを考える．一定時間おきに処理を繰り返す機能は，WS-Schedule Manager に Web サービスを一定時間おきに呼び出すスケジュールを登録することでまかなうことができる．あとは，現在の時刻に応じて次発列車の情報を提供する Web サービスを作成するだけでよい．

WS-Schedule Manager を用いることで，従来のシステムと比較して実装しなければならない機能が少なくてすむようになる．また，作成するそれぞれの Web サービスは，1 つのサービスにつき 1 つの意味のある機能のみについて作成すればよいため，開発したいシステムについてのプログラムの設計がシンプルになる．さらに，Web サービスはオープンなインターフェースを持っているので，作成した Web サービスは他のプログラムからも利用することができ，今後のソフトウェア開発でこの機能を再利用することも可能になる．

5. 考　　察

5.1 提案フレームワークの特長

本フレームワークを用いることで，要求/応答型で Web サービスを呼び出すだけではなく，呼び出したい時間を指定して呼び出す単発スケジュール，一定時間おきに Web サービスを呼び出す定期スケジュールを登録することができ，これらのスケジュールを時間駆動で呼び出せるようになった．このことにより，時間駆動で動作できない通常の Web サービスに対して，時間差でそれらを呼び出す “タイマー機能” を付加することができるようになった．このタイマー機能を複数連携させることで，前章で述べためざましサービスや，次発列車表示サービスといった，新しいサービスを作成することが可能になった．

EDA では，動作のきっかけとなるイベントは多様で，さまざまなイベントを網羅的に考慮する必要があり，システムが非常に複雑・膨大にならざるを得なかった．しかし，時間駆動イベントについて提案フレームワークを用いれば，コンパクトで軽量なフレームワークで時間イベントを容易に扱うことができるようになった．そのことにより，性能に制約があるコンピュータにも組み入れられる可能性が広がる．また，時間駆動に特化したフレームワークであるため，定期的なスケジュールも登録できる．

また，提案フレームワークを用いれば，一定時間おきに処理を行うようなソフトウェア開発において，その部分の機能を WS-Schedule Manager で実行できるため，新たに開発するソフトウェアに組み込む必要がなくなる．すなわち，単一の機能を持ったサービスを構築し，本フレームワークを用いてそのサービスを定期的に呼び出すといった設計にすればよい．そして，このようにして作成したサービスは，それぞれが単一の意味を持つように設計しているため，他のソフトウェアから再利用しやすくなっている．これらのことから，システムの開発量を抑え，理解しやすいシステム構成にすることが可能となる．

5.2 提案フレームワークの限界

本フレームワークを利用する際の限界としては，本フレームワークを動作させるためのコンピュータを設置する必要がある点があげられる．特に，ホームネットワークに設置する場合は，家庭にそのようなコンピュータが設置されていない場合が多い．ホームネットワークで動作させるためには，セットトップボックス (STB) やハードディスクレコーダなどの家電製品で動作させることで，そのようなコンピュータを確保できると考えられる．ただし，ホームネットワークのような Web サービスの公開範囲が限られている環境ではなく，誰でも使えるように WWW 上で一般に公開されている Web サービスを呼び出すのであれば，本フレームワークを Web 上に設置しておき，自由に利用できるような状態で公開しておくことで，それを利用して時間駆動での Web サービス呼び出しを行うこともできると考えられる．

6. おわりに

本稿では，時間駆動により Web サービスを呼び出すフレームワークである “WS-Schedule Manager” を提案した．イベント駆動の中でも特に時間駆動に着目し，それに特化させることで，時間イベントのみを考慮してシステムを構築できるため，システムの仕様を簡易化できるようになった．また，本フレームワークにより，時間駆動にカスタマイズされた便利な機能を提供することができた．例えば，Web サービスのインターフェースを持った家電にタイマー機能を付加させたり，定期的に実行されるイベントを容易に登録できるようになった．

今後の展望として，現在本フレームワーク内のスケジュール DB では，Outlook のような一般的なスケジュール管理ソフトでも用いられている iCalendar という仕様に基づいたフォーマット設計によるスケジュールを用いているため，WS-Schedule Manager と一般的なスケジューラで同じスケジュールを入力し，連動させることができると考えられる．また，一方で入力したスケジュールをもう一方で読み込ませることで，よりスケジュール管理が容易になると考えられる．

謝　　辞

この研究は，科学技術研究費 (若手研究 B 18700062)，21 世紀 COE プログラム「NAIST-IS:ユビキタス統合メディアコンピューティング」, および，沖電気工業株式会社の助成を受けて行われている．

文　　献

[1] Apache WebServices Pubscribe 1.1, http://ws.apache.org/pubscribe/ , Oct. 2005.
[2] Y. Huang and D. Gannon, “A Flexible and Efficient Approach to Reconcile Different Web Services-based Event Notification Specifications,” IEEE International Conference on Web Services (ICWS 2006), pp.735-742, Sept. 2006.
[3] 井垣 宏, 玉田 春昭, 中村 匡秀, 松本 健一, “サービス指向アーキテクチャを用いたホームネットワークシステムの設計と評価尺度,” 電子情報通信学会技術研究報告, ネットワークシステム研究会, No.NS2003-359, pp.333–338, Mar. 2004.
[4] Internet Calendaring and Scheduling Core Object Specification (iCalendar), RFC2445, http://www.ietf.org/rfc/rfc2445.txt , Nov. 1998.
[5] L. Li, W. Chou, F. Liu, and D. Zhuo, “Semantic Modeling and Design Patterns for Asynchronous Events in Web Service Interaction,” IEEE International Conference on Web Services (ICWS 2006), pp.223-230, Sept. 2006.
[6] M. Nakamura, A. Tanaka, H. Igaki, H. Tamada, and K. Matsumoto, “Adapting Legacy Home Appliances to Home Network Systems Using Web Services,” IEEE International Conference on Web Services (ICWS 2006), pp.849-858, Sept. 2006.
[7] OASIS Web Services Notification (WSN), http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsn , Oct. 2006.
[8] M. P. Papazoglou and D. Georgakopoulos, “Service-Oriented Computing,” Communications of the ACM, Vol. 46, No.10, pp.25-28, Sept. 2003.
[9] 田中章弘, 中村匡秀, 井垣宏, 松本健一, “Web サービスを用いた従来家電のホームネットワークへの適応,” 電子情報通信学会技術研究報告, Vol.105, No.628, pp.067-072, Mar. 2006.
[10] Web Services Activity, http://www.w3.org/2002/ws/ .
[11] Web Services Description Language (WSDL) 1.1, http://www.w3.org/TR/wsdl, Mar. 2001.
[12] What Is Service-Oriented Architecture, http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html, Sept. 2003.
[13] Yahoo!デベロッパーネットワーク Yahoo!オークション, http://developer.yahoo.co.jp/auctions/ .