2004－UBI－5　(10)
2004／6／21

コンテクストの構造化に関する考察とコンテクストアウェアフレームワーク

山田大輔、中村竜也、中山一美
株式会社 NTT データ　技術開発本部
{yamadad, nakamurattc, nakayamakzm}@nttdata.co.jp

さまざまなセンシング技術を基盤とするユビキタスコンピューティング環境が整備されるのに従い、ユーザの置かれた状況や環境、いわゆるコンテクストに応じたサービスや情報を提供するコンテクストアウェアコンピューティングが一般的になるものと予想される。本稿では、既存のコンテクストの構造化アプローチについて考察する。既存アプローチにおける問題点を指摘し、本稿が提案するアプリケーション主導のコンテクスト構造化アプローチについて解説する。また、OWL をコンテクストの構造共有基盤とするコンテクストアウェアなサービス提供フレームワーク YACAN（Yet Another Context AwareNess）を提案する。

Constructing a ContextAware Framework through Structured Context Representation

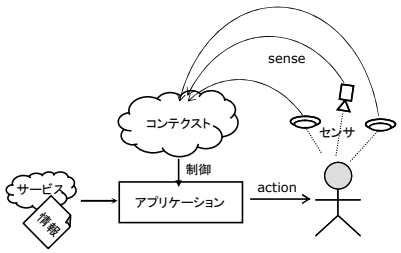
Daisuke Yamada, Tatsuya Nakamura, Kazumi Nakayama
NTT DATA CORPORATION, Research and Development Headquarters
{yamadad, nakamurattc, nakayamakzm}@nttdata.co.jp

Ubiquitous computing environment based on various sensing technologies has been improved. It is expected that context aware computing which provides services and information according to user location, environment, other context, becomes general. In this paper, we describe a consideration of existing structured context approaches and propose a new application based structured context. Furthermore, we propose Context Aware Framework called ″YACAN″ which adopts OWL as the base technorogy to share structured context information.

1 はじめに

ユビキタスあるいはパーベイシブコンピューティングなど様々なコンセプトの元に、位置情報や環境情報など多様なセンシング情報を活用しユーザのタスクをサポートするシステムの研究開発が進められている。これらの研究開発は、RFID を利用したトレーサビリティ実験に代表されるセンシングネットワークインフラやセンシング情報に基づく動的なサービス連携など、いくつかの基盤技術から成り立っているが、決定的なシステムアーキテクチャ提案には至っていない。しかし、いくつかの研究開発や基盤技術開発が実用化されるに従い、ネットワーク上に存在するコンテンツやアプリケーションといった仮想的なリソースと実世界にモノとして実在するリソースが、いつでもどこでも境界なく融合しユーザのタスクをサポートできる可能性が出てきた。

ユビキタスあるいはパーベイシブコンピューティングには、ユーザは目標としているタスクを処理するために計算機を利用するのではなく、ユーザを取り巻く環境や状況を計算機が把握し、タスクの処理に最適で環境や状況に応じたサービスや情報を提供するといった「人間中心の情報サービス環境」[1]がコンセプトとして含まれている。コンテクストアウェアコンピューティングは、環境や状況といったユーザを取り巻く情報をコンテクストとし、そのコンテクストに気づく（アウェアする）ことで、ユーザの目標としているタスクに最適な情報やサービスを提供するためのユビキタスあるいはパーベイシブコンピューティングにおける情報サービス概念である（図－ 1参照）。



図－ 1 コンテクストアウェアコンピューティング

現在進められているコンテクストアウェアコンピューティングの代表的な研究開発プロジェクトとしては、ジョージア工科大 Aware Home Research Initiative[1]やマサチューセッツ工科大メディアラボ Context Aware Computing グループ[2]などがある。これらコンテクストアウェアコンピューティングの課題は実世界あるいは仮想世界の対象について、対象のコンテクストを計算機可読な形で表現することに尽きる。この表現についての一般化が困難な背景には、コンテクストという語の意味が「（事件などの）周囲の事情、環境、背景、状況、（文章などの）前後関係、文脈」といったように多義かつ曖昧であり、適用領域に

応じて異なった解釈が可能な点がある。

本稿では、代表的なコンテクストアウェアコンピューティングの研究開発事例であるモデル主導の観点からコンテクスト構造化アプローチを考察する。考察では、モデル主導アプローチの基底となるコンテクストエンティティの分類に関する問題点を指摘する。この考察から、本稿ではアプリケーション主導のコンテクスト構造化アプローチを提案する。

また本稿では、OWL を利用するアプリケーション主導のコンテクスト構造化アプローチに基づくコンテクスト記述フレームワークを提案し、このフレームワークで記述されたコンテクストの適用や変更を監視し、コンテクストに従うサービスや情報を提供するためのコンテクストアウェアフレームワーク YACAN(Yet Another Context AwareNess)を提案する。

本稿では以降、関連研究の考察とアプリケーション主導のコンテクスト構造化(2章)、コンテクストアウェアフレームワークの概要(3章)、YACANフレームワーク(4章)について示し、最後にまとめと課題を示す。

2 関連研究

本章では、コンテクストの構造化アプローチとして代表的なモデル主導の研究開発事例について考察し、本稿が提案するアプリケーション主導なコンテクストの構造化アプローチと比較し考察する。

2.1 モデル主導のコンテクスト構造化アプローチ

ジョージア工科大の AHRI(Aware Home Research Initiative)[1]では、家庭内に設置されたさまざまなセンシングデバイスの情報を活用する日常生活支援環境の研究開発を進めている。その中で、状況に応じたアプリケーションを提供するフレームワークとしてコンテクストアウェアコンピューティングを捉えている。

AHRI におけるコンテクストアウェアコンピューティングについて A. Dey は、

- コンテクストは実体(人や場所、情報、モノ)の状況(状況や環境、背景)を特徴付けるすべての情報である。
- システムは、コンテクストを利用しユーザのタスクにとって適切な情報やサービスを提供するのであればコンテクストアウェアである。

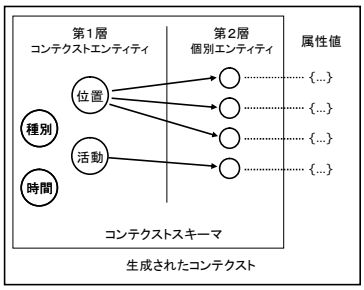
と定義している[3]。定義より、システムはコンテクストを利用してサービスや情報を提供する場合にコンテクストアウェアであると捉えられる。つまり、対象に関するコンテクストの構造化が前提となりコンテクストアウェアなシステムを構成するアプローチであるため、上記定義に基づくコンテクストアウェアなシステムの解釈をモデル主導であると呼ぶこととする。

この定義に従うコンテクストの構造化について、

- コンテクストは2階層の構造と仮定する。
- 第1の階層は状態を特徴付けると考えられる位置や種別、時間、活動の基礎的な四つのコンテクストエンティティにより構成する。
- 第2の階層は、個別エンティティにより構成される。個別エンティティはコンテクストエンティティを索引とし、ドメインごとの個別属性を現すエンティティである。
- 第2階層の個別エンティティは属性に従う実体(値)を持つ。

という仮説を設定している。

この構造化に従うコンテクストは、コンテクストエンティティに従い対象の詳細な属性を示す個別エンティティを抽出し、個別エンティティに値を与える。つまり、コンテクストは四つの基本的属性の組み合せで表現されると仮定し、基本属性のそれぞれを拡張して対象の具体的な属性を与える。これがコンテクストのスキーマとなり、コンテクストのスキーマに従って属性値を設定することでコンテクストを生成する(図－ 2参照)。



図－ 2 コンテクストの構造とコンテクスト生成

たとえば、あるユーザの外界をコンテクストとして構造化する場合には、外界に関する位置や種別、時間、活動に関する個別エンティティを定義する。個別エンティティに関して制約事項はなく、ドメインごとに個別属性を現すエンティティを決定してかまわない。

2.2 モデル主導アプローチについての考察

モデル主導のコンテクスト構造化アプローチのメリットは、コンテクストエンティティを用いた個別エンティティの高い拡張性にある。また、位置と種別、時間、活動の四つの観点からコンテクストを構造化するため、基本的な構造はどのようなコンテクストでも異なることがない。そのため、同様の構造化手法に基づくコンテクストであれば、共通な処理手続きによりコンテクストの共有化、コンポーネント化が実現できる。

ところで、H. Wu は[4]で、A. Dey のモデル主導アプローチの適用では、コンテクストエンティティの定義が曖昧であることや、四つのコンテクストエンティティのいずれかを組み合わせて表現するような境界的要素の構成が必要になる場合、コンテクストエンティティ自体を拡張する必要が生じる点やコンテクストエンティティ間あるいは個別エンティティ間の依存関係を記述するなど構造記述の複雑化が生じる点を指摘している。そこで、コンテクストの構造記述の複雑化を避けるため、モデ

ル主導の構造化アプローチを取りながら、対象ドメインに応じて異なった解釈に基づくコンテクストエンティティを導入し構造化を行っている。H. Wu も[5]で、Location、Proximity、Time、Person、AudioVisual、Computing&Connectivity という異なるコンテクストエンティティにより構造化を行っている。

以上より、モデル主導のコンテクスト構造化アプローチにおけるコンテクストエンティティの項目分類とコンテクストの構造化対象の間には依存性を排除できない。そのため、コンテクストアウェアコンピューティングが、コンテクストに応じたサービスや情報をなんらかのアプリケーションにより提供することが目的であるとすれば、アウェアされるコンテクストのコンテクストエンティティの分類と、それをアウェアするアプリケーションの間には依存関係があることになる。

モデル主導のコンテクスト構造化アプローチでは、コンテクストエンティティの共通性が低く曖昧であるため、対象とする領域ごとにコンテクストエンティティの分類を捉えなおさなくてはならない。モデル主導アプローチのメリットは、共通なコンテクストエンティティに基づくコンテクストの共有化、コンポーネント化にあることを示したが、コンテクストエンティティの共有が前提となる。適用対象ごとにコンテクストエンティティが異なるとすれば、対象に閉じるコンテクストの共有化、コンポーネント化は達成できたとしても、異なる対象間では前述のメリットは得られないと判断できる。

2.3 アプリケーション主導のコンテクスト構造化アプローチ

前節で示したように、モデル主導によるコンテクストの構造化アプローチでは、コンテクストエンティティの分類と対象を捉える観点の間には依存関係が生じる。これは、コンテクストを構造化するための基底概念は対象ごとに異なってしまうことを示唆している。この基底概念は、コンテクストを扱う側つまりアプリケーションとしてどのような状況を扱うかによって決定されるものと考えられる。

本稿では、アプリケーションが扱うコンテクストの基底概念は一定である必要はなく、アプリケーションが対象とする領域に応じて基底概念を自由に設定したとしても、コンテクストを拡張したり異なる領域間で共有できるフレームワーク化が行えるのではないかという仮定のもとにアプリケーション主導のアプローチを提案する。

アプリケーション主導のコンテクスト構造化アプローチでは、

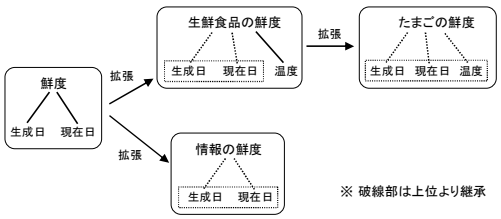
- アブストラクトなコンテクストが唯一存在する。
- コンテクストはアブストラクトなコンテクストを拡張して定義する。任意のコンテクストは拡張可能である。
- コンテクストの定義はコンテクスト属性により定義する。コンテクスト属性は、コンテクストが対象を観測するために指示する個別属性である。
- コンテクストを適用する対象の概念構造は既知である。

を仮説として設定する。

コンテクストに共通する基本的な属性や振る舞いが定義された、アブストラクトなコンテクストが存在し、このアブストラクトなコンテクストを拡張することで対象に適用するコンテクストを生成する。また、任意のコンテクストは拡張可能であるとし、同じ概念のコンテクストを異なる対象に適用するため拡張してかまわないものとする。

コンテクスト属性は、コンテクストに定義されるローカルな属性定義であり、この属性が個別対象のどの属性を参照するか指定する。つまり、コンテクスト属性により指定された対象の属性を指示することで、このコンテクストを利用するアプリケーションの観点から対象を観測する。そのため、対象の概念構造は既知である必要がある。

たとえば、鮮度というコンテクストを考えてみる。鮮度という概念は、生活世界における生鮮食品の状況を示すばかりでなく、情報など仮想世界においても状況を示す特徴として適用可能である。鮮度は新しさ、古さを示す数値の概念であるので、コンテクスト属性としては、生成日と現在日により個別対象を観測する。生活世界における生鮮食品を対象とする場合、生鮮食品の鮮度は生鮮食品をとりまく環境の温度にも依存すると仮定すれば、“生鮮食品の鮮度”は先の鮮度を拡張して、ローカルなコンテクスト属性として温度を定義する。たまごの鮮度は生鮮食品の鮮度を、コンテクスト属性を定義せず拡張する。仮想世界の例として web ページを対象とする場合、web ページの鮮度は生成日と現在日で指定できるので、コンテクスト属性を定義せず鮮度を拡張する。すると鮮度というコンテクストに関して図－ 3に示すような構造化が行える。

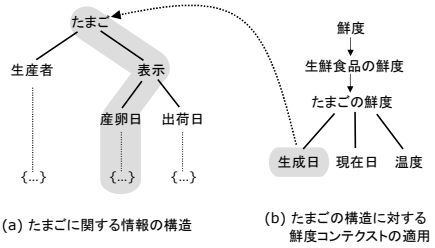


図－ 3　鮮度コンテクストの構造

このとき、たまごに関する情報の構造が図－ 4(a)として既知である場合、たまごの鮮度は図－ 4(b)に示すように適用される。たまごに関する情報の構造とたまごの鮮度に定義されるコンテクスト属性は異なるが、産卵日と生成日は意味的に同様であり適用可能であると判断できる。

H. Wu は[4]で、モデル主導アプローチのメリットとしては、コンポーネントとしての共通性が高く、ユビキタスコンピューティング環境との親和性が高いとしている。また、アプリケーション主導アプローチは、閉じた領域における適用性は高いが、分散しているコンテクストを扱う場合にはコンテクストの共有性が低く、ユビキタスコンピューティング環境との親和性が低い

と解説している。



図－ 4 たまごの情報構造へのコンテクスト適用

ところで、モデル主導アプローチのコンテクストエンティティは確かに共通であり一見コンポーネント化を達成しているようであるが、そのためには実世界のコンテクストを漏れなく記述するためのコンテクストエンティティを規定しなくてはならない。先に指摘したように、コンテクストエンティティが対象とする領域ごとに解釈を変え分類しなおさなくてはならないとすれば、コンテクストエンティティを規定することは不可能である。あるいは、A. Dey によるコンテクストエンティティに従ったとしても、コンテクストエンティティの曖昧性から対象とする領域ごとにことなる意味的解釈のもとに個別エンティティを付与することになるため、コンテクストの意味的な解釈共有が困難となる。

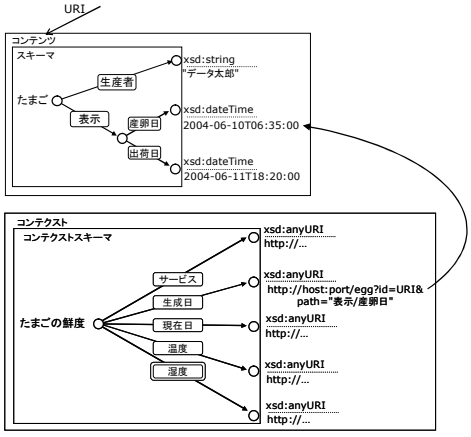
筆者らは、アプリケーション主導アプローチの欠点として指摘されたコンテクストの共有性の低さは、コンテクストの構造を共有するフレームワーク:OWL を援用することで解決できるものと判断している。次章以降、アプリケーション主導のコンテクスト構造化アプローチに従うコンテクストアウェアフレームワークについて詳述する。

3 フレームワークの概要

3.1 対象へのコンテクスト適用とコンテクスト生成

これまで対象として扱ってきた既知な構造を持つ情報を以降コンテクストと対比してコンテンツと呼ぶこととする。実世界あるいは仮想世界にかかわらず、コンテンツは既知の構造とその構造に従う値を持つものとし、コンテンツを参照するための URI を持つものとする。RFID タグにより参照可能な食品の情報や ISBN など実世界の情報に限らず、URL により参照可能な web ページや web サービスもコンテンツとして扱う。コンテクストも構造を持つ情報であるからコンテンツとして捉えることができる。よって、コンテクストはコンテクストを用いて構造化することが可能である。

たまごのコンテンツとたまごの鮮度に関するコンテクストの関係を図－ 5に示す。たまごのコンテンツは URI により参照可能で、個々の属性値はコンテンツの構造に従いアクセスできるものとする。つまり、コンテンツは URI により指示される元ノードとラベル付き有向グラフにより構造化される。属性値にアクセスする場合には、URI と元ノードから属性を辿るなんらかの経路記述の対で指示されるものとする。個々の属性値は XSD に従い与えられることを想定している。



図－ 5 たまごの鮮度コンテクストの生成

たまごの鮮度に関するコンテクストでは、コンテクスト属性として生成日と時刻、温度を持つものとする。コンテクスト属性は属性値の型を任意の URI とする。これは、コンテクストを適用するコンテンツへの参照が URI により行われると規定したことに従う。このように、コンテクストはコンテクストを適用するコンテンツに依存しない。

基底コンテクストは、コンテクストが適用されるコンテンツが発見されたり、コンテンツの更新が行われた場合に提供する情報サービスに関するコンテクスト属性:サービスを持つ。このコンテンツ属性は実装に依存すると思われるため次章にて詳述するものとする。

たまごのコンテンツにたまごの鮮度に関するコンテクストを適用(生成)するとは、個々のコンテンツが持つ実体としての値を参照することであるから、たとえば、コンテクスト属性:生成日がコンテンツの生産日を参照するよう URI とたまごの属性:生産日へのアクセスに必要な元ノードからの経路を用いて定義すればよい。

3.2 OWLによるコンテクスト記述

2.3で示したように、アプリケーション主導のコンテクスト構造化アプローチの欠点として、コンテクストの共有性があげられている。コンテクストとしてのある構造記述が他の構造記述から捉えられたときどのような関係にあるかを把握できることが、コンテクストの共有性を決定付けるものと考えられる。モデル主導のコンテクスト構造化アプローチでは、コンテクストエンティティにより一定の構造を与え、異なる適用領域間でも同じコンテクストエンティティには同じ意味から派生する個別エンティティが定義されていることを保証している。アプリケーション主導のコンテクスト構造化アプローチでは、アプリケーショ

ンに応じた基底コンテクストを導入してかまわないとしたため、モデル主導アプローチと比較して共有性は低いと判断された。

2.3で示したように、基底コンテクストから派生するコンテクストの構造共有化が達成できれば、モデル主導の構造化アプローチと同等のコンテクスト共有性が確保できるはずである。コンテクストの記述に OWL[6]を導入することで、ある適用領域で定義されたコンテクストが他の適用領域で定義されたコンテクストとどのような関係にあるかを定義することができる。OWL の導入により、アプリケーション主導アプローチでも、コンテクスト共有性が確保でき、モデル主導アプローチの欠点である適用領域依存や曖昧性が排除できる。

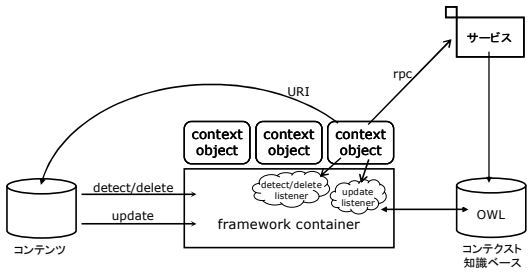
3.3 フレームワークの概要

コンテクストは基底コンテクストを拡張して定義するのと同様、フレームワークにおいても基底コンテクストを具体化するオブジェクトを定義し、このオブジェクトを拡張してフレームワークにおけるコンテクストとしての振る舞いを継承させる。

コンテクストのフレームワークにおける基本的な振る舞いは、

- コンテクスト適用対象コンテンツの検知(detect)による生成
- コンテンツの更新(update)
- コンテンツの消滅(delete)による破棄

をトリガとするイベント駆動である。フレームワークは、コンテクストの基本的な振る舞いをサポートするコンテナとその上で稼働するコンテクストオブジェクトから構成する。図－ 6にフレームワークの概念的構成を示す。



図－ 6 フレームワークの概念構成

コンテナは、コンテクストを管理する知識ベースを持ち、登録されているコンテクストを適用可能なコンテンツが検知されると、該当するコンテクストをコンテナ上に登録する。ロードされたコンテクストは、コンテナ上のコンテクストオブジェクトとして、コンテンツの detect、delete、update イベントを監視し、イベントが通知されると対応するサービスを起動する。

各イベントに対応して起動されるサービスの記述は、コンテクストオブジェクト内に定義される場合もあれば、rpc などを利用し外部サービスを起動する場合も考えられる。外部サービスであってもサービス起動のトリガとなるコンテクストが決定されている場合にはコンテクストを管理する知識ベースにコンテクストを登録することができる。つまり、既存のサービスであってもトリガとなるコンテクストが記述できるのであれば、コンテクストアウェアフレームワークを利用することが可能である。

次章では、Java を実装プラットフォームとして具体化したコンテクストアウェアフレームワーク YACAN の詳細を示す。

4 コンテクストアウェアフレームワークYACANの構成と仮想モデルによる検証

本章では、コンテクストアウェアフレームワークYACANを提案し、前章で示したフレームワークの具体化を行う。さらに、仮想モデルとして「たまごの鮮度を検知してサービスを提供する」というシーンを設定し、フレームワークの検証を行う。

4.1 フレームワーク概要

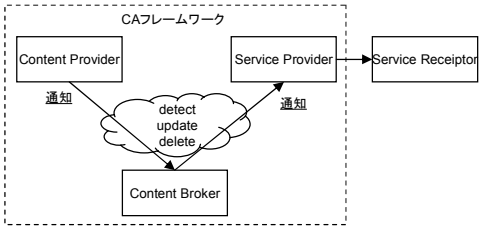
4.1.1 基本構成要素

図－ 6で示したフレームワークの概念構成では、コンテンツの変化イベントをコンテナが監視し、イベントを受け取ると対応するサービスを起動する。ここで、コンテンツからコンテナへのイベントの通知にはコンテンツの特性により様々な方式があると考えられる。たとえば、メールによる通知や Java イベントによる通知などが考えられる。通知を行うタイミングについても、イベントが発生した場合に即座にコンテナへと通知する場合や、イベントが発生しても直ぐには通知を行わずにコンテナ側からの問い合わせに応じて通知する場合などが考えられる。コンテナはコンテクストを適用するコンテンツに応じてイベント通知を受け取るインタフェースを持つ必要がある。

コンテクストアウェアフレームワーク YACAN では、上記に示したコンテナとコンテンツの方式依存性を排除するために ContentProvider、ServiceProvider、ContentBroker の三つの要素により構成することとする(図－ 7参照)。コンテンツとコンテナの間にイベントの通知方式を隠蔽するインタフェースを置くことで上記の問題を解決する。各要素の役割は、

- ContentProvider
  システムで取り扱うコンテンツを管理し、コンテンツが変化した場合に ContentBroker へ通知する。
- ServiceProvider
  サービスで利用するコンテクストを管理し、ContentBroker からコンテクストの変化を検知した場合に対応する情報やサービスの提供を行う。
- ContentBroker
  ContentProvider に対してはコンテンツの変化を公開するための手段を提供し、ServiceProvider に対してはコンテンツの変化を検知するための手段を提供する。

である。



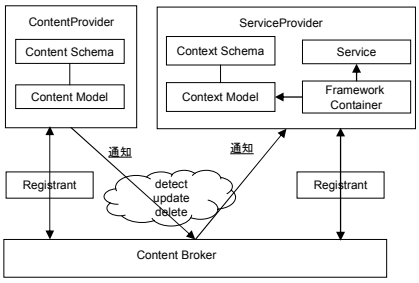
図－ 7 フレームワーク基本構成

本構成では、ContentProvider は管理するコンテンツのイベントを ContentBroker に対して通知する。ServiceProvider ではコンテクスト属性で指定されたコンテンツを ContentBroker へリスナー登録しておけば、そのコンテンツのイベントを ContentBroker から検知することができる。ContetnBroker は、ContentProvider の通知方法の多様性を吸収し、ServiceProvider へ既知の検知方法を提供している。

ServiceReceiptor は ServiceProvider によって提供されるサービスを利用する側であり、サービスによりその仕様は様々である。コンテクストアウェアフレームワークでは、コンテンツの変化により任意のサービスを起動する仕組みを提供するものであるので、ServiceReceiptor はコンテクストアウェアフレームワークの提供範囲外である。

4.1.2 構成要素詳細

図－ 7で示した基本構成図を元に、機能を詳細化したものを図－ 8に示す。



図－ 8 フレームワーク詳細構成

詳細化された要素は次のようになる。

- ContentSchema
  システムで扱うコンテンツの構成の定義。
- ContentModel
  ContentSchema を元に実体化したコンテンツ。
- ContextSchema
  サービスで利用されるコンテクストの構成の定義。また、対応するサービス指定もここで定義される。
- ContextModel
  ContextSchema を元に実体化したコンテクスト。
- FrameworkContainer
  コンテクストの変化を検知して、対応する ContextModel を参照してサービスを実行するための実行環境。
- Service
  提供する情報サービスロジック。
- Registrant
  ContentProvider と ContentBroker 、または ServiceProvider と ContentBroker の関連付けを行う。

4.1.3 サービス記述方式

2.3で、全てのコンテクストはある基底コンテクストの派生クラスであり、その基底コンテクストでは、コンテンツが発見されたり、コンテンツの更新がおこなわれた場合に提供する情報サービスに関するコンテクスト属性:サービスを持つとした。コンテクスト属性:サービスではサービスの場所を示す URI と、サービスが動作するプラットフォームなど実行のための情報を示す。サービスの詳細を記述する方法として、web サービスが提供する機能を記述した WSDL や、MIME のようにサービスの種別を type で指定することなどが考えられる。

本稿ではサービスの記述方法に関しては未検討である。そこで本稿では、コンテクスト属性:サービスで、サービスを実装したクラスを URI で指定することとする。Framework Container は URI で指定されたクラスをロードしてサービスを実行する。

4.2 仮想モデルによる検証

先に示したコンテクストアウェアフレームワークに対して、仮想モデルを設定して検証を行う。

仮想モデルとして、「新規に購入したたまごを冷蔵庫に保管する。冷蔵庫はたまごのこれまでの保管履歴と冷蔵庫内での状態を追跡して、所定の“鮮度”が失われると利用者に通知する。」ことを設定する。

4.2.1 仮想モデル要素

仮想モデルの構成要素を表 1に示す。仮想モデルは大分類として、たまごの保管履歴管理を行うトレーサビリティシステムと冷蔵庫内の管理を行う冷蔵庫システムから構成される。トレーサビリティシステムはたまごの履歴管理とともに、内部にたまごのコンテンツを管理するたまごコンテンツ(ContentProvider)をもつ。冷蔵庫システムはたまごの鮮度に従ってサービスを提供するたまご鮮度サービス(ServiceProvider)、冷蔵庫内の時刻を管理する時刻コンテンツ(ContentProvider)、同じく冷蔵庫内の温度を管理する温度コンテンツ(ContentProvider)をもつ。また、コンテンツとサービスのやり取りの仲介を行うブローカ(ContentBroker)、冷蔵庫内に新しく保管された物品を検知しコンテンツをブローカへ登録する新規物品登録(Rgistrant)も冷蔵庫システムでもつこととする。

表 1 仮想モデル要素

No	システム	コンポーネント	業務
1	トレーサビリティシステム	たまご コンテンツ	たまごのコンテンツを管理する ContentProvider。
2		たまご 保管履歴	流通の過程で通過していく各種の保管環境の履歴を管理する。
3	冷蔵庫システム	たまご鮮度 サービス	たまごの鮮度に関する ServiceProvider。庫内物品の出し入れ、温度、時間の変化を検知し、「鮮度」に対応するサービスを提供する。
4		時刻 コンテンツ	時刻を管理する ContentProvider。
5		温度 コンテンツ	温度を管理する ContentProvider。
6		ブローカ	たまごコンテンツ、時刻コンテンツ、温度コンテンツ、たまご鮮度サービスを結びつける ContentBroker。
7		新規物品登録	冷蔵庫内に新しく保管された物品を検知し、ContentProvider と ContentBroker の登録を行う Registrant。

4.2.2 仮想モデルシナリオ

上記の仮想モデルを用いた検証シナリオを以下に示す。

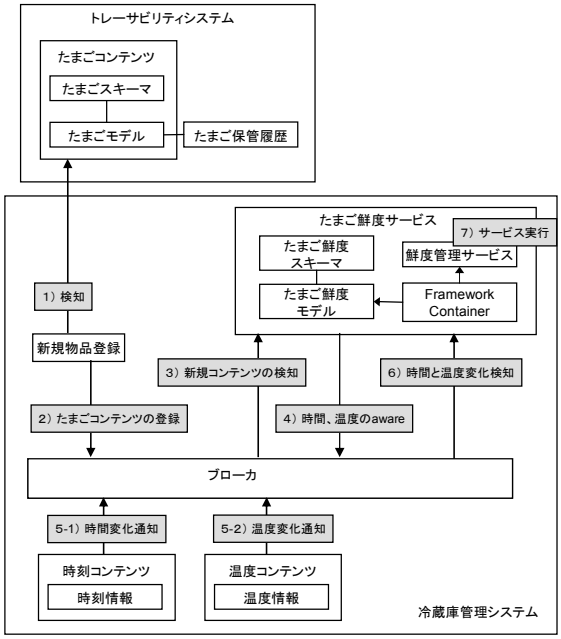
(1) 冷蔵庫システムの新規物品登録コンポーネントが冷蔵庫内で保管対象となるたまごを検知する。
(2) 新規物品登録コンポーネントはブローカコンポーネントにたまごコンテンツコンポーネントを登録する。
(3) たまご鮮度サービスコンポーネントは新規物品が登録されたことを検知する。
(4) たまご鮮度サービスコンポーネントはたまごコンテンツを監視対象とし、時間・温度をブローカコンポーネントに対してリスナー登録する。
(5) 時刻変化が時刻コンテンツコンポーネントから、温度変化が温度コンテンツコンポーネントからブローカコンポーネントに通知される。
(6) たまご鮮度サービスコンポーネントが時刻・温度変化をブローカコンポーネントから検知する。
(7) FrameworkContainer が鮮度管理サービスを実行する。

※ ただし前提として、冷蔵庫システム内の各コンポーネントとブローカコンポーネントは、予め関係付けられているものとする。

4.3 OWLによるコンテンツ、コンテクストの記述例

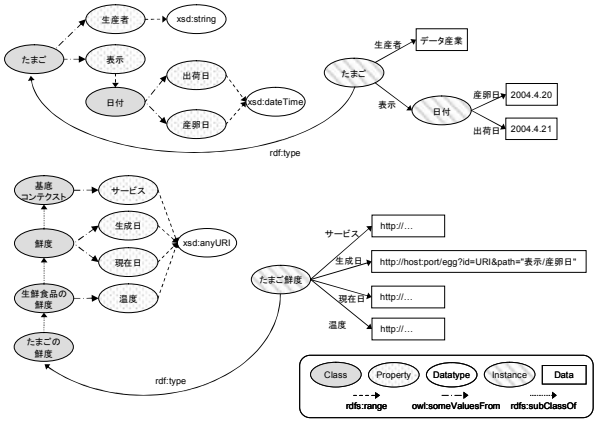
オントロジ記述言語 OWL によるコンテンツとコンテクストのモデル例を図－ 10に示す。たまごコンテンツのコンテンツスキーマとコンテンツモデル、たまご鮮度サービスのコンテクストスキーマとコンテクストモデルを OWL モデルで表現したものである。

たまごクラスはたまごの構造を定義するクラスである。たまごクラスは属性として生産者、表示をもつ。属性：生産者は xsd:string 型で保持し、属性：表示は日付クラスを保持する。表示クラスは出荷日と産卵日を属性としてもち、それぞれ xsd:dateTime で保持される。



図－ 9 仮想モデルシナリオ

たまごの鮮度クラスはたまごの鮮度に関するコンテクストクラスである。基底コンテクストクラスから鮮度クラス、生鮮食料品の鮮度クラス、たまごの鮮度クラスへと順に拡張されている。たまごの鮮度クラスではコンテクスト属性として生成日、現在日、温度をもち、それぞれ xsd:anyURI で保持されている。



図－ 10 OWL モデル例

5 課題

5.1 情報サービス提供の頻度・精度制御について

コンテクストを適用するコンテンツの管理とコンテクストアウェアシステムは独立しており、コンテンツ管理側が更新を通知する頻度（精度）と、コンテクストアウェア側がその通知に基づき情報サービスを提供する精度（頻度）は一般に異なっている。

また、コンテクストアウェアシステムがコンテンツ更新頻度を制御できるとは限らない。たとえば、4章で示した冷蔵庫の例では、冷蔵庫は一定の頻度／精度で時間／温度の変化を通知するが、生鮮食品の鮮度はその食品の性質により温度に依存して劣化するのか時間経過に依存して劣化するのか異なるものと考えられる。ある食品では温度変化で鮮度を管理したいし、またある食品では時間変化で鮮度を管理したい。

このように、サービス起動のトリガを与えるコンテクスト属性について、どのような精度（頻度）でサービスを提供するのか定義できるべきであると考えている。これは位置情報や時間、温度、湿度といった実世界を対象としたコンテクスト属性に限らず、仮想世界を含むコンテクスト属性一般に定義可能であるべきである。

筆者らは、サービス提供の頻度／精度をコンテクスト属性に対する制約として考え、コンテクストの構造として記述する方式を検討中である。この制約記述より、情報サービス提供の頻度／精度を制御するためのプログラムを自動生成できるものと考えており、ケーススタディを通じて制約記述言語の構造化を行う予定である。

6 まとめ

モデル主導とアプリケーション主導のコンテクスト構造化について、コンテクストの適用対象依存性や構造共有の観点から考察を行った。H. Wu が指摘する通り、コンテクストの構造共有性は遍在する機器を前提とするユビキタスコンピューティング環境との親和性を決定するものと考えられる。H. Wu がアプリケーション主導アプローチのコンテクスト共有性の低さを指摘しているが、これはOWLを基盤としたオントロジサービスにより回避できるものと期待できる。アプリケーション主導アプローチでは適用対象に非依存な概念的コンテクストが記述できるため、実世界や仮想世界を融合して扱うユビキタスコンピューティングの目標に適合していると考えている。

また、アプリケーション主導アプローチで構造化したコンテクストに基づくコンテクストアウェアフレームワーク YACAN を提案した。YACAN ではコンテクストを OWL に基づくオントロジとして扱うが、セマンティックウェブ記述言語の上位階層であるLogic（ルール）については特に意識しておらず、コンテクストのモデリング言語としての活用に終止している。コンテクストの記述にもルールに関する記述を導入できるものと期待でき、ルールに基づくコンテクスト更新の伝播が実現できるのではないかと考えている。ルールの導入については今後検討したいと考えている。

7 参考文献

[1] Aware Home Research Initiative, Georgia Institute of Technology. http://www.cc.gatech.edu/fce/ahri/
[2] Context Aware Computing group, Media Laboratory, Massachusetts Institute of Technology. http://cac.media.mit.edu:8080/contextweb/jsp/index.htm
[3] Anind K. Dey, “Providing Architectural Support for Building Context-Aware Applications”,PhD Thesis, November 2000, Georgia Institute of Technology. http://www.cc.gatech.edu/fce/ctk/pubs/dev-thesis.pdf
[4] Huadong Wu, “Ph.D. Thesis Proposal: Supporting Sensor Fusion for Context-Aware Computing”, Jun 30, 2001, Carnegie Mellon University.
[5] Huadong Wu, Mel Siegel, and Sevim Ablay, “Sensor Fusion for Context Understanding”, IEEE Instrumentation and Measurement Technology Conference, May 2002.
[6] Web-Ontology(WebOnt) Working Group, World Wide Web Consortium. http://www.w3.org/2001/sw/WebOnt/