ユーザ文書のオンライン生成
Online generation of user documentation

品川一貴† 平井航一‡ 山本喜一‡
Kazutaka Shinagawa† Koichi Hirai‡ Yoshikazu Yamamoto‡
†慶應義塾大学大学院 理工学研究科
†Graduate school of Science and Technology, Keio University

要旨
チュータ，ウィザード，ヘルプなどのアプリケーションのユーザ文書を，ユーザからの要求に応じてオンラインドキュメントとして動的に生成，表示するためのフレームワークを提案し，OpenOffice.org の Calc を例に実現したシステムを紹介する。ドキュメントの要素の記述を集めたファクトベースの定義と作成，ファクトベースに基づくドキュメント生成と表示方式を述べる。このアプローチを用いることによって，ユーザにとっては，自分のスキルレベルに応じて適切な分量のドキュメントを得ることができ，アプリケーションのドキュメント開発者にとっては，アプリケーション開発の初期段階からドキュメントの開発を同時並行的に行うことができるようになる。

1. はじめに

複雑な機能を持つアプリケーションが数多く使われるようになり，その一方でコンピュータのユーザは小学生から高齢者まで広がって，ユーザのスキルレベルがかつてないほど広がっている。多くのアプリケーションでは，従来の分厚い紙のユーザマニュアルに代わり，いわゆるオンラインドキュメントが多く使われるようになってきた。残念ながら現状では，紙の文書をそのままオンラインファイルとして利用したり，精々ハイパーテキストの機能を持たせたりしているに過ぎないものも多い。

われわれが提案するオンライン文書は，現在のほとんどのアプリケーションで使われている GUI ベースの作業に基づき，ユーザがアプリケーションを利用するときに，操作対象を指定し，機能を実行するための操作を行うというオブジェクト指向モデルを利用し，ヘルプ，チュータ，ウィザード，参照用文書などすべてのユーザ文書をオンライン文書として自動的に生成する。また，ヘルプ，チュータ，ウィザードについては，ユーザが自分のスキルレベルを選択することによって，ユーザに合わせた粒度のオンライン文書を動的に生成する。

ユーザにとっては，現在の固定的な文書に比べ自分のスキルレベルに合わせて動的に生成された文書を得られるという利点があるとともに，アプリケーション作成者，特にユーザ文書を作成しているテクニカルライタにとっては，同様な記述を何度も繰り返さずにすむだけでなく，アプリケーションリリース直前の集約的な作業を軽減し，アプリケーションの修正変更に伴う文書の変更にも柔軟に対応できる。

本研究は，現実のアプリケーションに応用することを目指し，OpenOffice.org の表計算アプリケーションである Calc を対象として実現している。

2. GUI アプリケーションにおける作業

ユーザが GUI アプリケーションを使って作業を行うとき，そのアプリケーションに用意されている機能を特定の対象に対して適用する。Calc においてたとえば，あるセル範囲をコピーし，別の場所に貼り付ける機能を実行するとき，ユーザの行う操作は次のようになる。

(1) 対象とするセル範囲を選択する。
(2) メニューバーの [編集] ボタンを押す。
(3) ドロップダウンリストから [コピー] を選択する。
(4) セルポインタを目的のセルに移動する。
(5) メニューバーの [編集] ボタンを押す。
(6) ドロップダウンリストから [貼り付け] を選択する。

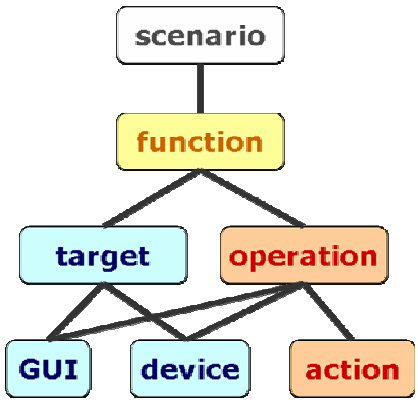

図 1　ファクトベース

これら一連の操作列によって一つの機能を表現することができ，それぞれの操作は，“[編集] ボタン”のような対象を“押す”という動作の組合せによって表現できる。実際には，上述の“セル範囲を選択
する”という操作は，
(1)　あるセルを選択する。
(2)　セル範囲の終わりまでマウスポインタをドラッグする。
という二つの操作に分割する必要がある。

GUI アプリケーションにおけるすべての作業を，“対象＋動作”という形式の並びとして表現することができるならば，テクニカルライタの立場からは上述の“セル範囲を選択する”操作を一度だけ詳細に定義しておけば，それ以降は必要に応じて詳細な操作列が生成できることになり，無駄な記述を省くことができる。一方，ヘルプを要求するユーザから見ると，“セル範囲の選択”の方法を知っているユーザにとっては，その詳細な操作法の表示はかえって邪魔になるので，自分のスキルレベルに応じた表示がなされるほうがよい。

3.　ファクトベース

ユーザに提示されるヘルプ，チュータ，ウィザードの対象をユーザの行う作業とし，その作業を説明する記述をシナリオと呼ぶ。シナリオは，テクニカルライタが記述するものであるが，基本的には 2.で述べたような操作列になる。シナリオの作成については，6.で述べる。

2.で述べたシナリオを表現するために，我々は**図 1** に示すファクトベースと呼ぶ構造を定義した。ファクトベースは，最も細かな粒度をもつ対象と動作を末端のノードとする階層構造として，アプリケーション作成者が定義するものとする。**図 1** のファクトベースの構成要素を次のとおりに定義した。

(1)　GUI 構成要素 (GUI)：ウィンドウ，ダイアログボックス，メニュー，ボタンなどアプリケーションの画面を構成する要素。
(2)　デバイス情報 (device)：キーボード，マウスなど入力機器を構成するキー，左ボタン，右ボタンなど。
(3)　基本動作 (action)：マウスボタンを“押す”，“放す”，“クリックする”，あるキーを“押す”などユーザが行うそれ以上分解できない動作。ダブルクリックは“クリックする”を 2 回連続する動作なので，基本動作としては定義しない。
(4)　対象 (target)：GUI 構成要素やデバイス情報では表現しきれない操作対象を定義する。たとえば，“複数のセル”や“セル範囲”は，GUI 構成要素のひとつであるセルの集合として定義する。
(5)　操作 (operation)：基本動作では表現できない動作で，複数の動作の組合せ，デバイス情報と基本動作の組，GUI 構成要素と基本動作の組となる。たとえば，“ドラッグする”という操作は，“マウスの左ボタンを押す”，“マウスを移動する”，“マウスの左ボタンを離す”によって表現できる。
(6)　アプリケーション機能 (function)：アプリケーションが提供する機能。メニューやツールアイコンによって表現されている機能に対応する。

これらのファクトベースのうち，デバイス情報として 2 ボタンマウス，3 ボタンマウス，106 キーボード，101 キーボード，基本操作としてそれぞれのデバイスに対応する動作を定義しておけば，新たなデバイスを使用しない限りこれらを変更する必要がない。また，アプリケーションによって追加・削除があるものの，GUI 構成要素の多くの定義もそのまま使用できる。

シナリオを含むファクトベースはすべて XML によって記述している。また，それぞれのファクトの XML 記述を形式的に定めるために，XML スキーマ記述言語である XML Schema を用いた。たとえば，アプリケーション機能のファクトに対する XML Schema は，図 2 のように模式化できる。それぞれのアプリケーション機能に対して機能 (function) 要素を定義し，それぞれの機能 id 属性によって区別し，内

容を記述する要素として，多言語対応のための言語要素を置き，その下に名前，説明，イメージ，動画の各要素を持たせている。

図 2 の XML Schema に従った“セル範囲の選択”機能の XML 記述は，図 3 のとおりである。図 3 に示すとおり，セル範囲を選択するとき，マウスをドラッグして選択する方法と，Shift キーを押しながらマウスの左ボタンをクリックする方法とがあるが，それらを<case>タグによって表現している。図 3 には言語要素が含まれていないが，言語要素は多言語対応のために含めた要素で，アプリケーション機能の記述自体には影響しないため，言語要素だけを別に記述することで機能記述の簡潔化を図っている。

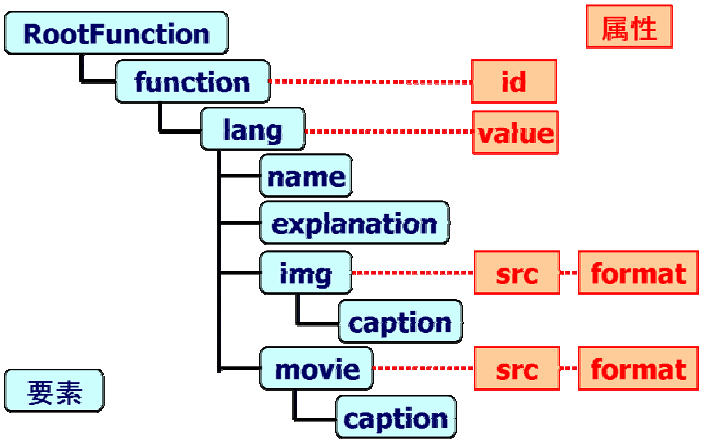

図 2　機能のファクトの要素

```
<?xml version="1.0" encoding="Shift_JIS"?>
<RootFunction>
    <function id="SelectCellRange">

        <case num="1">
            <step num="1">
                <component id="RootWindow_CellField_Cell" />
                <operation id="LeftClick" />
            </step>
            <step num="2">
                <device id="Mouse" />
                <operation id="Drag" />
            </step>
        </case>

        <case num="2">
            <step num="1">
                <component id="RootWindow_CellField_Cell" />
                <operation id="LeftClick" />
            </step>
            <step num="2">
                <device id="ShiftKey" />
                <action id="Pressing" />
            </step>
            <step num="3">
                <component id="RootWindow_CellField_Cell" />
                <operation id="LeftClick" />
            </step>
        </case>

    </function>
</RootFunction>
```

図 3　セル範囲選択機能の XML 記述

4. オンラインヘルプの動的生成

我々のシステムの現在の実現では，図 4 に示すとおり Apache Tomcat をサーバとして利用し，JSP (Java Servlet Pages)を使ってユーザからの要求によってヘルプを動的に生成している。この方式は，一つの実現例として示したものであり，これが唯一の実現方法ではない。すなわち，ここで提案しているのは，**図 1** に示したファクトベースの構造を用いることによって，実現方法によらず柔軟なオンラインドキュメンテーション環境が構築できることである。

ヘルプ画面は，XSLT によって表示形式を与え，シナリオとファクトベースから生成した XML を HTML に変換し，Web ブラウザに表示する。現在は，ユーザのスキルレベルを，初心者，中級者，上級者の 3 段階に分け，ユーザが申告した自分のレベルに応じた詳細さでヘルプを表示する。この場合には，ユーザのスキルレベルに対応する内容だけが生成される。たとえば，上級者ならば最上位レベルの箇条書きの文章だけを生成し，初心者の場合には詳細なステップで図を含めた文書を生成する。

また，別の XSLT によって Windows ヘルプのような目次見出し形式での表示もできるが，この場合にはすべての内容を含む文書を生成し，それを見出しに応じて折り畳むことになる。

さらに，前述した多言語化機能によって，ファクトベースの記述に対応する言語記述をするだけで容易に別の言語のドキュメントを生成できることを確認している。

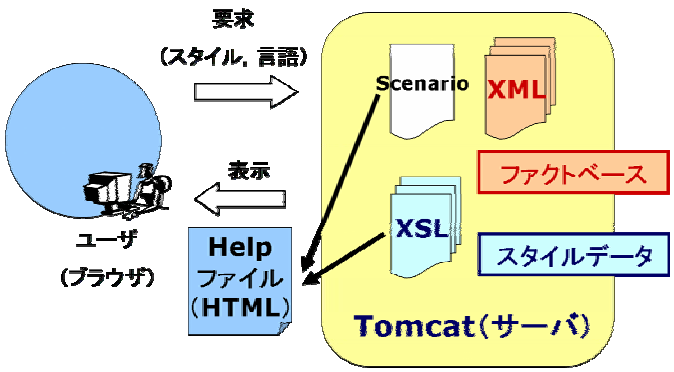

図 4　オンラインヘルプの自動生成

```
<?xml version="1.0" encoding="Shift_JIS"?>
<RootScenario lang="ja">
  <scenario id="SelectingThemesforSheets">
    <title>セル範囲を移動する</title>
    <step>
      <function id="SelectCellRange" />
      <function id="Execute_RootWindow_Menu_Edit_Copy" />
      <function id="SelectCell" />
      <function id="Execute_RootWindow_Menu_Edit_Paste" />
    </step>
  </scenario>
</RootScenario>
```

図 5　生成したシナリオ

5. ファクトベースの作成

ここで提案した枠組みを実際のアプリケーション開発に利用するには，ファクトベースを効率的に作成する必要がある。XML Schema でファクトベースの形式が規定されているので，市販のシステムを使って XML 記述を人力で作ることはできるが，その労力はかなり大きくなってしまう。

ひとつの解決方法として，GUI ベースのアプリケーションの場合，GUI 構成要素が木構造で表現できることを利用し，GUI 要素を Excel の表で表現し，そこから GUI 要素のファクトベースを自動的に生成するプログラムを作成した。これによって，多言語対応を含め GUI 要素についてはアプリケーション開発過程の初期段階からドキュメント作成に取り掛かっても，GUI 要素の変更修正があったときに直ちに簡単にファクトベースを変更修正できる。

6. シナリオの作成

本システムでは，ユーザに実際に表示されるドキュメントは，テクニカルライタが記述するシナリオに基づいて動的に作成することになる。テクニカルライタのシナリオ作成が簡単でなければ，結局は従来のドキュメント作成と同じことになってしまう。そこで，我々は実際にアプリケーションを使用して，シナリオに従った操作を行い，そのときのイベントを獲得して自動的にシナリオの骨組みを生成するシステムを作成している。

作業の説明に相当する文章はテクニカルライタが入力する必要があり，説明のための図版も作成する必要があるが，少なくとも操作手順に関しては，システムがアプリケーション機能の系列として XML 記述を生成するので，テクニカルライタの労力を大幅に削減できる。

Calc におけるシナリオ作成の例として，“セル範囲の複写” を行って生成したシナリオを図 5 に示す。

7. まとめ

ここで提案した枠組みは，特定のアプリケーションを対象としたものではなく，さらに特定の実現方法に限られるものでもない。図 1 のファクトベースの構造を用いることによって，ユーザドキュメントを動的に生成でき，ユーザ，アプリケーション開発者双方にとって利点が大きいことを示した。

現実のアプリケーションに対する適用例として，OpenOffice.org の Calc を使ったが，いわゆる Web サービスアプリケーションに対しても簡単に適用できるなど，きわめて広い応用が期待できる。また，ユーザタスク認識と統合することによって，高性能なインテリジェントヘルプを生成できる可能性がある。

参考文献

[1] P. N. Sukaviriya and J. D. Foley, “Supporting adaptive interfaces in a knowledge-based user interface environment,” Proc. of the 1993 International Workshop on Intelligent User Interfaces, ACM Press, 1993-10, pp. 107-113.