XMLアプリケーション開発のためのパターン体系の開発

山本晃治[†] 上原忠弘[†] 松尾昭彦[†]
大橋恭子[†] 猪股順二[††]
松田友隆[††] 山本里枝子[†]

オブジェクト指向技術やXML，webサービスなど新しいソフトウェア開発技術が次々と登場している．しかし現状では新技術が現れてしばらく時間が経過しても，新技術を利用するための全般的な知識が必要とする技術者に広まっていない．この問題を解決するため，我々は1つの試みとして新技術を利用したアプリケーションを開発するための知識をパターン化し体系的かつ相互に関連づけた形態，すなわちパターン体系を開発した．新技術としてXMLを取り上げ，このXMLを利用したアプリケーション開発のためのパターン体系を実際のアプリケーション開発に適用し効果を得た．本稿では我々がパターン体系を開発した過程，体制と得られた効果について述べる．

A Development of a Pattern System for XML Application Development

Kouji Yamamoto,[†] Tadahiro Uehara,[†] Akihiko Matsuo,[†]
Kyoko Ohashi,[†] Junji Inomata,[††] Tomotaka Matsuda[††]
and Rieko Yamamoto[†]

New software development technologies such as OO-technologies, XML, and web services are appearing one after another. However, the knowledge necessary to use these new technologies are not always possessed by software development teams. We have developed a pattern system for the development of applications that use new technologies by interweaving several patterns we wrote. We chose XML as a target technology for the pattern system. We applied our system to practical development projects and obtained good results. This paper describes the process, the organization, and the effect of our pattern system.

1. はじめに

オブジェクト指向技術やXML，webサービスなど，次々に新しいソフトウェア開発技術が登場している．新しい技術をキーワードにしたビジネスチャンスを獲得するために，各企業はこれらの技術を適用したアプリケーション開発を確立しなければならない．

しかし新しいソフトウェア開発技術が登場してしばらく時間が経過しても，システム開発に関わる部署の中に新技術に関する全般的な知識を持っている者はそう多くはいない．また新技術に関する知識を持つ者が各開発部署に均等に出てくるわけではない．そのため，新しいソフトウェア開発技術の利用を検討できない，または新技術を利用することの開発リスクが大きいため利用を見送る，といった事態が起こる可能性が高い．

新しいソフトウェア開発技術に関するノウハウの集約と発信は新しいソフトウェア開発技術全般が必要としている課題である．この解決策として企業内オブジェクト指向相談室を設立しコンサルティング環境を構築する活動[1)]などが企業で行われている．

我々はこの課題に対し1つの試みとして，新しいソフトウェア開発技術を用いたシステム開発のノウハウをパターン化し体系的かつ相互に関連づけた形態，すなわち新技術を利用するシステム開発のパターン体系を開発した．

新しいソフトウェア開発技術として我々はXMLを取り上げた．これは次の2つの理由による．第1にXMLに関する基本仕様，応用技術，APIはひととおり出揃っている．システムの新規開発やリプレースではXMLの利用を検討することが当たり前になった．第2に，後述するように我々はXMLをアプリケーション開発の1つの革新であると認識しており，さら

† 株式会社富士通研究所ソフトウェアイノベーション研究部
Software Innovation Laboratory, Fujitsu Laboratories Ltd.
†† 富士通株式会社ソフトウェア事業本部MDA技術部
MDA Technology Division, Software Group, Fujitsu Ltd.

に XML を利用するシステム開発では XML のスキーマ設計が新たにキーとなる作業であると考えている．このことから XML を利用するアプリケーション開発に関してパターン体系化する時機と考えた．

我々は XML を利用するシステム開発のパターン体系を開発し，展開した．このパターン体系は実際のアプリケーション開発に適用され，コスト削減効果を示すことができた．

本稿では 2 章で XML を利用するアプリケーション開発作業の特徴を述べ，3 章でパターン体系の概要を述べる．4 章，5 章でパターン体系の開発に関して報告し，6 章で開発したパターン体系の評価を行う．7 章で関連研究について述べ，8 章，9 章で考察とまとめを行う．

2. XML 利用アプリの開発作業の特徴

2.1 従来のデータ表現形式との処理方式の違い

XML は「1 つの新たなデータ表現形式」である．XML データの処理方式が，これまでのデータ表現形式の処理方式と異なるのは，データフォーマットの変更（データの中の項目の増減）にともなうプログラム変更が少ない点である．

これまでのデータ表現形式，たとえば固定長電文や CSV では，データの中の項目が増減してデータフォーマットが変更になるたびにデータを扱うプログラム（データを作成する部分やデータを解釈する部分）を変更する必要があった．

別のデータ処理方式として，データオブジェクトを用いる方式，すなわち Java などのオブジェクト指向言語を用いてデータとそのロジックをオブジェクトの中に隠蔽しデータの中の各項目へのアクセスインタフェースを表に出す方式がある．この方式を用いることでデータフォーマットを変更しても，データを扱うプログラム（無変更の項目を扱う部分）は変更する必要がなくなった．しかしデータを表す Java クラス自体はデータフォーマットの変更にともなって変更し，再コンパイルする必要がある．この手間を省くために開発現場で行われたのは，データの中に用途の決まっていない予備の項目をいくつも用意しておいてデータ項目の増加に備えることだった．

XML はデータ自体を処理するプログラムが DOM [10]，SAX [11]，JDOM [12] などのライブラリとして用意されており，データオブジェクト方式のような再コンパイルは必要ない（もちろんデータフォーマットを変更してもデータを扱うプログラムを変更する必要がない）．データフォーマットの変更が XML 自体を処理するプログラムから分離できているのは，XML の仕様がデータの項目を表すタグを必ず付ける仕様であることと，データの項目を木構造で格納する仕様であることが理由である．

XML 自身を処理する部分が汎用化されており作り直す必要がないという XML の特徴は革新的である．我々がパターン体系で提供すべき情報の 1 つである．

2.2 XML のスキーマ設計作業

XML のスキーマ（データ構造）の設計作業は早い段階で行われる場合が多い．疎に結合されたシステム間でのデータ交換に XML を用いる場合，特に異なる企業間でのデータ交換に XML を用いる場合は，開発の非常に早い段階で XML のボキャブラリ（XML のデータ構造と XML の各要素の意味）を決める．また，既存のシステムの置き換えのためデータ交換する内容がほぼ確定している場合や，既存の RDB テーブルの内容を XML データに変換する場合など，XML のボキャブラリを早くから決めることが可能な場合も多い．

一方 XML のスキーマ設計はデータの構造や型を意識する必要がある．CSV などと異なり，XML はデータを木構造に柔軟な形で格納できる．また XML Schema [13] や RELAX NG [14] など DTD より後に出て広まったスキーマ言語は，XML の中の各データに型を付けることができる．

構造や型を意識したスキーマ設計をどのように行うかも，我々のパターン体系で提供すべき重要な情報である．

3. パターン体系

パターン体系を Buschmann らは文献 5) の 5 章において次のとおり定義している：「ソフトウェアアーキテクチャのためのパターンのコレクションであり，その実装のためのガイドラインやソフトウェア開発における実際の利用やガイドラインを含んでいる」．

我々の開発したパターン体系はこの定義の内容に加え，企業でのソフトウェア開発体制に合わせた情報やガイドラインを取捨選択して盛り込んでいる．以下に詳細を示す．

3.1 対 象 範 囲

我々のパターン体系がカバーする範囲は XML を利用するビジネスアプリケーションであり，その中でもサーバ側の開発に必要な技術に絞っている．これはアプリケーション開発の中で XML を利用することで変更が生じ，ノウハウを提供しなければならない技術がクライアント側（ユーザインタフェース側）ではなくサーバ側に多くあると考えたためである．実際ユーザ

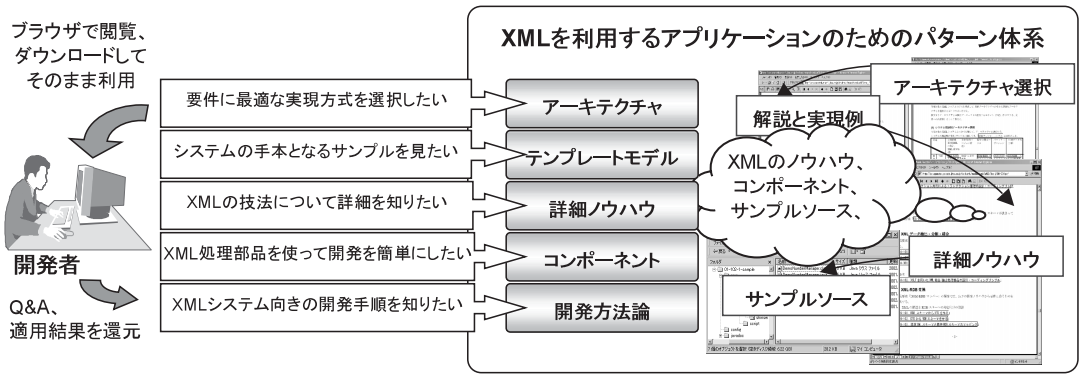


図 1 パターン体系の全体構成
Fig. 1 The overview of the pattern system.

インタフェース側は各社のさまざまなコンポーネントやアプリケーションが出揃っており，これらを使えば要件を満たせる場合が多い．

3.2 パターン体系の構成

我々のパターン体系は個々の技術ノウハウを相互にリンクしたドキュメントとして提供し，膨大なノウハウから必要なノウハウだけを効率的に見ることができる．開発の状況に応じて以下のさまざまな入口（アーキテクチャ，テンプレートモデル，詳細ノウハウ，コンポーネント，開発方法論）からハイパーリンクをたどり，大きな粒度のノウハウから細粒度のノウハウに至ることができる．一覧から直接細粒度のノウハウに至ることもできる．

同じビジネスアプリケーションの開発に関わる人間でも，たとえばプログラマと SE ではアプリケーションに対する観点や必要とするノウハウが異なる．我々は上記のようにさまざまな入口と経路を設けることで，すべての人が「最適なノウハウに最短で」到達できる体系を目指した．

また細粒度のノウハウ，詳細設計や実装に関するノウハウについては，そのサンプルコードとコンポーネントをそのまま利用できる形で提供している．

以下にパターン体系を構成する 5 つの入口を示す（**図 1**）．

(1) XML システムのアーキテクチャ

XML を入出力とするシステムの概要モデルを解説している．XML 自体を処理する方式とデータを格納する方式をそれぞれ分類して特徴やトレードオフを解説している．システムの要件に応じて方式を選択するためのガイドラインも提供している．また各アーキテクチャで使用できる実際の製品名をあげている．

ここから詳細化したアーキテクチャやテンプレートモデル，詳細ノウハウ，コンポーネントに至ることができる．

(2) テンプレートモデル

システムのタイプ別にアプリケーションの実現例とサンプルコード，および，それを用いた開発に必要な作業手順を提供している．

ここから関連するアーキテクチャやテンプレートモデル，詳細ノウハウ，コンポーネントに至ることができる．

(3) 詳細ノウハウ

XML アプリケーションの開発に必要な個々の技法に関する知識とサンプルコードを提供している．ノウハウはテーマ別（RDB 格納法など）に分類してある．

ここから関連するコンポーネントに至ることができる．

(4) コンポーネント

使用頻度の高い XML 処理を部品化しコンポーネントとして提供している．あわせて設計情報，動作保証情報，使用法を示すサンプルコードも提供している．コンポーネントの設定を外部ファイル（XML）で与えてカスタマイズを可能にしてある．

(5) 開発方法論

我々の開発手法[2] をベースに XML を利用するアプリケーション開発のための作業手順（分析作業からテスト作業まで）とドキュメント（UML ベース）を開発した．

開発方法論はパターン体系が提供する 5 つの入口やパターンの実際の使い方を示している．この方法論は入口やパターンをどのソフトウェア開発フェーズで使うか，各開発フェーズでどのように使うか，各開発フェーズではどこまで

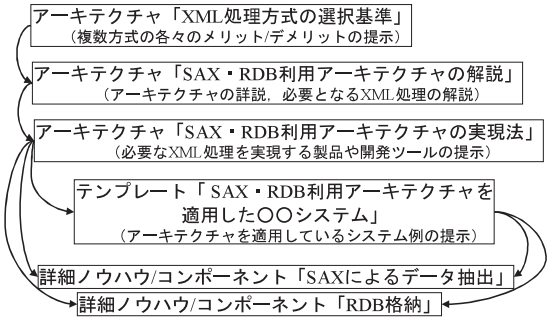


図 2 パターンのたどり方
Fig. 2 How to travel the pattern system.

（たとえばどの詳細度まで）作業すべきか，という情報を提供する．

また代表的なアプリケーション開発を例に全ドキュメント事例も提供している．

このパターン体系のパターンのたどり方の例を簡単に示す（**図 2** 参照）．入口「XML システムのアーキテクチャ」から「XML 処理方式の選択基準」が書かれたパターンにたどり着くことができる．このパターンに従って処理方式を選択し，パターンに張られたハイパーリンクをたどって選択したアーキテクチャ（たとえば SAX と RDB を利用する XML 処理アーキテクチャ）やその実現法に関するパターンを参照できる．さらにこのパターンからアーキテクチャを利用したテンプレートモデルやアーキテクチャを実現するための詳細ノウハウ（たとえば図 3 の SAX に関するノウハウ）やコンポーネントのパターンにたどり着くことができる．

2 章であげたパターン体系で提供すべき情報のうち，2.1 節のような XML の技術的特徴に関する情報は，アーキテクチャ，ノウハウ，コンポーネントを入口とするパターンとして提供している．2.2 節のような開発手法や作業手順に関しては，開発方法論で情報を提供している．

3.3 パターンテンプレート

パターンを統一された形式で記述するために，我々のパターン体系用のパターン記述テンプレート（以降パターンテンプレート）を用意した．記述する項目は社内での利用に必要なものを列挙した．

我々のパターンテンプレートが持つ項目の中で，一般のパターンテンプレート（たとえば文献 5) の 1 章に示されている記述テンプレート☆）にない項目として「必要となるスキル」がある．この項目にはたとえば「DOM API，JDBC，Java に関する知識」と書かれている．この項目を設けることで習得技術の異なる各部署の開発担当者は現在の技術知識で読むべきか否かを判断できる．

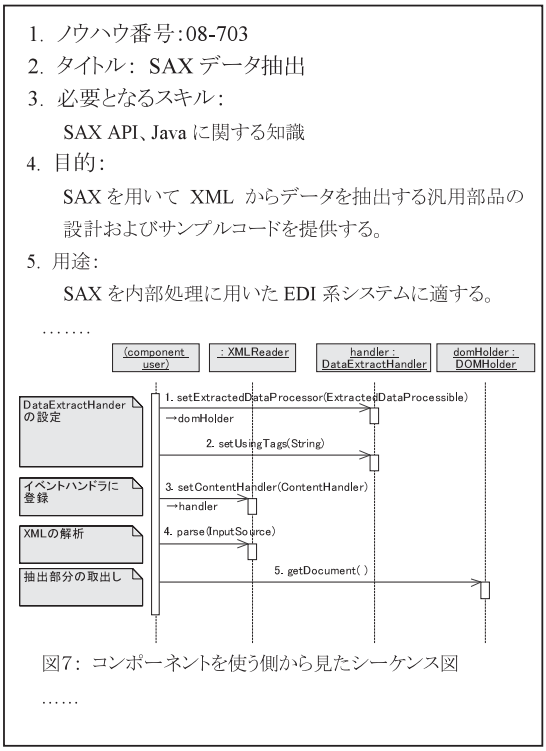


図 3 パターンの例
Fig. 3 An example of patterns.

また我々のパターンには，GoF デザインパターン[6]で「動機」に書かれる内容，すなわちパターンが解決すべき問題の文脈をほとんど書いていない．代わりに 3.2 節に示した 5 つの入口のうち「アーキテクチャ」，「テンプレートモデル」にこの内容が書いてあり，説明文中，図，表，リストから適宜ハイパーリンクでパターンに到達できる．またこの形式により関連するパターンにより簡単に到達できると考える．

その他，パターンの項目として，タイトル，ID 番号，目的，用途，概要，外部仕様，内部仕様，関連するパターン☆☆，効果などを用意した．

パターンの例を**図 3** に示す．

詳細設計や実装に関するパターンには，理解の助けのためにサンプルコードを動作する完全形で提供した．

☆ 記述する項目として次が列挙されている：名前，別名，例，前提，課題，解決策，静的側面，動的側面，実装，補足，バリエーション，適用例，結論，参考．

☆☆ 同時に本文中で適宜関連パターンへハイパーリンクを張っている．

サンプルコードはそのまま流用することも可能である．またすべてのサンプルコードには UML などで記述した設計情報を用意しており，サンプルコードの利用プログラム言語（Java）とは異なるプログラム言語の再利用を助けている．

4. 開発作業

パターン体系は現場での経験をもとに内容をまとめるか，仮説をもとに適用範囲や内容をしぼり，実際に適用してフィードバックしなければ，机上の空論になる．我々の XML に関する経験は多くはなかったため，次の 3 ステップを行った．

(1) 知識の集約，第 0 版のパターン体系の作成，サンプル業務システムへの適用
(2) XML を用いる業務システム開発の支援，パターン体系の適用，実際に必要で一般性のある技術要件のパターン化
(3) パターン体系へのフィードバック

それぞれのステップの詳細を述べる．

4.1 第 0 版のパターン体系の作成

3.1 節で述べたように，我々のパターン体系の対象はアプリケーションシステムのサーバ側である．我々パターン開発メンバの多くは，今回の開発以前にサーバ側のアプリケーション開発支援を数多く経験し，また他のパターン体系の開発と試行も経験している[2)~4)]．その経験からパターン体系の開発に関して，仮説（3.2 節で示した 5 つの入口）を持って取り組んだ．

XML の基礎技術や応用方法についての知識は研究開発を行ってきた部署やすでに数多く出版されていた書籍や Internet 上の文献から得た．

これらの経験と知識をもとに，第 0 版のパターン体系を作り，それを実際に適用した．このパターン体系作成ステップでは次の 3 つの作業を行った．

(1) 知識の集約

XML 技術に関する文献と，開発メンバがこれまで開発支援してきた経験，特に DB に関する経験から，XML の処理方式や DB 格納法を組み合わせ，XML を用いるシステムのアーキテクチャをまとめた．同時に XML に関する技術要素をリストアップした．リストアップした技術要素からパターンとすべき内容を選び，パターン体系の入口（アーキテクチャ，テンプレートモデル，詳細ノウハウなど）とテーマごとに大項目番号を振り，個々のパターンごとに小項目番号を振った．

ここで付けた番号は変更を不可としており，この番号で別のパターンから参照される．

(2) パターンの作成

パターンを作成し，開発メンバ内レビューや第三者レビューを繰り返した．

(3) サンプル業務システムへの適用

具体的な業務を想定し，そのシステムの開発に作成したパターンを適用した．パターン開発者とは異なる第三者に開発してもらい，パターンテンプレートの不備，解説している技術要素に関する不備を洗い出した．

4.2 パターン体系の適用

第 0 版のパターン体系を web 上で公開し社内で自由に利用できる状況を作った．同時にパターンの開発に携わるメンバがいくつかのシステム開発プロジェクトの方式設計作業に参加し，技術情報の提供を行った．提供した内容は，XML の適用の可否や XML 処理方式を決定するための情報，開発手順，XML 処理方式や詳細設計のプロトタイプ，コンポーネントなどである．またドキュメントや XML ボキャブラリ案の提示なども行った．

4.3 パターン体系へのフィードバック

プロジェクトの方式設計作業に参加したことで，多くのフィードバックが得られた．

我々が提供したノウハウには，パターン化していなかったノウハウや，パターン化していたが内容に不足があったノウハウがあった．これらのノウハウについて，内容を検討しパターンの洗練や追加を行った．

またパターン体系の全体構成やパターンテンプレートの見直しが必要なフィードバックを受け，次の 4.3.1 項から 4.3.6 項の変更を検討した．

4.3.1 ダウンロード形式の変更

パターン体系全体を 1 つにまとめたダウンロードファイルを新たに公開した．

当初,「利用者はパターン体系をある程度オンラインでたどっていき，必要な部分だけダウンロードする」と判断して，小さな単位でパターンをひとかたまりにダウンロード単位としていた．たとえば詳細ノウハウに関するパターンをテーマ別にひとかたまりにしてダウンロード単位とし，アーキテクチャやテンプレートに関するパターンはそれぞれ 1 つのパターンをダウンロード単位としていた．

しかし実際は，複数に分かれたダウンロードファイルをすべてダウンロードし，手元に置いておくという使われ方が多いことが分かった．

4.3.2 性能に関する情報の追加

XML 処理方式ごとの性能に関するデータを求めら

れた．基本的な XML 処理性能測定値や性能改善ノウハウを追加した．

当初は処理性能測定値を載せなかった．これは次の理由による．処理性能測定値は，それぞれの開発プロジェクトが実運用かそれに近い環境（OS の種類，搭載メモリサイズ，CPU の種類と数，動作クロック，ネットワーク，ハードディスク性能など）で測定する必要があり，開発するシステムの要件ごとに測定すべき観点も異なる．また XML 処理方式単体の処理性能ではなく，実際のシステムに近いコンポーネントの組合せで処理性能を測定する必要がある．これらのさまざまな要因の組合せを測定するには工数がかかりすぎる．

開発プロジェクトごとに性能測定するのが最適，という当初からのスタンスを保っており，基本的な XML 処理方式を除き，性能に関する情報は加えていない．

4.3.3 サンプルコードのコンポーネント化

使用頻度の高い XML 処理を部品化しコンポーネントとして提供した．使用法のドキュメントや，サンプルコード，シーケンス図も追加した．

4.3.4 動作確認情報の付加

上記のコンポーネントについて動作確認がとれた環境の情報（ミドルウェアや XML プロセッサなどの種類やバージョン）を追加した．

4.3.5 パターンの分割

コンポーネントとして部品化した処理に関するパターンを 2 種類に分割した．1 つはコンポーネントを使うユーザの観点でノウハウを記述したパターン，もう 1 つはコンポーネントで行っている XML 処理内容についてのノウハウを記述したパターンである．

部品化したパターンのいくつかは,「処理が複雑でパターンの対象読者が誰なのか明確でない」という指摘を受けていた．この指摘への対応としてパターンを分割した．

4.3.6 ツール群の提供

実際の開発プロジェクトで使われていたツールが非常に便利だったため，許可を得て汎用化し公開した．

また開発プロジェクトからの要望が多かった，コンポーネントの設定を行う外部ファイル（XML）のエディタを開発した．

5. 開発体制

我々のパターン体系は XML を利用するアプリケーションの開発に関するノウハウを集めている．実用に耐えるパターン体系に成熟させるために実際の開発プロジェクトに参加して効率的に必要な情報（求められているノウハウ，提供しているパターンの技術情報の過不足など）を収集する必要がある．そのためには我々を含め次のチーム構成が必要だった．

5.1 パターン体系開発チーム

業務開発支援経験を持ち，第 0 版のパターン体系を作成．実際の開発プロジェクトに参加し必要な情報を得てパターン体系を成熟させていくチーム．我々はここに属する．

5.2 商談支援チーム

開発プロジェクトとパターン体系開発チームの橋渡しを行うチーム．開発プロジェクトからの開発支援要請を受け，必要に応じてパターン体系開発チームを開発プロジェクトに参加させる．

通常は開発支援要請を受けても，商談支援チームが持つ XML 関連の技術情報（パターン体系を含む）に基づき質問に回答したり，適切な開発部隊やパターン体系を紹介する．開発プロジェクトにパターン体系開発チームが必要だと判断する，またはパターン体系開発チームにとってプロジェクトへの参加が望ましいと判断すると，パターン体系開発チームを開発プロジェクトに紹介する．

5.3 コンテンツ展開チーム

パターン体系を含む，XML 関連の技術情報を全社に周知，展開していくチーム．技術情報の説明会やパターン体系を適用した業務テンプレートの開発を行う．

また商談支援チームと業務開発に関する情報を共有しており，これに基づいたパターン体系のレビューも行っている．

5.4 教育チーム

パターン体系など XML 関連の技術の教材開発と教育体系の整備を行うチーム．たとえば，さまざまなスキルレベルを持つ人に対応できる形で，パターン体系の e-Learning 化を行っている．

5.5 コア製品開発チーム

DOM，SAX など XML プロセッサや XML 関連の仕様など，XML を利用するアプリケーションの開発の基盤となる技術や情報を開発・調査しているチーム．

パターン体系開発チームが実際の開発プロジェクトに入っていて，XML 処理方式の性能比較の実データや XML の新しい仕様の詳細など基盤時術に関わる情報が必要なときこのチームからの支援が必要となる．

ここにあげたチームが互いをサポートし合うことで，それぞれの目的，パターン体系開発チームにとってはパターン体系開発を達成することができる．

XML はつねに新しい技術情報が出てくる状態であるため，それに追随するためにもつねにパターン体系の改訂が必要である．この開発体制がうまく機能した

ため，つらすぎない程度の負荷で十分な情報やフィードバックを収集することができ，ほぼ 3 カ月に 1 回のペースで改訂を行うことができた．

6. 評　　価

社内公開したパターン体系に対し過去 16 カ月で数千件のアクセスがあった．アクセスしたユーザに対してパターン体系の利用目的や適用状況などを調査した．調査は，アクセス時のアンケートによる所属や利用目的などの収集，時間をおいて行った確認アンケートによる利用方法や効果などの情報の収集，およびいくつかの代表的な適用プロジェクトへのヒアリングによって行った．

利用法で多いのは，提案書を含むドキュメント作成の参考，プロトタイプ構築，新技術の導入の参考，コンポーネントとしての利用，XML 技術の学習教材としての利用などである．また設計を採用し他の言語で実装した例もある．

現在 50 プロジェクトに適用済みであり，数値は述べられないが大きなコスト削減効果が確認できた．パターン体系を全面適用した実アプリケーションのテンプレートをコンテンツ展開チームが開発，社内公開した．こちらも再利用によるコスト削減効果をあげている．

また普及への別なアプローチとして，このパターン体系を元に e-Learning 教材を開発した．こちらは過去約 1 年間で約 1,000 人が利用し，社内の開発要員の新しい技術の獲得に貢献できた．

パターン体系化したことでさまざまな観点で利用者が情報へたどり着くことができるため，上記のとおりさまざまな形で利用され，コスト削減効果を押し上げたと考える．

7. 関 連 研 究

Buschmann らは「パターンは，それ単独に存在するものではない．パターン間には多くの依存関係が存在する．しかし，すべてのパターンを平板に列挙したログのようなリストでは，これらの多種多様な関係は反映できない．このような方法ではなく，パターン体系の中に 1 つずつパターンが織り込まれるように記述されるべきである」と述べている[5]．しかし Buschmann らの文献を含め，実際に「パターンが織り込まれたパターン体系」の開発例を見つけることは難しい．たとえば，Keller らはネットワーク管理インタフェースについてパターン体系を開発したと述べている[8]が，パターンのリストアップにとどまり，パターンを織り込んでパターン体系にするところまでは至っていない．これは文献中で彼らも認めている．これに対し我々は次の 3 点に注意してパターンを織り込むことを目指した．

(1) パターン体系や個々のパターンの使い方を開発方法論で示す．
(2) 5 つの入口（3.2 節参照）を設けさまざまな観点から必要なパターンにたどり着ける構成を作る．
(3) パターン間にハイパーリンクを張り関連するパターンを見逃さない構成を作る．

1999 年の APSEC で GoF の Johnson は「相当数のパターンが貯まってからでなければ，パターンコミュニティがパターンを体系化することは難しい」という意味の発言をしている[9]（3.3 章）．我々はパターン体系の対象範囲を絞ったことで，最初にパターン化すべきノウハウを列挙することができ，初めからパターン体系化を念頭にパターンを構築した．

8. 考　　察

8.1 パターン体系の構成

我々はパターン体系に 5 つの入口を設け，さまざまな観点から必要なパターンにたどり着ける構成とした．この構成の是非と過不足について考察する．

評価で述べたようにこの構成によってパターンの利用機会が増えており，多くの入口を設けた効果があったと考える．

アーキテクチャとテンプレートは上流の開発作業で参照する入口として用意する必要があった．方法論は新しい技術を入れた開発の作業体系を提示している．これは開発プロジェクトを円滑に混乱なく進めるために必要であった．また詳細ノウハウとコンポーネントに関するパターンは，XML に関する知識や利用経験を持つユーザがリファレンスマニュアルとして詳細なノウハウの確認に使う．このようなユーザのために詳細ノウハウやコンポーネントも入口として用意する必要があった．

ほかに必要な入口がないか判断するのは難しいが，少なくとも利用者からは新たな入口の要求は起こっていない．

8.2 開 発 体 制

各チームがさまざまな役割を分担したことで，純粋に開発（プロジェクトの方式設計作業への参加やプロジェクトへのヒアリングを含む）に時間を割けた．この結果，不足している情報やパターンの理解のしやすさなどのフィードバックを得て，それに対応する形でパターンの追加，洗練を繰り返すことができた．現在

までに 8 回のパターン全体の定期的な改版を繰り返している．

また特に商談支援チーム，コンテンツ展開チームからのフィードバックがなければ，パターン体系や各パターンを成熟させることはできなかった．

ただし，パターン体系開発の初期段階からしばらくの間は各チーム自身がパターン体系の理解とパターンの内容の理解を行う必要があった．この期間を短くすることは今後の課題である．解決策として，たとえばパターン体系開発の初期段階から参加する，などがある．

8.3 開発・展開に必要な要件

XML 技術のパターン体系を開発・展開し効果を確認できた理由として，我々のケースでは次の要因があると考える．

- 開発体制が整ったこと．
- 専任の担当者が就いたこと．
- トップダウンのプロジェクトに組み込めたこと．XML を普及させるという全社的な動きと連動できた．
- XML に特化したこと．そのため短期間で提供内容の充実を図れた．

8.4 今後の課題

我々が開発プロジェクトに入ったとき，またパターン体系を各地に説明して回ったとき，こちらの想定していなかった質問や意見をいくつか受けた．これらから，次の点を改善すべきと考える．

8.4.1 対象範囲外との連携方式の提示

クライアントサーバシステムのクライアント側で XML を利用する方式は，我々のパターン体系の対象範囲外である．しかし対象範囲外のシステムとの連携に関する質問を何度も受けた．

企業で利用されるパターン体系に特有の要件であろうが，パターン体系の範囲内と範囲外の境界部分についてもノウハウを提供する必要がある．

なお，我々のパターン体系に関しては，コンテンツ展開チームの開発したパターン体系適用例の中で，境界部分のノウハウを提供した．

8.4.2 企画作業（方向付け）

我々のパターン体系は分析作業からテスト作業までの方法論を提供する．しかし，お客様からいつから XML の使用を本気で検討すべきかとご質問を受けた，といった，我々の対象範囲外の質問への回答例を尋ねられた．

このような質問に答えられるところまでパターン体系の範囲を広げる必要はないと考える．仮に我々のパターン体系を含む知識共有体系を作ったときは，我々のパターン体系からのリンクを用意すべきだろう．

9. おわりに

新しいソフトウェア開発技術の発信の試みとして，新技術を利用するシステムの開発ノウハウのパターン体系化を行い，このパターン体系を社内へ展開した．具体的な技術として XML を取り上げ，XML を利用するシステムの開発ノウハウのパターン体系を開発し，実際に適用して一定の効果を得た．我々は，今後現れる新たな技術要素に対処する際にも 5 つの入口（アーキテクチャ，テンプレートモデル，詳細ノウハウ，コンポーネント，開発方法論）からなるパターン体系が有効であると考えている．機会を得て別のソフトウェア開発技術を取り込んだパターン体系を開発し，現在の構成を成熟させたい．

また XML に関連する仕様，技術は現在も進化，変化し続けている．これらに合わせて，パターン体系を引き続き改版していく予定である．

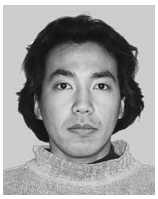
参考文献

1) 井上 健，桐嶋正光，林 康二：企業内オブジェクト指向相談室の設立と運用経験，オブジェクト指向最前線 2002，pp.27–34, ソフトウェア工学研究会 (2002).
2) 吉田裕之，山本里枝子，上原忠弘，田中達雄：UML によるオブジェクト指向開発実践ガイド，技術評論社 (1999).
3) 上原忠弘，山本里枝子，吉田裕之：パターン指向開発とパターン自動適用ツール，オブジェクト指向'98 シンポジウム，pp.29–36 (1998).
4) 山本里枝子，上原忠弘，大橋恭子，中山裕子，吉田裕之：ビジネスアプリケーションむけパターン体系とその適用，ソフトウェア工学研究会 99-SE-124 (1999).
5) Buschmann, F., Meunier, R., Rohnert, H. and Stal, M.: *Pattern-Oriented Software Architecture — A system of Patterns* (1996). 金澤典子ら（訳）：ソフトウェアアーキテクチャ ソフトウェア開発のためのパターン体系，トッパン (1999).
6) Gamma, E., Helm, R., Johnson, R. and Vlissides, J.: *Design Patterns Elements of Reusable Object-Oriented Software* (1995). 本位田真一ら（監訳）：オブジェクト指向における再利用のためのデザインパターン，ソフトバンク (1999).
7) Vlissides, J.: *Pattern Hatching Design Patterns Applied* (1998). 長瀬嘉秀ら（訳）：パターンハッチング，ピアソン・エデュケーション (1999).
8) Keller, R., Tessier, J. and Von Bochmann, G.:

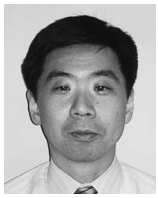
A pattern system for network management interfaces, CACM 41, Vol.9, pp.86–93 (1998).
9) 鈴木純一，長瀬嘉秀，田中　祐，松田亮一：ソフトウェアパターン再考，日科技連 (2000).
10) Le Hors, A., Le Hegaret, P., Wood, L., Nicol, G., Robie, J., Champion, M. and Byrne, S.: Document Object Model (DOM) Level 2 Core Specification (2000). http://www.w3.org/TR/DOM-Level-2-Core/
11) SAX. http://www.saxproject.org/
12) JDOM. http://www.jdom.org/
13) Fallside, D.C.: XML Schema Part 0: Primer (2001). http://www.w3.org/TR/xmlschema-0/
14) RELAX NG home page. http://relaxng.org/

(平成 15 年 10 月 14 日受付)
(平成 16 年 3 月 5 日採録)

山本　晃治（正会員）
2000 年東京工業大学情報理工学研究科計算工学専攻博士課程修了．同年（株）富士通研究所入社．ソフトウェア開発手法・開発環境等の研究開発に従事．IEEE CS，ACM，日本ソフトウェア科学会会員．博士（工学）．



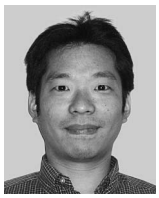
上原　忠弘（正会員）
1993 年東京工業大学工学部制御工学科卒業．1995 年同大学院総合理工学研究科知能科学専攻修士課程修了．同年（株）富士通研究所入社，現在に至る．オブジェクト指向開発技法，ソフトウェア再利用，ソフトウェア開発環境等の研究開発に従事．



松尾　昭彦（正会員）
1987 年東京理科大学物理学科卒業．同年（株）富士通研究所入社．ソフトウェアリエンジニアリング，テスト，開発環境等の研究開発に従事．

大橋　恭子（正会員）
1986 年津田塾大学学芸学部数学科卒業．同年富士通（株）に入社．論証支援システム，オブジェクト指向開発技法，テスト技法，ビジネスモデリング技法の研究開発に従事．

猪股　順二
1997 年東北大学工学部情報工学科卒業．同年富士通（株）入社．システムエンジニアとして通信業のシステム開発に従事．その後，現在までソフトウェア開発環境等の開発に従事．



松田　友隆
1993 年岡山理科大学電子工学部電子工学科卒業．1995 年同大学院電子工学専攻修士課程修了．同年（株）富士通入社．ソフトウェア開発環境等の開発に従事．

山本里枝子（正会員）
1983 年早稲田大学理工学部電子通信学科卒業．同年富士通（株）入社．オブジェクト指向開発技法，ソフトウェア再利用，ソフトウェア開発環境等の研究開発に従事．1999 年情報処理学会山下記念研究賞受賞．