ラベル付きリストと XML 関連

1. 概要

MZ Platform では、ラベル付きリストと呼ぶ独自のデータ構造とそれを扱うコンポーネント群を提供しています。この文書では、ラベル付きリストとそれを扱う各種のコンポーネントの使用方法について説明します。

1.1. ラベル付きリストとは

ラベル付きリスト(PFLabeledObjectList)とは、MZ Platform の基本データ型の一つであるリスト(PFObjectList)の派生データ型で、リストの要素に対して名前でアクセスできるように拡張されたものです。通常のリストは各要素に対してインデックス(0 から始まる番号)を指定してアクセス(値を取得/設定)しますが、ラベル付きリストでは各要素に名前を付けられるので、その名前を指定して各要素にアクセスすることができます。

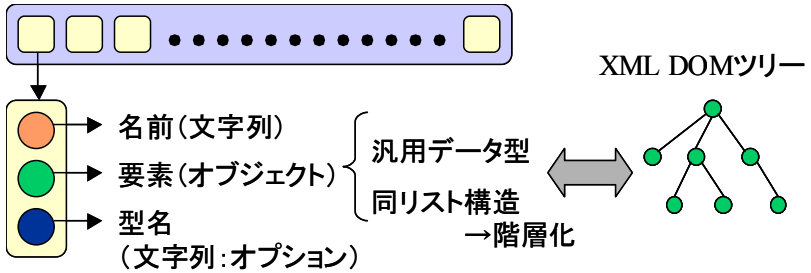


図 1　ラベル付きリストの概要

1.2. ラベル付きリストの役割

ラベル付きリストの要素には、リストの場合と同様に任意のオブジェクトを与えることができます。そのため、ラベル付きリストの要素としてラベル付きリストのオブジェクトを与えると、ラベル付きリストで階層化されたデータが実現できます。このようなデータではツリー構造のような複雑なデータが表現でき、その複雑なデータ内の各要素に対して名前(パス)を指定してアクセスすることが可能です。ラベル付きリストと呼ばれるデータ型の役割は、この

ような複雑なデータ構造をソースコードを書かずに自由に設定できるようにすることです。また、このようなツリー型の構造は XML 形式への変換が容易であり、MZ Platform ではラベル付きリストデータと XML の変換機能を標準で用意しています(後述)。

1.3. ラベル付きリストと XML に関するコンポーネント

この文書で説明するコンポーネントのリストとそれらの機能は次の通りです。

- ラベル付きリスト格納変数：　　ラベル付きリストデータに対してアクセスや操作をおこなうための機能
- ラベル付きリスト実体：　　ラベル付きリストデータの実体を管理してシリアライズ･デシリアライズを行う機能
- ラベル付きリストツリー変換：　　ラベル付きリストデータからツリーデータに変換する機能
- ラベル付きリスト用文字列検索：　　ラベル付きリストデータの中を走査して任意の文字列を検索する機能
- XML 変換：　　XML ファイルから DOM ツリーを作成して XSLT で他形式に変換する機能
- ラベル付きリスト XML 変換：　　上記の XML 変換に DOM ツリーとラベル付きリストの相互変換機能を付加したもの

また、これらをビルダーで追加する際のメニューは次の通りです。

- ラベル付きリスト格納変数：　　[コンポーネント追加]-[処理部品]-[変数]-[ラベル付きリスト格納変数]
- ラベル付きリスト実体：　　[コンポーネント追加]-[処理部品]-[オブジェクト]-[ラベル付きリスト実体]
- ラベル付きリストツリー変換：　　[コンポーネント追加]-[処理部品]-[ユーティリティ]-[ラベル付きリストツリー変換]
- ラベル付きリスト用文字列検索：　　[コンポーネント追加]-[処理部品]-[ユーティリティ]-[ラベル付きリスト用文字列検索]
- XML 変換：　　[コンポーネント追加]-[入出力]-[ファイル]-[XML 変換]
- ラベル付きリスト XML 変換：　　[コンポーネント追加]-[入出力]-[ファイル]-[ラベル付きリスト XML 変換]

2. 用途

この文書で説明するコンポーネント群の用途として以下のようなものが挙げられます。

- 自由なデータ型を作成して XML 形式の入出力や文字列検索を実現する場合に用いる

3. ここで使用するイベントとメソッド

各コンポーネントに関して、この文書内で使用されるイベントとメソッドの一覧を記します。ここに記す以外にもイベントやメソッドがありますが、それらの情報が必要な場合は各コンポーネントのリファレンスやJavadocドキュメントを参照してください。

- ラベル付きリスト格納変数
  - データ設定イベント

イベント発生条件	内包データ	イベント番号
データ設定のメソッド呼び出し	ラベル付きリストデータ	0

  - データ更新イベント

イベント発生条件	内包データ	イベント番号
データ内の要素を更新するメソッド呼び出し	ラベル付きリストデータ	0

  - メソッド一覧

メソッド名	機能
リストを設定する(PFLabeledObjectList)	引数で指定したラベル付きリストデータを格納する
全要素を削除する()	格納するラベル付きリストデータ内の要素を全て削除する
最後尾に他集合の要素を全て追加する(Collection)	ラベル付きリストに限らず集合データの要素を変換してリストの最後尾に追加する
リストを取得する()	格納するラベル付きリストデータを戻り値として返す
空のリストを設定する()	空のラベル付きリストを作成して変数に格納する
リストの名前を設定する(String)	引数で指定したテキストをラベル付きリストの名前として登録する
リストの名前を取得する()	登録してあるラベル付きリストの名前を戻り値として返す
要素の名前を指定して選択する(イベント発生なし)(String)	引数で指定したテキストを名前として持つ要素を選択状態にする
選択中の要素内のデータが取得可能かどうか判定する	選択状態にした要素のデータを取得可能かどうかの論理値を戻り値として返す
要素内のデータを名前指定で設定する(String, Object)	第1引数で指定した名前の要素が既にあれば第2引数で指定したデータに置き換え、まだなければ新たに要素を追加する

- ラベル付きリスト実体
  - イベント：なし
  - メソッド一覧

メソッド名	機能
ファイル名を指定してオブジェクトをロードする(String)	引数で指定した名前のファイルからラベル付きリストデータを読み込む
ファイル名を指定してオブジェクトを保存する(String)	引数で指定した名前のファイルにラベル付きリストデータを書き出す

- ラベル付きリストツリー変換
  - データ生成イベント

イベント発生条件	内包データ	イベント番号
ラベル付きリストデータの設定とツリー変換の実行	生成されたツリーデータ	0

  - メソッド一覧

メソッド名	機能
名前指定でリストデータを設定しツリーに変換する(String, PFObjectList)	引数で指定したリストをラベル付きリストに変換してからツリーデータに変換する

- ラベル付きリスト用文字列検索
  - イベント：なし
  - メソッド一覧

メソッド名	機能
リストデータ内を正規表現文字列で検索する(String, PFObjectList)	第1引数で指定したテキストが第2引数で指定したリスト内に存在するか検索する
検索にヒットしたデータリストを取得する()	検索に使用したリストの中で、ヒットした要素を集めたリストを作成して返す
検索ヒット結果のリストを取得する()	検索にヒットしたデータのパスを集めたリストを返す

- ラベル付きリストXML変換
  - データ設定イベント

イベント発生条件	内包データ	イベント番号
XML ファイルをロードして DOM ツリーが生成されたとき	生成された DOM ツリー	0

- メソッド一覧

メソッド名	機能
XML ファイルをロードして DOM ツリーを作成する(String)	引数で指定した名前の XML ファイルをロードして DOM ツリーを作成する
ラベル付きリストデータを XML ファイルに保存する (String, String, PFObjectList)	第 1 引数で指定した名前のファイルに、第 2 引数で指定した名前の XML データとして、第 3 引数で指定したラベル付きリストを保存する

4. コンポーネント使用例

4.1. サンプルアプリケーションの概要と使い方

ラベル付きリストと XML 関連のサンプルアプリケーションは”AP_DATA¥Sample¥ラベル付きリストと XML 関連.mzax”にあります。図 2 にサンプルの実行画面を示します。このサンプルでは、複数の異なる型のデータをまとめた一つのデータを、ユーザが定義したデータ型としてアプリケーション内で扱い、そのデータを使って実現できる便利な機能が用意されています。このサンプルを用いて、ラベル付きリストや XML に関連するコンポーネントの使い方を説明します。

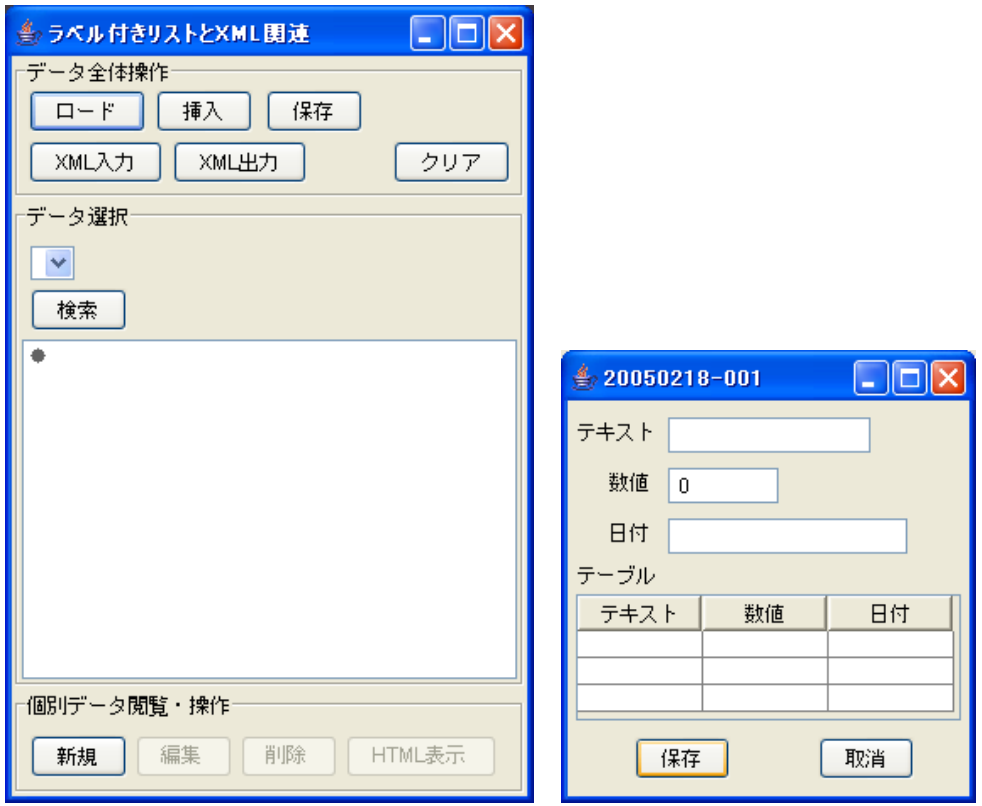


図 2　ラベル付きリストと XML 関連サンプルの実行画面

まず、基本的なデータ入力の方法について説明します。サンプルアプリケーションを開始すると図 3 に示すメインウィンドウが開きます。このアプリケーションはユーザが定義したデータを複数管理して、その中から選択されたデータの編集や閲覧を行うことができます。最初はデータがありませんので、一番下の「個別データ閲覧・操作」領域にある「新規」ボタンを押してデータを新たに作成します。

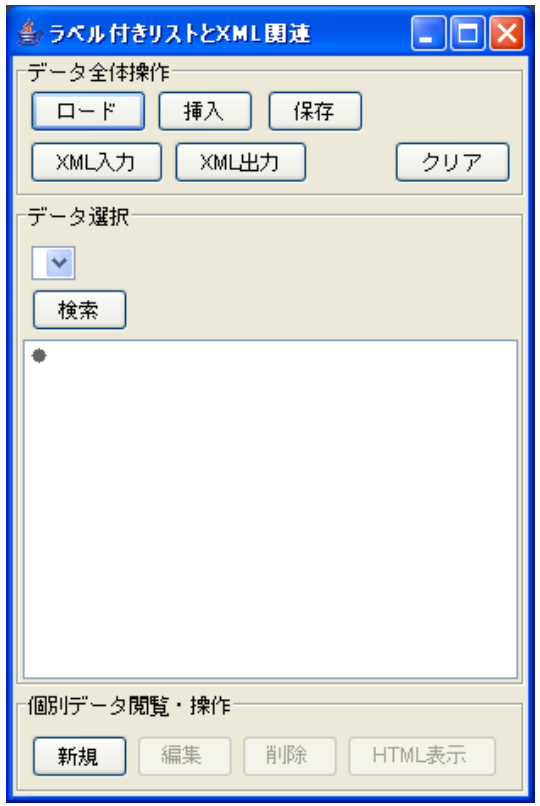


図 3　サンプルのメインウィンドウ

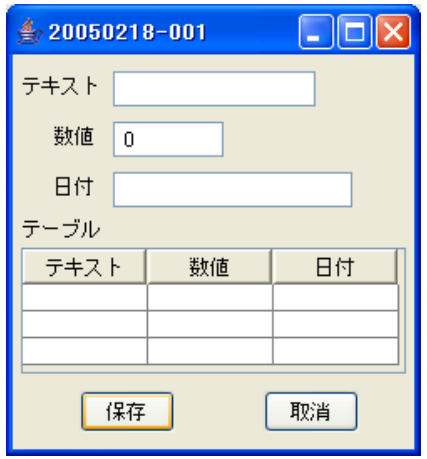


図 4　サンプルの編集ウィンドウ

「新規」ボタンを押すと、図 4 に示す編集ウィンドウが表示されます。このサンプルでは、テキスト・数値・日付・テーブルの組み合わせで構成されるデータを作成して、各データには固有の ID が割り振られる仕様になっています。固有の ID は文字列で、日付を表す文字列と 3 桁のシリアルナンバーの組み合わせ（YYYYMMDD-###）で表現されます。それぞれの値を入力したのち、「保存」ボタンを押すと入力したデータが登録されます。登録されたデータはメインウィンドウの「データ選択」領域のコンボボックス内にリストアップされ、コンボボックスの一覧の中から登録されたデータを選択すると、そのデータがツリービューに表示されます。データが選択された状態になると「個別データ閲覧・操作」領域の他のボタンが有効化され、選択したデータの編集や削除をすることができます。

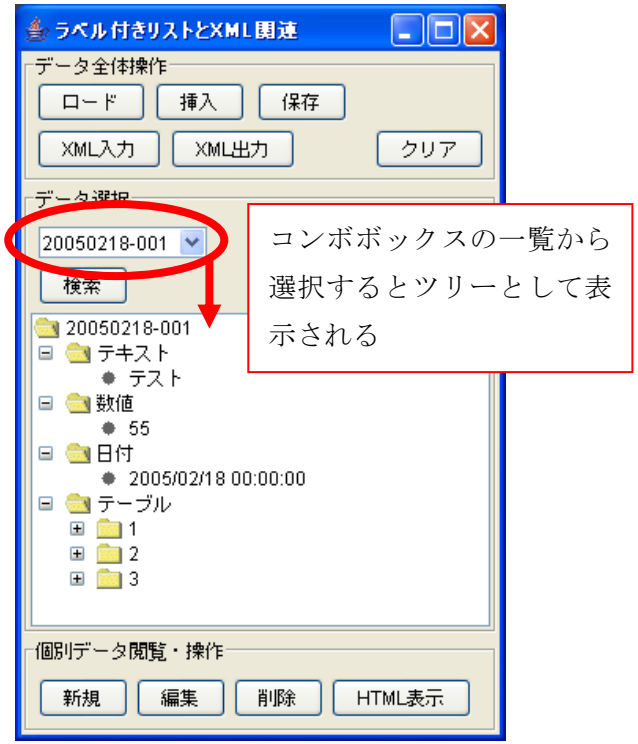


図 5　データの選択とツリー表示

新規作成を繰り返すと、コンボボックスから選択可能なデータがいくつも作成できます。これらのデータは全体をまとめて外部ファイルに出力したり、読み込んだりすることができます。「データ全体操作」領域の「保存」ボタンは、作成したデータのすべてをバイナリ形式の『シリアライズデータ』(*.ser)として保存します。適当な場所にシリアライズデータとして保存してみてください。また、「ロード」ボタンを押すことで、保存されたシリアライズデータを読み込むことができます。一度「クリア」ボタンを押してデータを全て消去してから「ロード」ボタンで保存したファイルを読み込んで、データが元に戻ることを確認してみてください。

ここまでのアプリケーションの動作と、ラベル付きリストの関係を簡単に説明します。このアプリケーションでは、扱うデータ全体を一つのラベル付きリストとして管理しています。最初はリストが空の状態で開始して、「新規」ボタンから作成したデータを保存したとき、リストに要素が一つ追加されます。この追加された要素も、ラベル付きリストとして定義してあります。入力ウィンドウに配置されたテキスト・数値・日付・テーブルは、要素としてのラベル付きリストの要素となっています。

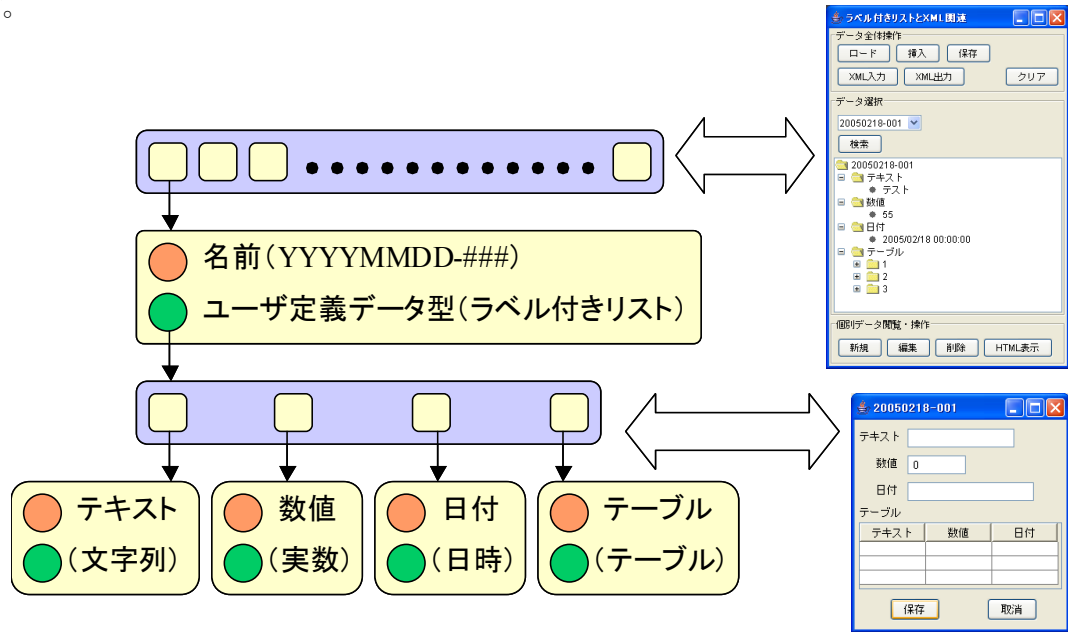


図 6　サンプルでのラベル付きリストの使い方

4.2. 変数コンポーネントと実体コンポーネント

サンプルアプリケーションで扱うようなデータの管理や操作には、変数コンポーネントと実体コンポーネントが重要な役割を果たします。ここではこれらに関する概要と、サンプルアプリケーション内での使用方法を説明します。

4.2.1. 変数コンポーネントの役割

MZ Platform には変数コンポーネントと呼ばれる一群のコンポーネントが提供されています。これは、MZ Platform 上で扱われるデータを一時的に格納して、そのデータにアクセスするためのコンポーネントです。

例えば、GUI のテーブルからテーブルデータを取得して、GUI のグラフにそのテーブルデータを設定するなど、MZ Platform 上ではデータのやりとりがよく行われます。このようなデータ(オブジェクト)は、実際には自分自身に定義された処理(テーブルなら行や列の追加や削除・値の変更等)を持っていますので、そのような処理を実行したい場合には、変数コンポーネントにデータを設定することで処理を呼び出すことができます。ラベル付きリストのデータを扱う場合にも、ラベル付きリスト格納変数にデータを設定して、名前を指定した要素へのアクセスなどの処理を実行することができます。

4.2.2. 実体コンポーネントの役割

変数コンポーネントがデータを一時的に格納して処理を呼び出すためのものであるのに対して、実体コンポーネントは中に入れたデータの保存とロードを可能とします。現在は、データ型を特に限定しないオブジェクト実体と、ラベル付きリストに限定したラベル付きリスト実体の２つが提供されています。データ型を限定しないオブジェクト実体には、任意の型のデータを設定できますが、設定したデータの型はアプリケーション側で把握する必要があります。

実体コンポーネントと変数コンポーネントの役割を明確にするため、実体コンポーネントで実行できる処理はデータの保存とロードに限定し、データの持つ各処理は変数コンポーネントに格納したときのみ実行できるように定義しています。

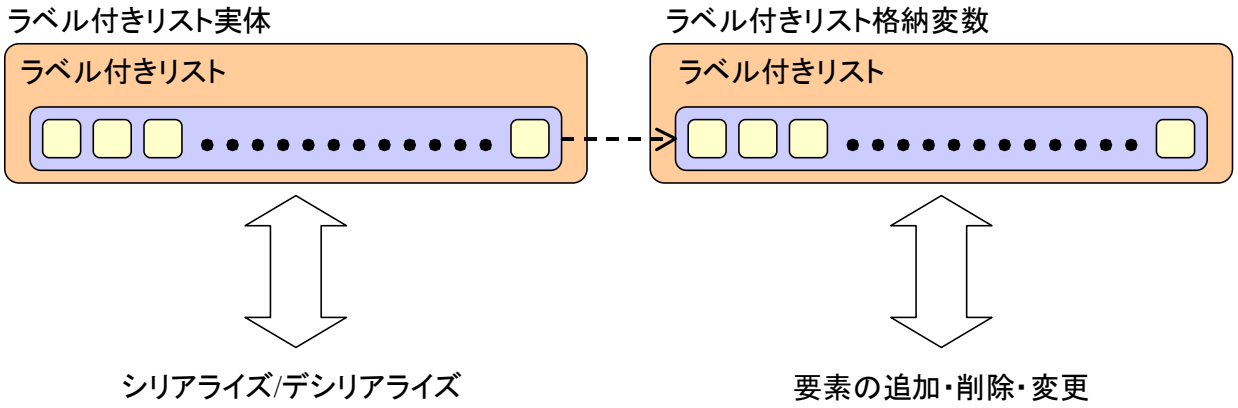


図 7　変数コンポーネントと実体コンポーネントの役割

4.2.3. サンプルアプリケーションにおけるラベル付きリストの実体/変数コンポーネント

サンプルアプリケーションにおいても、内部で扱うデータ全体を対象としたラベル付きリスト実体と、そのデータに対して操作を行うためのラベル付きリスト格納変数が定義されています。実体コンポーネントに設定しておいたデータはアプリケーションデータと一緒に保存され、アプリケーションデータをロードしたときにも同様に使用可能です。そして、データに対する処理を行うためには、実体コンポーネント内のラベル付きリストを一度変数コンポーネントに格納する必要があります。下図は、サンプルアプリケーションの開始時に呼ばれる開始サブルーチンで、複合コンポーネントの「データ管理コンポーネント」のメソッド「初期化処理を呼び出す」が呼び出されているところを示しています。

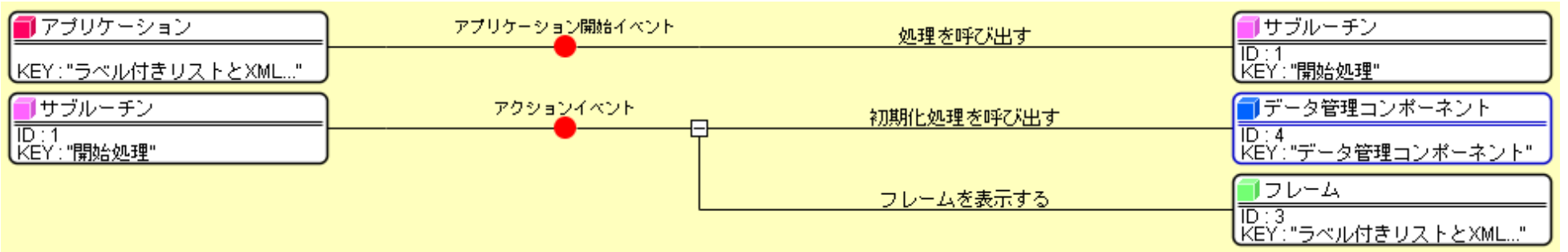


図 8　サンプルアプリケーションの開始処理

この「データ管理コンポーネント」の中に、サンプルアプリケーション内で扱うデータ全体を対象としたラベル付きリスト実体と、ラベル付きリスト格納変数が配置されています。アプリケーション開始時のメソッド呼び出しによって、この複合コンポーネントに定義された初期化サブルーチンが実行されて、ラベル付きリスト格納変数に対して実体コンポーネントに含まれるデータが設定されます。

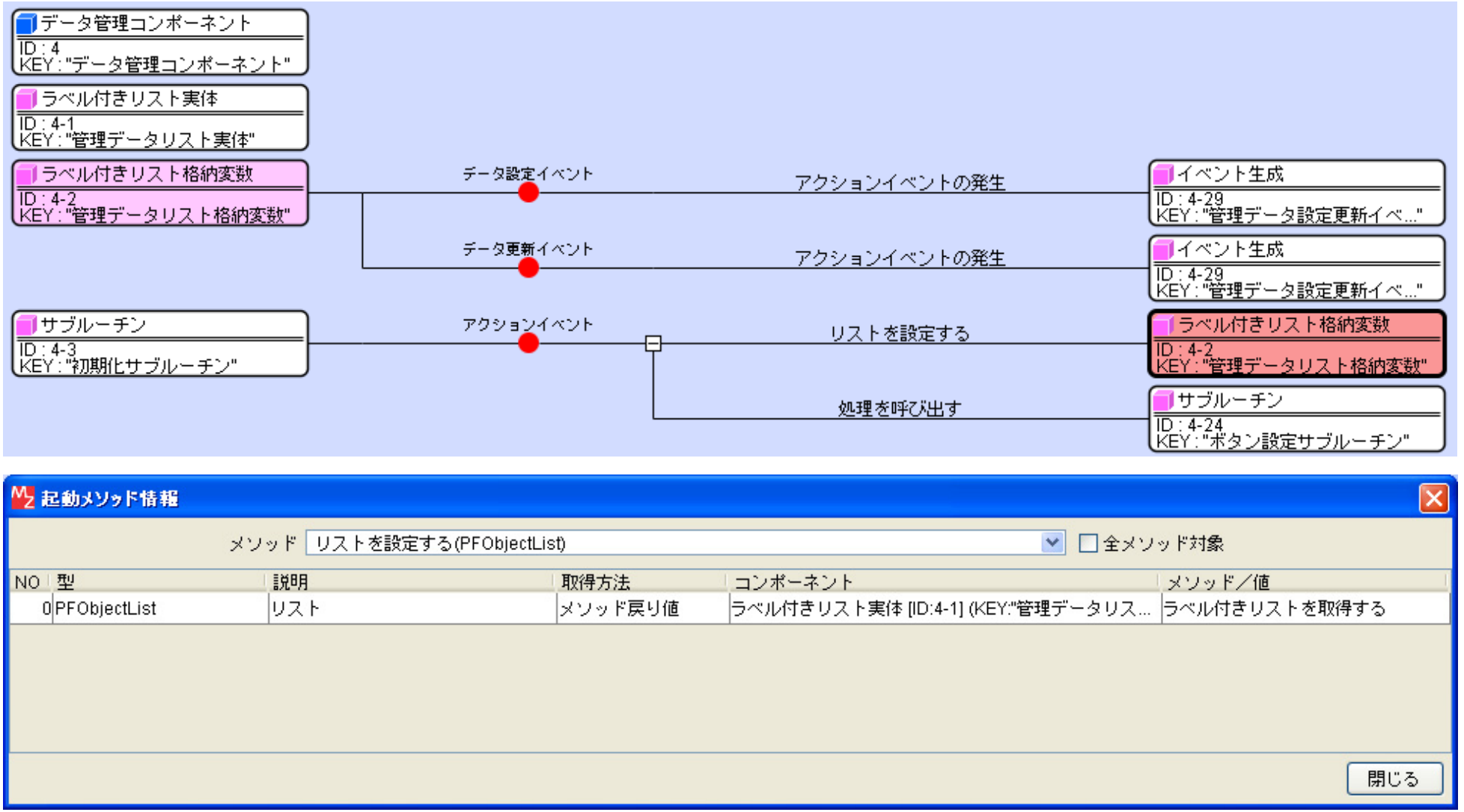


図 9　データ管理コンポーネント内の初期化処理

アプリケーションで扱うデータに対する操作は、変数コンポーネントを介して実行されます。このサンプルでは、ラベル付き格納変数のメソッドをデータ管理コンポーネントの公開メソッドにして、上位の階層でデータの編集を行っています。

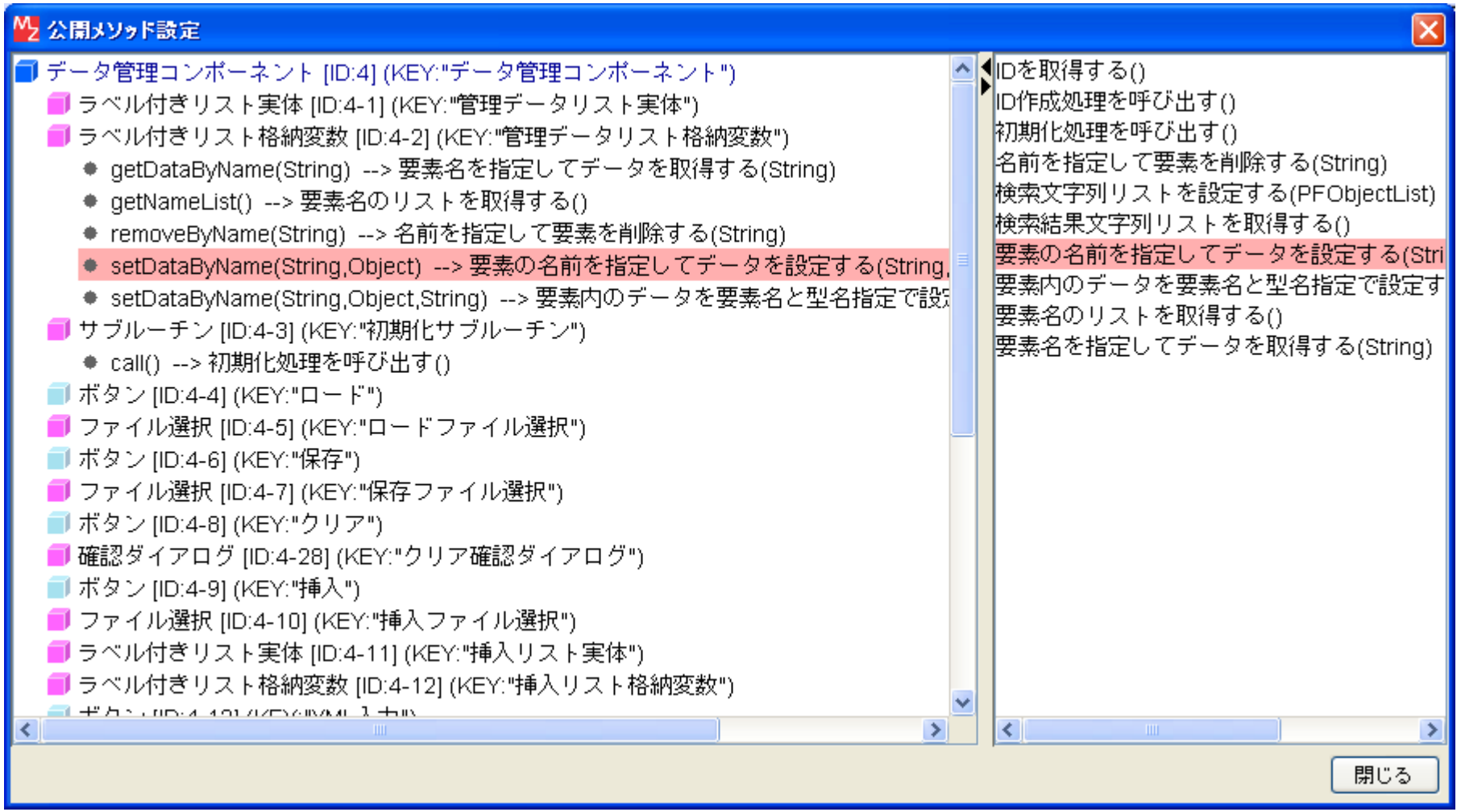


図 10　変数コンポーネントのメソッド公開

以下に示す図は、データ管理コンポーネントで公開したメソッドを、上位の階層で呼び出してデータの操作を行っている例です。これは編集ウィンドウで「保存」ボタンを押したとき、編集データを全体データに登録するときに実行されます。

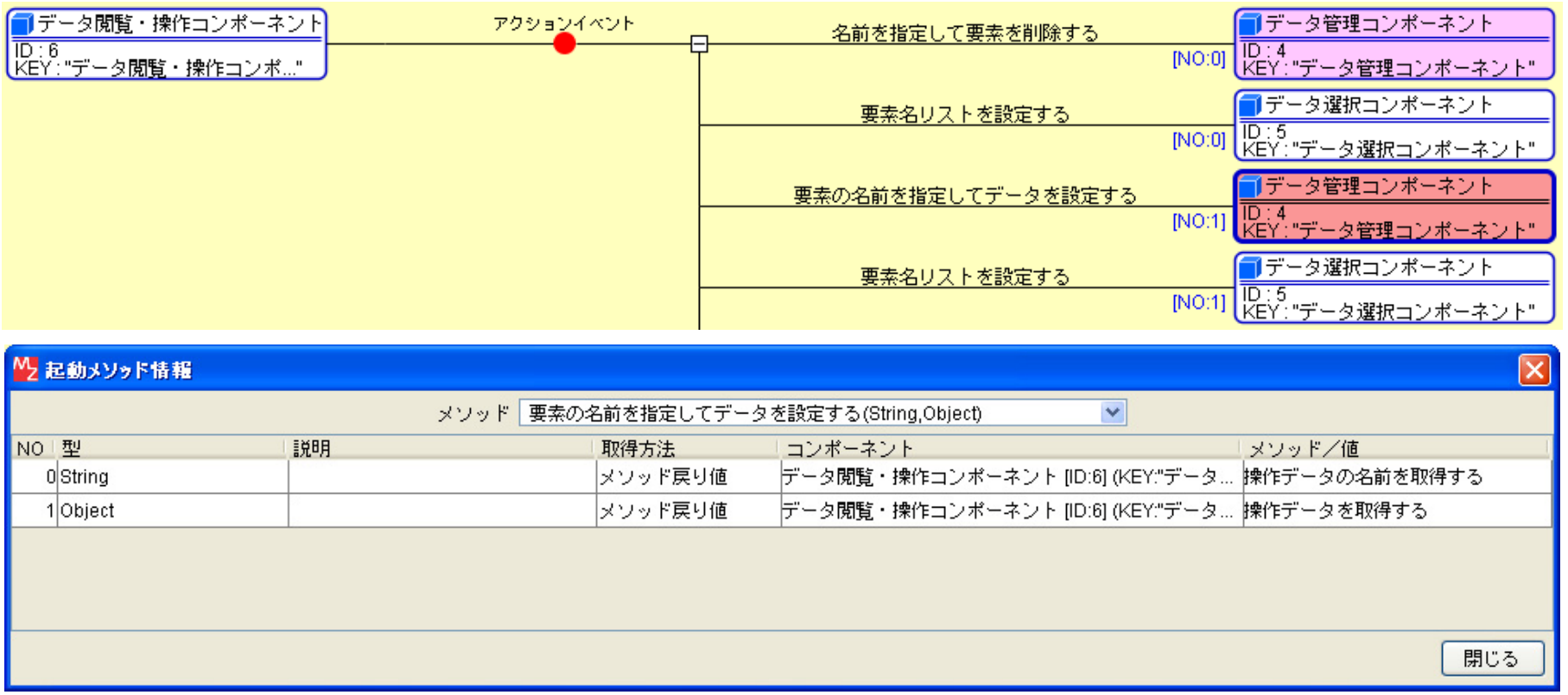


図 11　公開メソッドの上位からの呼び出し

さらに、実体コンポーネントはそれに含まれるデータだけを別ファイル（シリアライズデータ）として保存・ロードする機能があり、サンプルアプリケーション内で使うデータを個別に管理できます。以下の図は、「ロード」ボタンと「保存」ボタンを押したときの処理の呼び出し方を示しています。

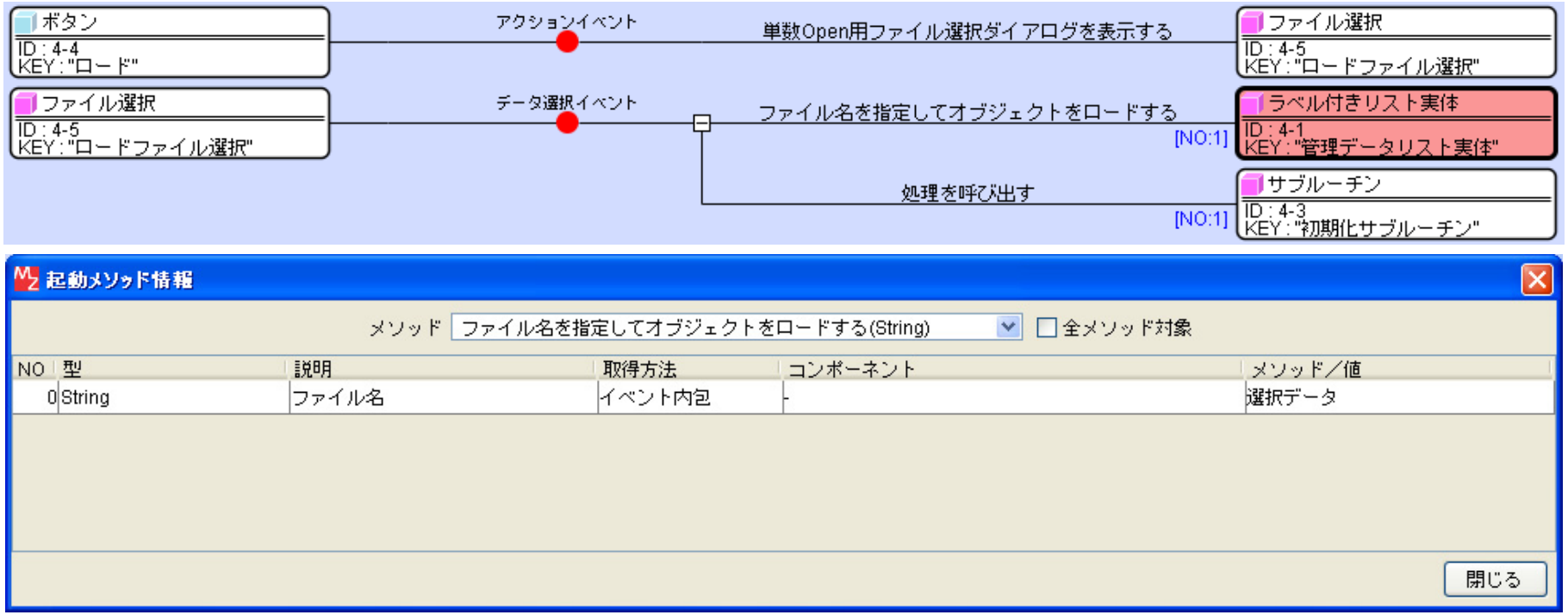
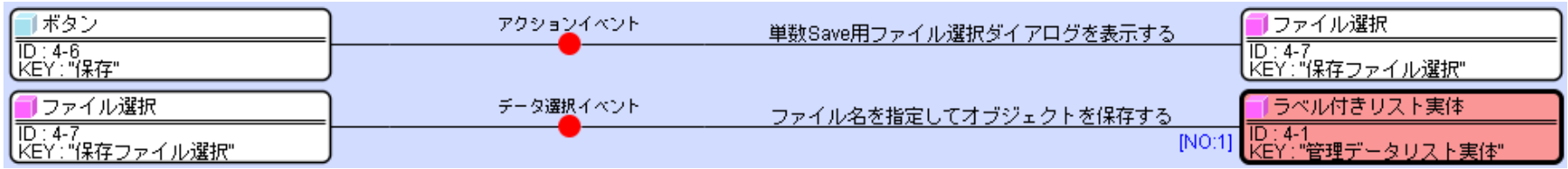

ボタン
ID : 4-4
KEY : "ロード"
アクションイベント
単数Open用ファイル選択ダイアログを表示する
ファイル選択
ID : 4-5
KEY : "ロードファイル選択"
ファイル選択
ID : 4-5
KEY : "ロードファイル選択"
データ選択イベント
ファイル名を指定してオブジェクトをロードする
[NO:1]
ラベル付きリスト実体
ID : 4-1
KEY : "管理データリスト実体"
処理を呼び出す
[NO:1]
サブルーチン
ID : 4-3
KEY : "初期化サブルーチン"
起動メソッド情報
メソッド
ファイル名を指定してオブジェクトをロードする(String)
全メソッド対象
NO
型
説明
取得方法
コンポーネント
メソッド／値
0
String
ファイル名
イベント内包
-
選択データ
閉じる

図 12　ロード時の処理


ボタン
ID : 4-6
KEY : "保存"
アクションイベント
単数Save用ファイル選択ダイアログを表示する
ファイル選択
ID : 4-7
KEY : "保存ファイル選択"
ファイル選択
ID : 4-7
KEY : "保存ファイル選択"
データ選択イベント
ファイル名を指定してオブジェクトを保存する
[NO:1]
ラベル付きリスト実体
ID : 4-1
KEY : "管理データリスト実体"

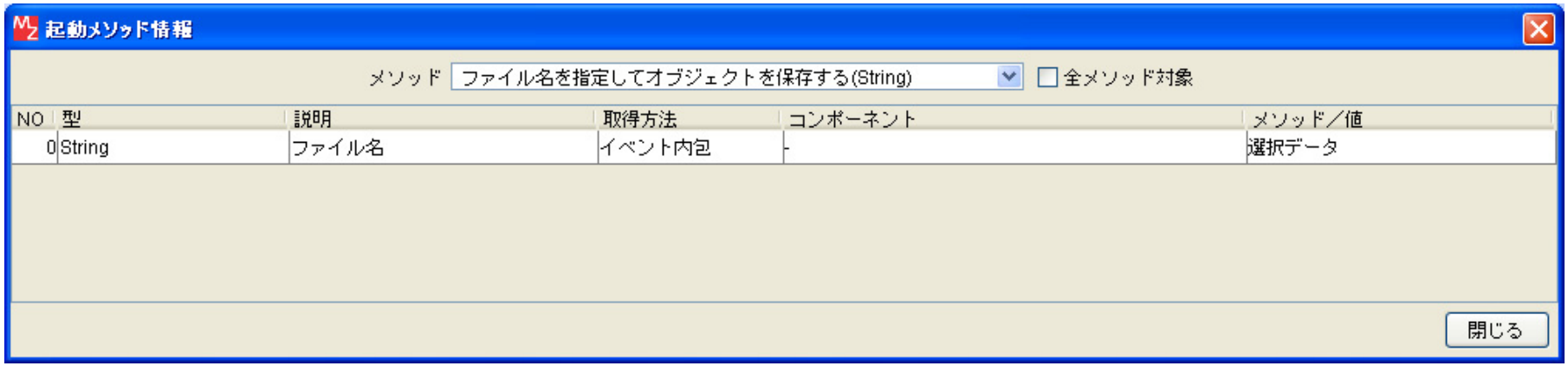


図 13　保存時の処理

4.3. ラベル付きリストを用いたユーザ定義データ型の作成

サンプルアプリケーションでは、データ全体をラベル付きリストとして管理している上に、個々の要素となるデータもラベル付きリストで定義しています。ここでは、個々の要素となるラベル付きリストに対する操作について説明します。

サンプルでは、「新規」や「編集」ボタンを押すと、編集ウィンドウが開きますが、これは「データ閲覧・操作コンポーネント」内にある複合コンポーネント「データ入力コンポーネント」の中で定義されています。「データ入力コンポーネント」内には、編集対象となるラベル付きリストデータを操作するための変数コンポーネントがあります。

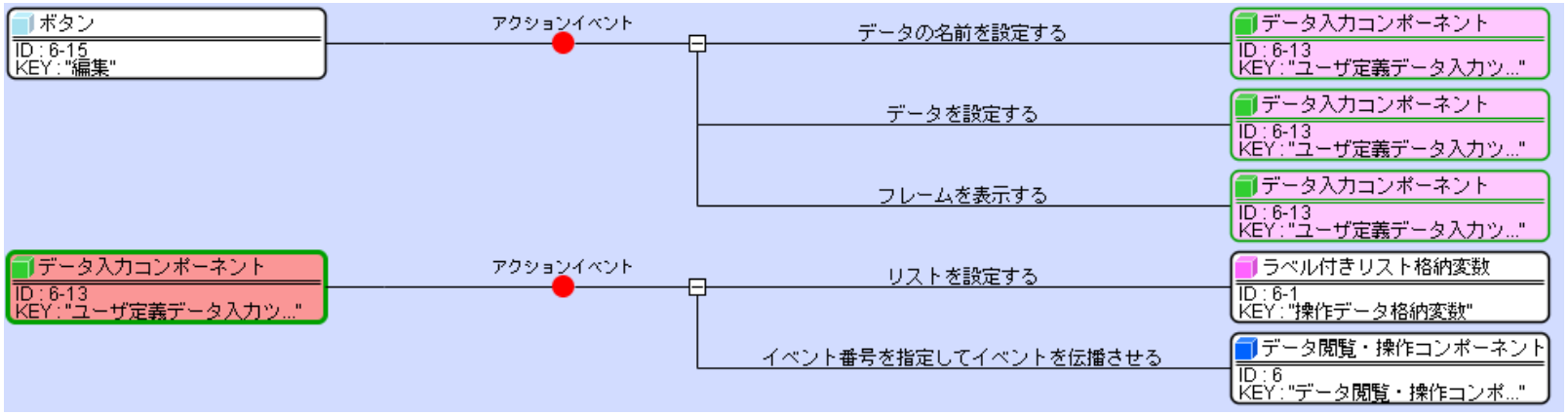


図 14　データ閲覧・操作コンポーネント内のデータ入力コンポーネントの呼び出し

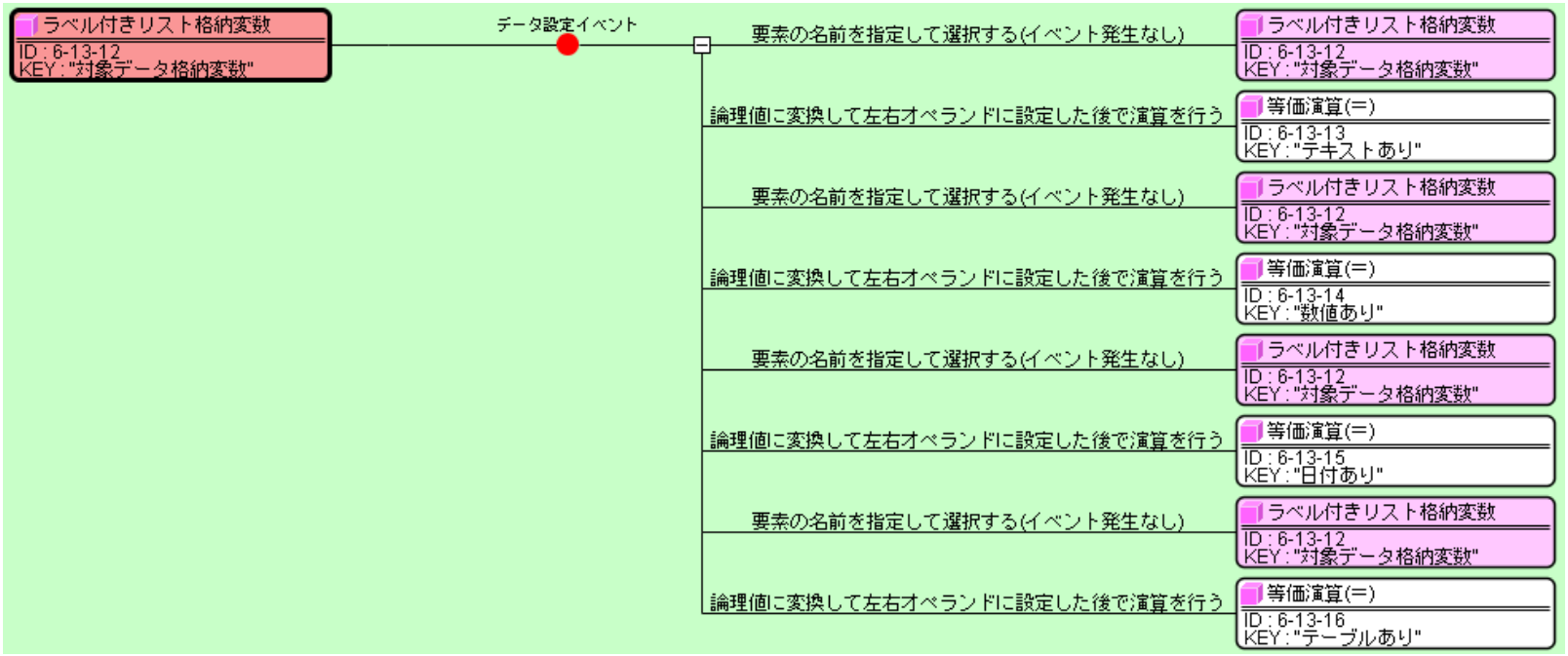


図 15　データ入力コンポーネント内のデータ設定イベント処理

編集を開始すると上位の階層からこの変数にデータが設定され、データ設定イベントが発生します。このイベント処理で、対象とするユーザ定義データ型の要素に対応するデータがリスト内にあるかどうか判定しています。

このサンプルで扱うデータには 4 種類の要素がありますが、すべて以下の作業の繰り返しとなります。まず、対象となるリストに対して「要素の名前を指定して選択する（イベント発生なし）」を呼び出します。このときの引数は対象とする要素の名前で、ここではウィンドウ上に表示するラベルからテキストを取得しています。

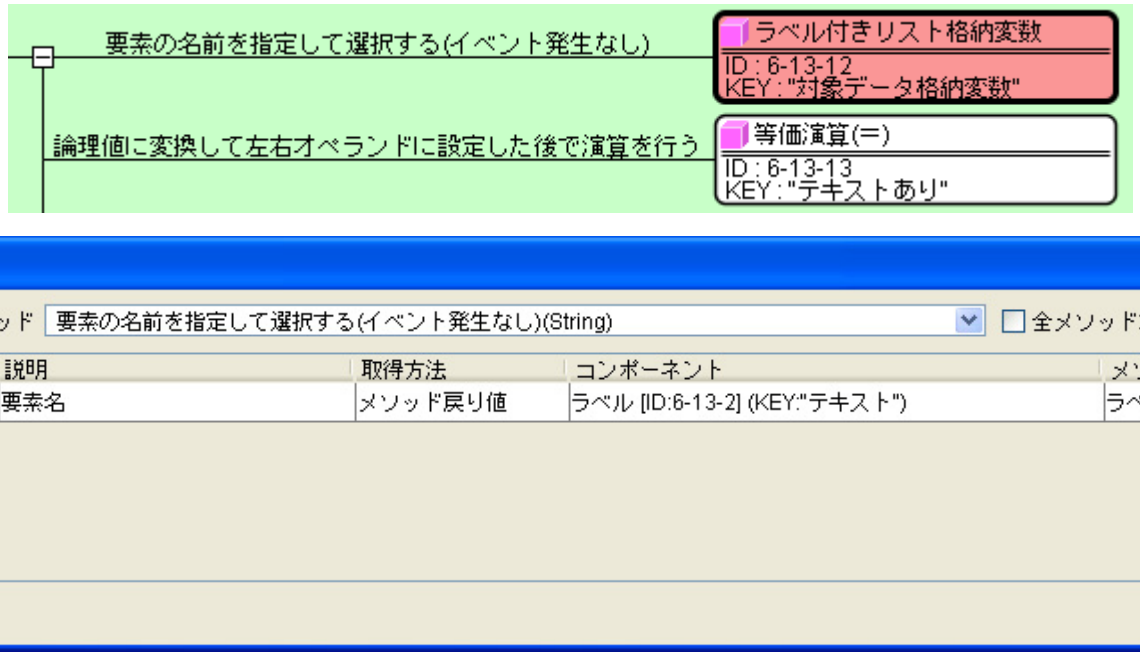


図 16　リスト内の要素の選択

続いて、各要素に対して定義した等価演算コンポーネントを使って、上で選択した要素が取得可能かどうか判定します。

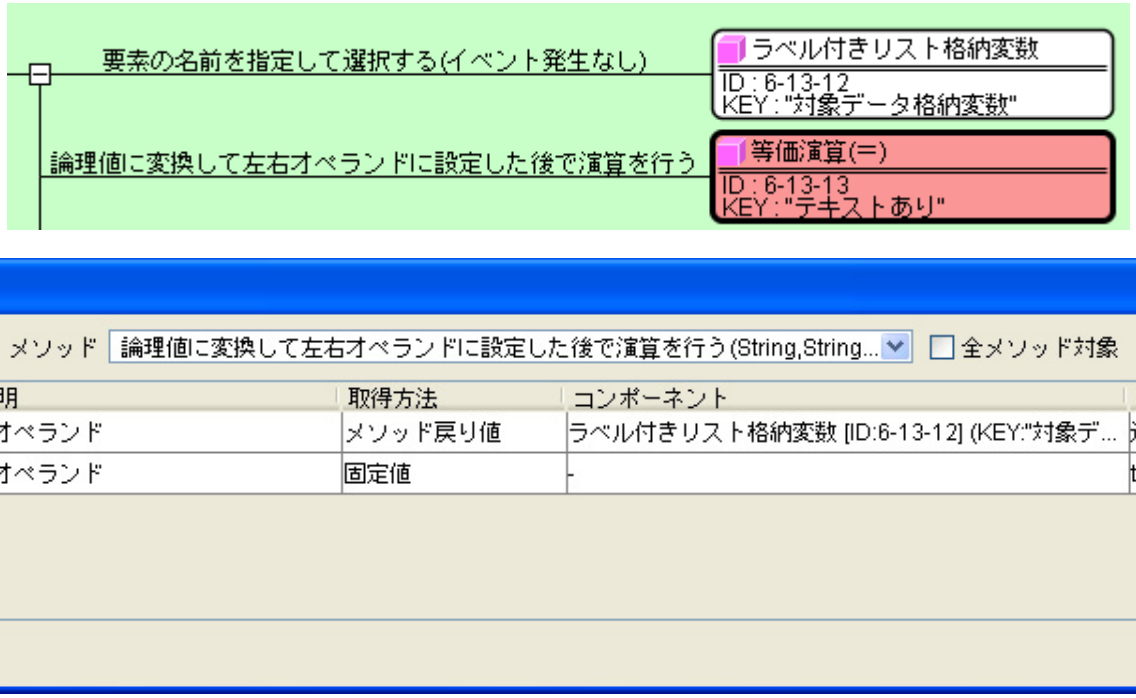


図 17　選択要素が取得可能かどうか判定

演算を実行すると等価演算コンポーネントから処理完了イベントが発生するので、取得可能な場合にはリスト内で選択中の要素のデータを取得して、データを編集するためのコンポーネント(ここではテキストフィールド)に設定します。取得可能でない場合にはデフォルト値を設定しておきます。

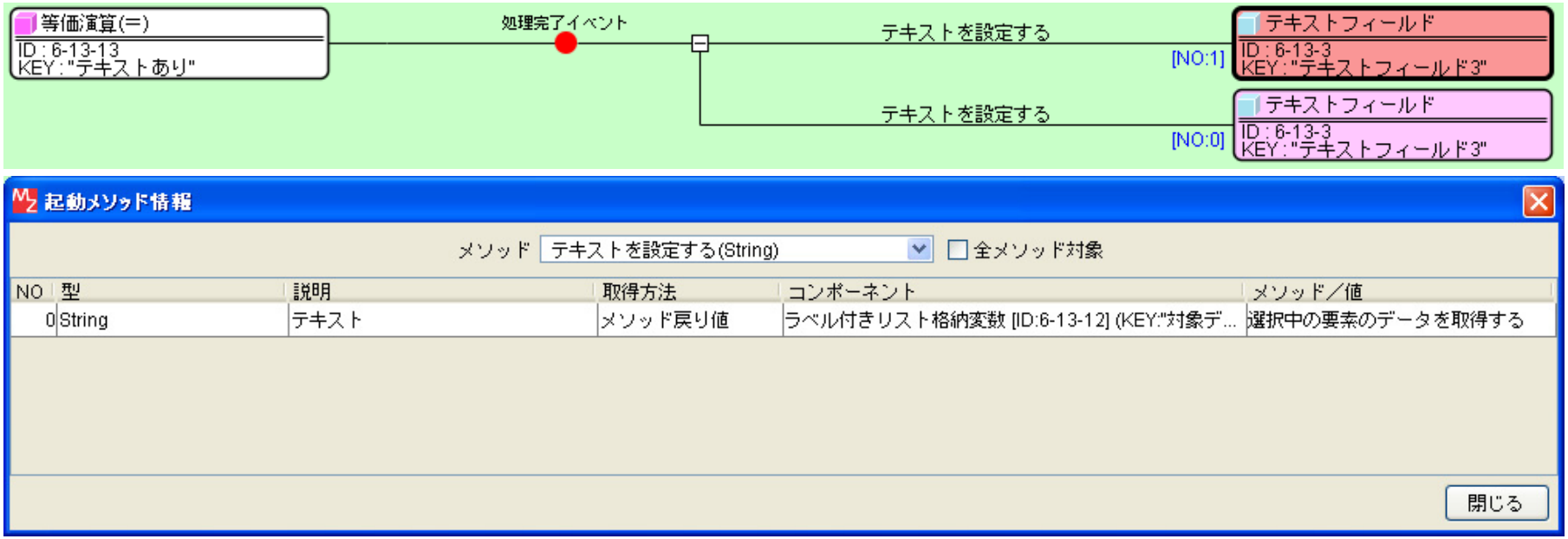


図 18　テキストフィールドへのデータ設定

設定された値に対して編集ウィンドウで入力を行ったあと、「保存」ボタンを押すと編集したデータを全体データに登録するという処理を行います。「保存」ボタンを押すまでは編集用の画面構成部品（テキストフィールド等）にデータがあるだけで、実際のラベル付きリストには反映されていないので、まずこれを行います。4 種類の要素に対して以下の操作を繰り返して行っています。対象とするリストが格納された変数に対して、「要素内のデータを名前指定で設定する」を呼び出して、引数としてラベルの文字列と入力の画面構成部品からデータを取得します。この操作によって、もともと同じ名前でデータが存在する場合にはデータを上書きし、その名前のデータが存在しない場合には新たに要素を作成します。

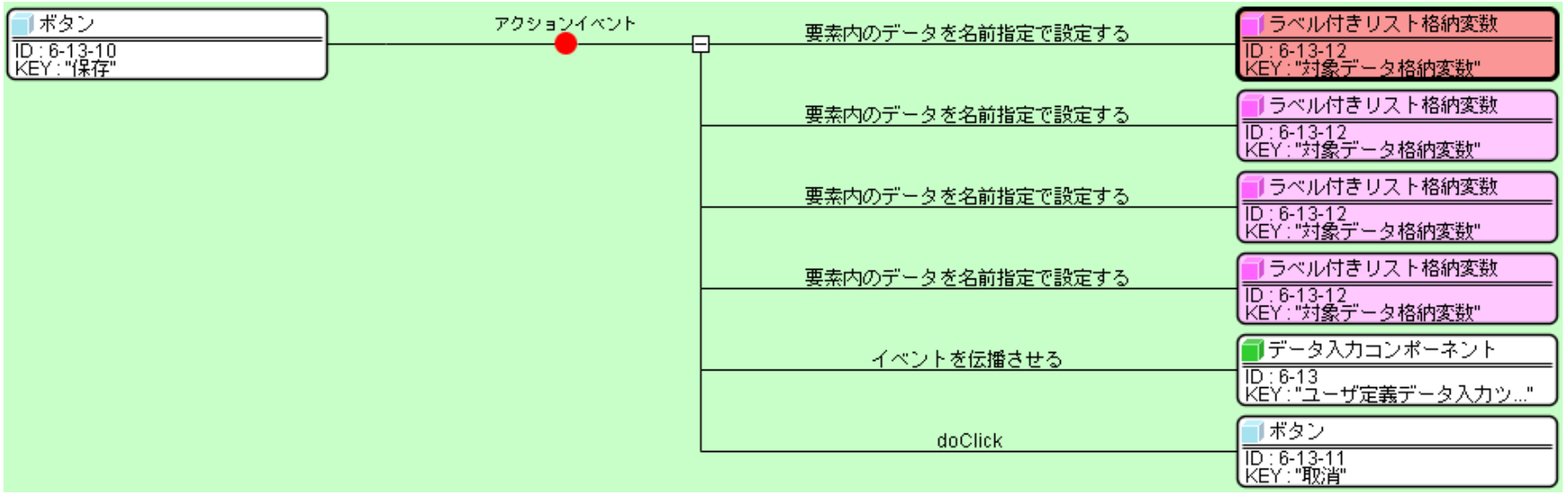


図 19　編集データの保存

このサンプルではすべての要素をリストに設定した後上位にイベントを伝播させています。これは、上位の階層からこのデータを全体のリストに加える操作を行うための通知です。

このサンプルアプリケーションの中で、ここで述べた「データ入力コンポーネント」の内容を修正するだけで、容易に独自のデータ型が定義できます。

4.4. ラベル付きリストを用いた処理の紹介

ラベル付きリストは複雑なデータを自由に定義するために導入され、これを有効利用するための処理がいくつか準備されています。代表的なものは、ツリーデータへの変換処理とラベル付きリスト内の文字列検索処理です。

4.4.1. ツリー変換

ラベル付きリストを階層的に作成することでツリー構造のデータを作成できることは既に述べました。実際にツリー構造になるかどうかはアプリケーション側に依存しますが、ツリー構造になっていればツリーデータとして画面構成部品のツリーに表示することができます。これを行うためにはラベル付きリストツリー変換を利用します。ラベル付きリストツリー変換にラベル付きリストデータを設定すると、データ生成イベントが発生し、そのイベントにはツリーデータが内包されるので、そのツリーデータを画面構成部品のツリーに設定すると、ラベル付きリストの内容をツリーとして表示することができます。

サンプルアプリケーションでツリー変換の使用方法を見ることができます。サンプルでは、コンボボックスのリストからデータを選択すると、そのデータがツリービューに表示されます。これは、個々の選択データがラベル付きリストになっていて、選択データが設定されるたびにツリーに変換して表示しているためです。この処理は複合コンポーネント「データ選択コンポーネント」の中で行われ、その中にある「選択データ格納変数」にデータが設定されるとラベル付きリストツリー変換にデータを渡して、生成されたツリーデータを画面構成部品のツリーに設定しています。

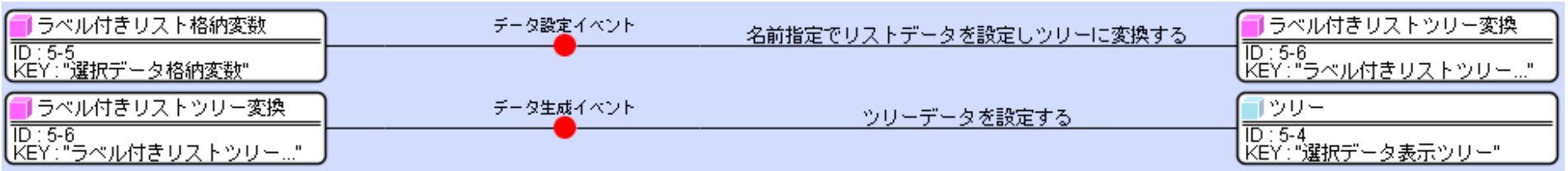


図 20　ラベル付きリストツリー変換の使用法

4.4.2. 文字列検索

次に、ラベル付きリスト内の文字列検索処理について述べます。これはラベル付きリスト用文字列検索の機能で、ラベル付きリストに含まれるすべての要素内を探索して、指定した文字列と一致する文字列を含む要素のリストを取得します。

以下の図はサンプルアプリケーションでの検索例を示しています。まずはサンプルの編集ウィンドウで適当なデータを入力してみてください。続いて、「データ選択」領域にある「検索」ボタンを押すと検索用ウィンドウが表示され、「検索文字列」フィールドに文字列を入力して「全文検索」ボタンを押すと、検索結果が下のリストに表示されます。検索して見つかった場合はその文字列が存在するデータの位置が表示され、リストから項目を選択すると対応するデータのツリーが表示されます。ツリー内の対応する項目を表示してみると、確かにその文字列が含まれていることがわかります。

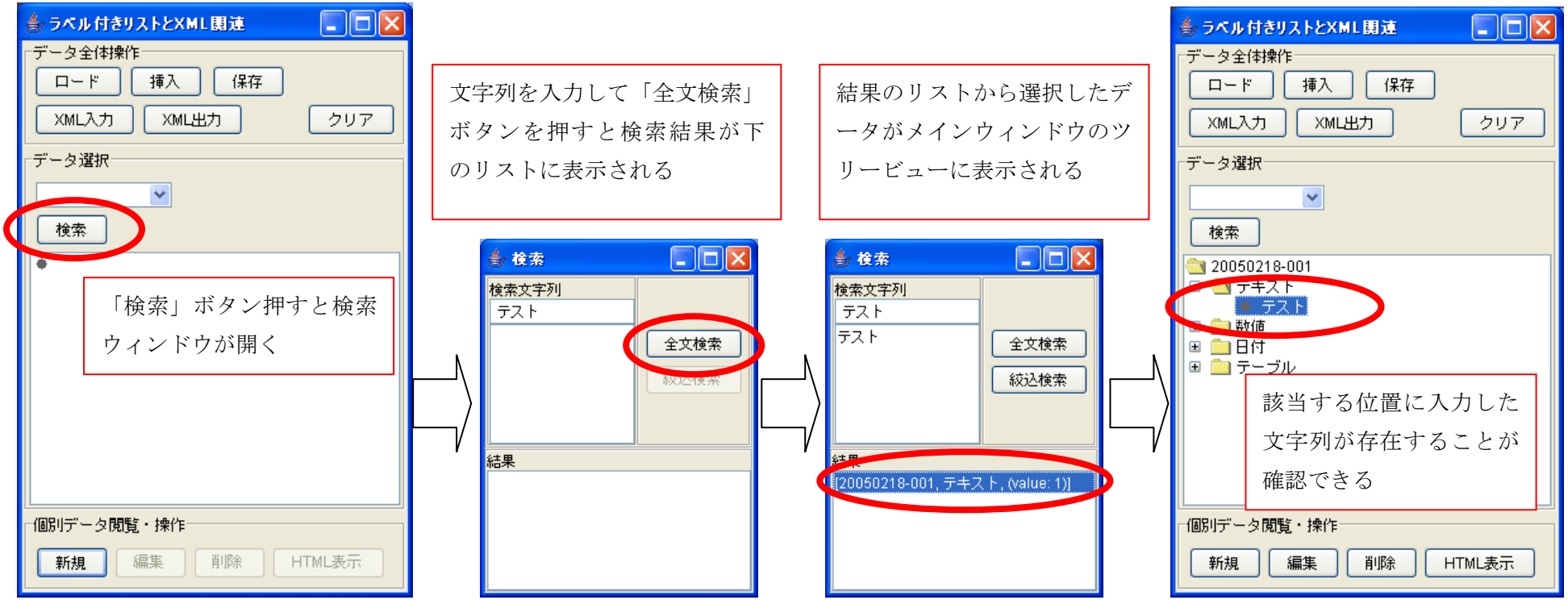


図 21　文字列検索の概要

4.5. XML 処理

4.5.1. XML 変換コンポーネント

MZ Platform 上で XML データを扱うコンポーネントとしては、XML 変換コンポーネントが挙げられます。これは、XML ファイルと XSL スタイルファイルを読み込んで、任意の形式で出力できる XSLT の機能を実装したコンポーネントです。内部的には XML ファイルを読み込んで DOM ツリーを作成し、XSL スタイルファイルに従った形式でファイル出力するという処理を行っています。ただし、このコンポーネントでは DOM ツリーをプログラム側から利用するための処理が定義されていません。

4.5.2. ラベル付きリストの XML 入出力(外部データ交換)

XML ファイルから作成された DOM ツリーをプログラム側から利用するためのコンポーネントとして、ラベル付きリスト XML 変換が提供されています。これは、前述の XML 変換の派生コンポーネントであり、XML ファイルを任意の形式で出力できる機能の他に、DOM ツリーをラベル付きリストに変換して MZ Platform 側で利用するための機能を持っています。さらに、任意のオブジェクトで構成されたラベル付きリストから DOM ツリーを作成し、XML ファイルとしてや XSL スタイルファイルで指定された任意の形式で出力することができます。

サンプルアプリケーションにおいては、管理するデータ全体を XML ファイルとして入出力する機能が用意されています。メインウィンドウの「データ全体操作」領域にある「XML 入力」ボタンと「XML 出力」ボタンを押すことでそれらの処理を実行することができます。

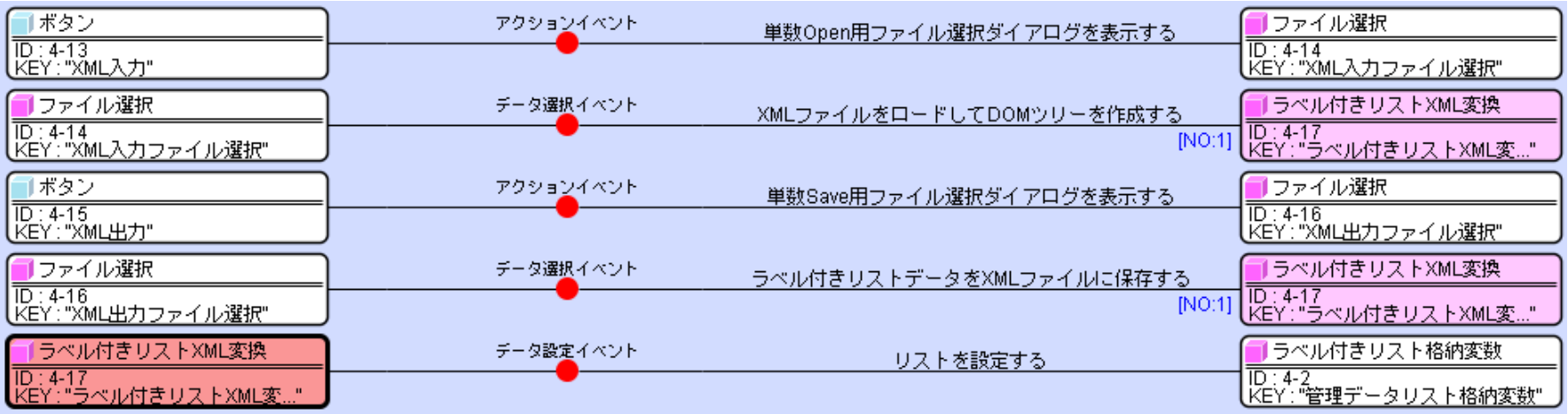


図 22　サンプルにおける XML 入出力

また、選択データを HTML ファイルに出力してウェブブラウザで表示する機能が用意されています。メインウィンドウの「データ選択」領域にあるコンボボックスからデータを選択して、「個別データ閲覧・操作」領域の「HTML 表示」ボタンを押すと、以下の図に示す設定ウィンドウが開きます。使用するウェブブラウザのパスの他に、入力となる XSL スタイルファイルと出力先の HTML ファイル名を指定して、「表示」ボタンを押すとブラウザで HTML ファイルが表示されます。XSL ファイルはサンプルアプリケーションと同じフォルダにある”mzpf1a-html.xsl”を指定して、HTML ファイルは同じフォルダ内に適当な名前(例えば”output.html”)で作成してください。ここで指定した XSL スタイルファイルで作成される HTML ファイルは同じフォルダにある”stylesheet.css”ファイルを参照するので、必ず同じフォルダ内に作成するようにしてください。

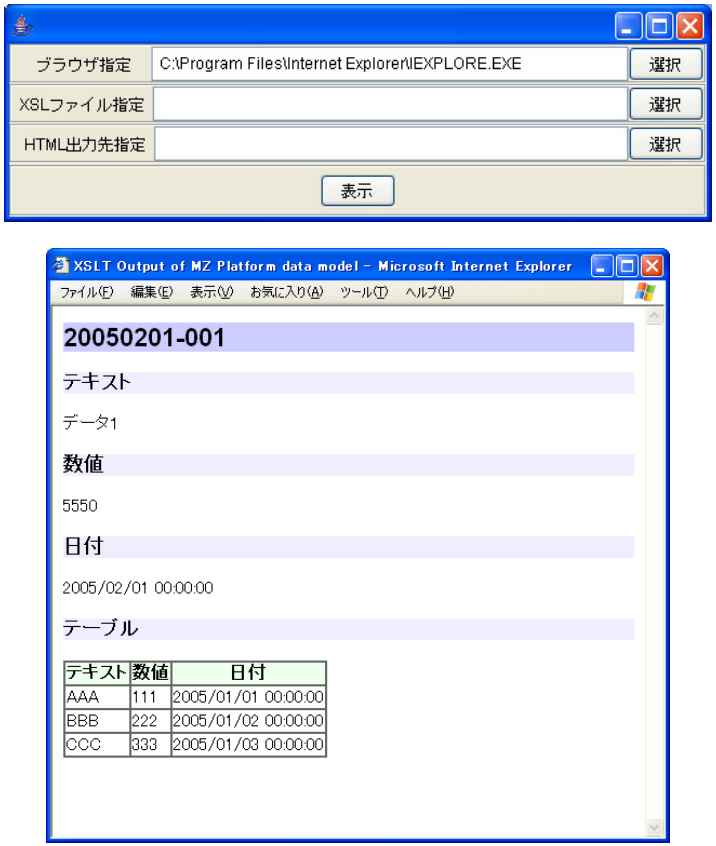


図 23　HTML 表示