2009－SE－163 (34)
2009／3／19

設計書からの網羅的な設計項目の抽出と トレース関連一括定義方式について

宇田川 佳久
三菱電機インフォメーションシステムズ株式会社

アブストラクト: CMMI などの開発標準では，品質向上のため工程を跨る設計項目の来歴管理を義務付けている．多くのシステム開発における設計項目数は数百以上であることから，来歴管理を実施するためには，網羅的，客観的かつ効率的な設計項目の抽出が前提となる．一方で，設計書は，設計の目的に最適化した記法によって記述されており，設計書間の整合性の確保を難しくしている．本文は，Web システムの外部設計と内部設計を対象とし，各種の設計書から設計項目を木構造として網羅的に抽出する方法について論じている．また，来歴管理の一手法として提唱されている設計項目間の “トレース関連” を一括して定義する方法を述べている．

A method for exhaustive extraction of design items from design documents and bulk definition of trace-relation

Yoshihisa Udagawa
Mitsubishi Electric Information Systems Corporation

Abstract: Keeping traceability on design items between development processes is mandatory to improve design quality in system development standards such as CMMI. As for more than hundreds of design items are involved in a system development, implementation of traceability depends on an exhaustive, objective and efficient extraction of design items from design documents. This paper describes a method for extracting design items exhaustively in a tree structure from several kinds of documents on a Web system design. In addition, we discuss an efficient bulk method to define “trace-relation” which is proposed for managing traceability on items in a system development.

1. はじめに

情報システムは，主要な産業の競争力，および，社会生活を支える基盤となっている．その一方で，高機能化・大規模化・短納期化など，情報システムの品質確保を難しくする要因が増大しており，体系的な手法に基づく品質管理の必要性が高まっている．

実用的な情報システム開発では，開発規模の大きさから，組織的な開発が必須で，多くの場合，ウォーターフォールモデルに基づく開発手法が採用されている [1), 5)]．ウォーターフォールモデルでは，要件定義，設計，コーディング，テストへと作業工程が順次に進むため，他の開発手法に比べて，進捗の把握，資源や品質管理が計画的に実施できる特徴を有するためである．

図1は，ウォーターフォールモデルに基づくシステム開発の典型的な資源配分を示している．要件定義から設計，製作に至るフェーズでは，工程の進捗とともに，開発者が新たに参画する．要件定義書や設計書は，その目的に最適化された表現で，ワープロソフトなどで作成されているため，新規に参画した開発者は，設計書の内容を“解読”する必要があるが，数百ページにおよぶ設計書から，限られた時間内に自己の職務遂行に必要な部分を正確に引き抜くことは困難であり，設計項目の見落とし，誤解などが混入し，システム開発の品質を低下させる可能性がある．

この品質低下への対策として，設計項目の来歴(トレーサビリティ)を管理することが有効であり，CMMI レベル 2 [3)] 以上や ISO 9000 [4)] では，設計項目の来歴の管理を義務付けている．ただし，CMMI，ISO 9000 とも来歴管理の具体的な方法には言及しておらず，個々のプロジェクトに一任しいて，統一した来歴管理方法が確立していないのが現状である．

これに対し，参考文献[2]では，一般に設計項目が階層的に管理できることに着目し，設計項目間のトレーサビリティを，2 つの木構造の関連として，グラフ理論に基づく定義を与え，CMMI，ISO 9000 でのトレーサビリティの概念と区別するために，これを “トレース関連” と呼んでいる．さらに，トレース関連が，設計項目の来歴探索を容易にするだけでなく，前工程の設計項目が後工程に網羅的に引き継がれていることを検証する手段を提供することを論じている．

本文は，Web システムを事例として，各種の設計書から設計項目を木構造として網羅的に抽出する方法と，トレー

ス関連を一括して定義する方法について論じている．2章では，トレース関連の概要と実務適用への課題について述べる．3章では，各種の設計書（画面フロー図，外部・内部画面設計書，処理機能設計書，データベース設計書）の性格と設計項目間のトレース関連について述べる．4章では，各種の設計書から設計項目を木構造として網羅的に抽出する方法について，5章では，設計項目をトレース関連によって一括して定義・管理する方法について論じている．

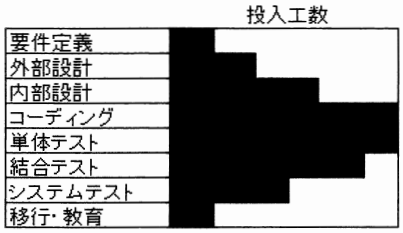

図1．工程と投入工数の例
Fig1. An example of process and resources

2. トレース関連について

2.1. トレース関連の概要

ウォーターフォールモデルは，開発工程を要件定義，外部設計，内部設計，コーディング，単体テスト，結合テスト，システムテストなどに分け，各々の工程を完結した後に次の工程に進む．ウォータフォールという名前が示すとおり，上流工程から下流工程へと後戻りせずに開発を進めていく特徴がある．

要件定義から内部設計までの各工程では，前工程の設計を後工程で詳細化している．一般に，各工程で作成される設計項目は，目次として木構造によって表現される．従って，設計項目の来歴は，前工程の設計項目の木構造から後工程の設計項目の木構造への写像として定義することができる [2]．

図2は，トレース関連のイメージを図示している．トレース関連の元になる木構造を“ソース木”，トレース関連の先になる木構造を“ターゲット木”と呼んでいる．原則として，ソース木の全ての要素は，ターゲット木の要素に対応付けられなければならない．これは，例えば，外部設計の全ての設計項目が，内部設計に引き継がれなければならないことに対応する．ソース木（前工程）の設計項目から，トレース関連が定義されている設計項目を差し引くことにより，ターゲット木（後工程）に引き継がれていない設計項目を検出できる．例えば，図2の場合，設計項目 S_{23} が後工程に引き継がれていないことを検出できる．

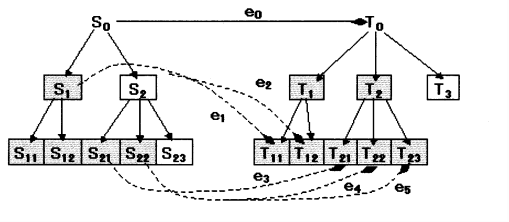

図2．設計工程のトレース関連の例
Fig2. An example of trace-relation in design

2.2. 実務適用への課題

プロジェクトは予算と工期が限られているため，設計書の品質管理に割ける工数には限りがある．一方，実用的なシステム開発では数百以上の設計項目が含まれていることから，トレース関連に基づく設計項目の管理は，網羅的かつ効率的な設計項目の抽出とトレース関連の定義が前提となる．さらに，設計書は，設計の目的に最適化した記法によって記述されており，設計項目が木構造として露になっていないものもある．以上より，トレース関連の実務適用への主な課題は下記の通りである．

① 設計書の多様性への対応
② 設計項目の網羅的，効率的，客観的な抽出
③ トレース関連の効率的な定義・管理

3. 設計書と設計項目の階層構造

3.1. 設計対象システムについて

一般に，システム開発で作成すべき設計書は，プロジェクトごとに決められているため，画一的な議論が難しい現実がある [5]．ここでは，論点を絞るため，Web システム開発の外部設計と内部設計を前提とした設計書について，設計項目の網羅的な抽出方法について論じる．また，汎用性があり，かつ，実務システムで実装されている機能として，Web システムへのログオン機能を対象とする．なお，ここで述べるログオン機能は，トレース関連を一元管理するための SQM-Trace (Systematic Quality Management by Trace-relation) と称する Web システムで実装しているものである．

3.2. 要件定義

図3は SQM-Trace の概要を示している．SQM-Trace は，トレース関連を管理する Web システムであり，ユーザは，プロジェクトの管理者，設計者などの役割で参加する．SQM-Trace のユーザ認証機能に対する要件として下記を前提とする．

(要件 1) プロジェクトに属するユーザごとに，パスワードによるユーザ認証を行う．
(要件 2) パスワードは一定期間（90 日）で更新し，期限切れ 10 日前から警告画面を表示する．
(要件 3) 警告画面では，パスワード変更画面に遷移するか，トレース関連管理機能の初期画面に遷移する．

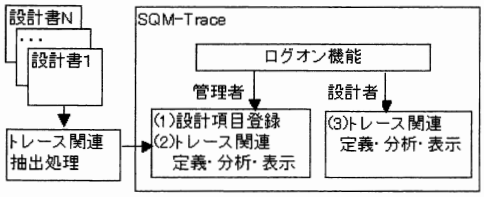

図3 SQM-Traceによる設計項目の管理
Fig.3 Design item management by SQL-Trace

3.3. 外部設計

外部設計は，要件定義で定められた要件を，情報システムの外部から，主に利用者の視点で定義するもので，発注者と開発者のコミュニケーションの中心を為すものである．外部設計は，後続の内部設計とシステムテストの入力となる重要な工程でもある．SQM-Trace のログイン機能の場合，外部設計では，画面フロー設計，外部画面設計，データベース概念設計（E-R 図）を行う．

Web システムの外部設計は，多くの場合，画面フローの全体を設計することから開始される．図４は，SQM-Trace の起動から SQM-Trace の初期画面までの画面フロー図である．一般に画面フロー図では，画面の表示順序と画面から画面への遷移の契機となるイベント（ここではボタン操作）の関係を設計することを目的としている．図4では，ボタン操作に分岐が含まれる場合は，分岐を一意に識別できるよう，ボタン名の後に":"と処理結果を追記している．

個々の画面は外部画面設計書で定義する．図5は，外部画面設計書の例であり，図4の「ブラウザ画面」と「ログオン画面」に対応するものである．この外部画面設計書は，画面に表示される入出力データとイベント（ボタン操作）によって起動される機能を定義することを目的としている．要件 1 に対応するためログオン画面では，入力データとして，ユーザ名，パスワードに加え，参加するプロジェクト名を定義している．要件 2 と 3 から，ボタン操作は，入力データに応じてログオン成功，ログオン失敗およびパスワード期限切れに分岐する可能性があり，図5では，分岐に応じた処理内容を定義している．本研究では，トレース関連の管理対象とする設計項目を【〈項目カテゴリ〉｜〈項目名〉】なる文字列で表記している．なお，外部画面設計書では，画面の表示イメージも定義するが，ここでは省略する．

図6は，SQM-Trace のユーザとプロジェクトに関する ER 図（Entity-Relationship diagram）である．1 人のユーザは，複数のプロジェクトに参画できることを考慮している．

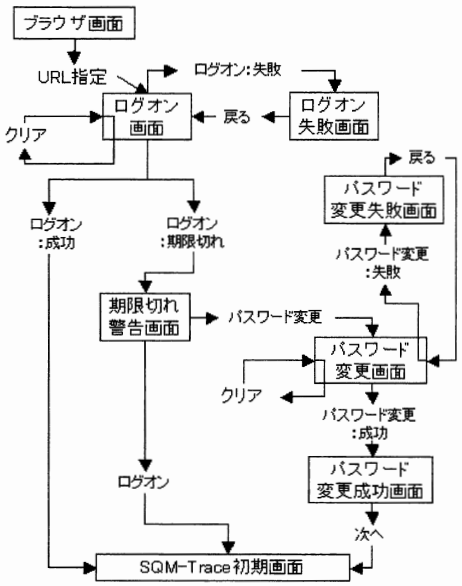

図4 画面フロー図
Fig.4 Screen Flow Diagram

3.4. 内部設計

内部設計は，外部設計で定めた各機能を，情報システムの開発者の視点から定義する．内部設計は，実装方式を前提として記述されるため，一般に，開発者間のコミュニケーションのために使われる．また．後続のコーディング，結合テストの入力となる工程でもある．

SQM-Trace の場合，Struts フレームワーク，Java と JSP 言語による実装を前提とし，内部画面設計，処理機能設計，データベーステーブル設計を行う．

図7は，ログオン画面に対応する内部画面設計書であり，フォームの定義，入出力データ，ボタンなどの主要な画面構成を HTML 言語をベースに記述している．

図8は，図5の外部画面設計書に対応する処理機能設計書の抜粋である．処理機能設計書では，Struts の実装方式を反映して，機能 ID，Action Form，Action Servlet，遷移先を定義している．以下，Struts の計算モデルと処理機能設計書の設計項目の対応について述べる．

図9は，Struts と MVC モデルとの関連を示している．ブラウザからのリクエストを契機として，Struts の計算モデル（MVC モデル）に従って処理を実行する．設計の目的は，画面とのデータ授受を行う Action Form，処理を実行する Action Servlet，処理結果に応じた遷移先（JSP）を設計することである．

【画面名｜ブラウザ画面】

項目名	タイプ	内容
【ボタン｜URL指定】	ボタン(I)	【機能｜ログオン画面表示処理】を実行する。 ログオン画面に必要なプロジェクト情報を検索し、ログオン画面を表示する。

【画面名｜ログオン画面】

項目名	タイプ	内容
【出力｜ユーザ名】	テキスト(O)	"ユーザ名"を表示する。
【入力｜ユーザ名】	テキストボックス(I)	"ユーザ名"を入力する。
【出力｜パスワード】	テキスト(O)	"パスワード"を表示する。
【入力｜パスワード】	テキストボックス(I)	"パスワード"を入力する。
【出力｜プロジェクト名】	テキスト(O)	"プロジェクト名"を表示する。
【入力｜プロジェクト名】	選択リスト(I/O)	"プロジェクト名"を選択する。
【ボタン｜ログオン】	ボタン(I)	【機能｜ログオン処理】を実行する。 ユーザIDとパスワードがデータベースと一致し、期限切れまで11日以上の場合、SQM-Traceの初期画面に遷移する。 ユーザIDとパスワードがデータベースと一致し、期限切れ10日以内の場合、期限切れ警告画面に遷移する。 ユーザIDとパスワードがデータベースと一致しない場合、ログオン失敗画面に遷移する。
【ボタン｜クリア】	ボタン(I)	【機能｜クリア処理】を実行する。ログオン画面をクリアする。

図5 外部画面設計書の例
Fig.5 An example of external screen design

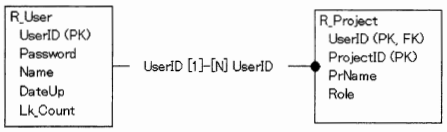

図6 ER図の例
Fig.6 An example of ER diagram

```
<title>ログオン画面</title>
ユーザログオン
<form method="post" action="/LogOnAction">
ユーザ名: <input type="text" name="UID" size="12">
パスワード: <input type="text" name="PWD" size="12">
プロジェクト名:<select name="PID">
[ <option value="pjt"> xxx ]
 </select><p>
<input type="submit" value="ログオン">
<input type="reset" value="キャンセル">
</form>
```

図7 内部画面設計書の例
Fig.7 An example of internal screen design

(1) Action Form は，入力データを取り込み，入力データに対する処理結果を保管する．図8の 2.1 では，LogOnForm という名称の Action Form を定義している．

(2) Action Servlet は，処理の中心であり，図8の 2.2 では，ログオン処理を実行する LogOnProcess という名称の Action Servlet を定義している．

(3) 遷移先(JSP)は，処理結果としての画面を生成する．図8の 2.3.1 では，ログオン成功時に初期画面表示(Trace.do)を実行することを設計している．2.3.2 では，期限切れ警告画面(LogOnWarning.jsp) の表示，2.3.3 ではログオン失敗画面(LogOnFail.jsp)の表示処理を定義している．

なお，外部設計と内部設計では，設計の視点の違いから，同じ対象を異なる名称で言及しており，対象の同一性を判断するためには，用語の置き換えが必要になる．例えば，外部設計の「ログオン処理」は，内部設計では「LogOnAction」に読み替える必要がある．詳細については，5章で述べている．

3.5. 設計項目の階層構造

図10は，以上で述べた設計書に関する設計項目の木構造とトレース関連を示している．

3.5.1. 外部設計の設計項目

画面フロー図は，ある画面のボタン操作(イベント)によって引き起こされる画面遷移を記述している．画面の下位要素としてボタンがあり，ボタンの下位要素として遷移先画面(次画面)があることから，画面―ボタン―次画面という木構造によって，画面フロー設計書をモデル化(設計書の骨子となる内容を表現)することができる．画面の下位項目にボタンがあるのは，画面の存在を前提として，ボタンが存在するからである．言い替えれば，画面がなければボタンは存在しえない．同様に，ボタンの下位項目として次画面があるのは，ボタンの存在を前提として次画面が存在するからである．

外部画面設計書は，出力データ，入力データ，画面に表示されるボタンおよび機能を記述している．このため，外部画面設計書を，画面名―出力データ，画面名―入力データ，画面名―ボタン―機能という木構造でモデル化する．画面名が最上位にあるのは，画面の存在を前提として，出力データ，入力データ，ボタン，および機能が存在するからである．

画面フロー図と外部画面設計書とには過不足があってはならない．この条件は，外部画面設計の [画面名―ボタン名] から画面フロー図の [画面名―ボタン名] へのトレース関連に過不足がないことで確認することができる．

時間的な順序としては，画面フロー図が外部画面設計書に先行するが，画面フロー図は，外部画面設計書よりもボタンに関して詳しく定義しているため，外部画面設計書を画面フロー図に先行する設計書と位置付けるのがトレース関連の管理上好ましい．

1. LogOn
　・LogOn初期画面を表示する。
　1.1 Null
　1.2 LogOnDispProcess
　　・すべてのProjectIDを検索し、画面(LogOn.jsp)の選択項目に書き込む。
　　SELECT DISTINCT ProjectID FROM R_Project
　1.3 遷移先
　　1.3.1 LogOn.jsp←right
　　・LogOnDispProcess の戻り値を"right"とし、LogOn.jspを実行する。
2. LogOnAction
　・ログオン処理(LogOnAction.do)を実行する。
　2.1 LogOnForm
　・ユーザID、パスワード、プロジェクトIDを画面から取得する。
　2.2 LogOnProcess
　・ユーザIDをキーとしてユーザテーブルを検索し、パスワードと期日を検索する。
　　SELECT Password, DateUp FROM R_User U, R_Project P
　　WHERE U.UserID=P.UserID and UserID=$UID
　　and ProjectID=$PID
　・パスワードが一致し、期日と本日の差が10日以上なら、SQM-Traceにログオン(Trace.do)する。
　・パスワードが一致し、期日と本日の差が10日以内なら、期限切れ警告画面(LogOnWarning.jsp)を表示する。
　・パスワードが一致しない場合、ログオン失敗画面(LogOnFail.jsp)を表示する。
　2.3 遷移先
　　2.3.1 Trace.do←Success
　　・LogOnProcess の戻り値を"Sccess"とし、SQM-Traceの初期画面表示を実行する。
　　2.3.2 LogOnWarning.jsp←Warning
　　・LogOnProcess の戻り値を"Warning"とし、LogOnWarning.jspを実行する。
　　2.3.3 LogOnFail.jsp←Fail
　　・LogOnProcess の戻り値を"Fail"とし、LogOnFail.jspを実行する。

図8 処理機能設計書
Fig.8 Process design

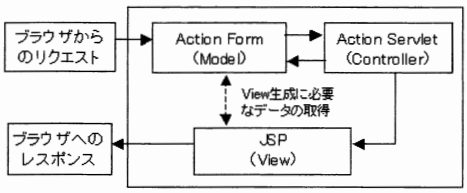

図9 StrutsにおけるMVCモデル
Fig.9 Struts and MVC process model

3.5.2. 内部設計の設計項目

内部設計では，内部画面設計，処理機能設計，テーブル設計が行われる．内部画面設計書の設計項目は，外部画面設計書の設計項目と同様であり，この 2 つの設計書間のトレース関連によって，設計項目間の整合性を検証することができる．

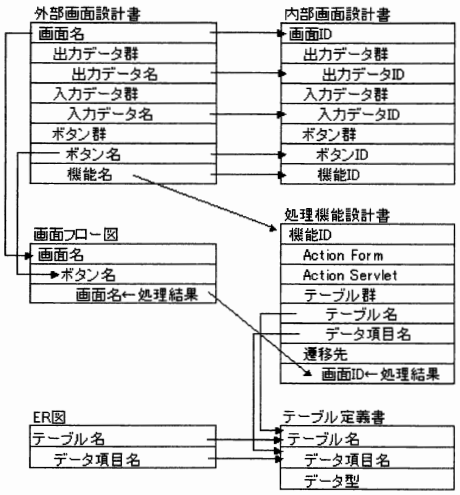

図10 設計項目の階層構造とトレース関連
Fig.10 Design Item Structure and Trace-relation

処理機能設計書に記載されたSQL文で使われているテーブル名とデータ項目名は，テーブル定義書で定義されていなければならない．この条件は，処理機能設計の[テーブル名—データ項目名]から，テーブル定義書の[テーブル名—データ項目名]へのトレース関連によって検証することができる．

3.5.3. 外部設計書と内部設計書間の設計項目

外部画面設計書と内部画面設計書は，類似した設計項目の構造を有している．したがって，図10に示した設計項目間のトレース関連によって，過不足を検証することができる．ただし，設計項目の名称は，外部設計と内部設計で異なるものが使われている．例えば，外部設計書の[機能名]は，処理機能設計書の[機能ID]に対応する．また，画面の遷移先は，画面フォロー図の[画面名←処理結果]と処理機能設計書の[画面ID←処理結果]に対応する．外部設計と内部設計で使われる名称の対応関係は，5章で述べる用語の対応表で管理している．

4. 設計項目の抽出

3章で述べたように，設計書はその目的に最適化した表現法で記述されている．したがって，図10に示した設計項目の木構造を各設計書から網羅的に抽出するためには，設計書ごとに固有の抽出処理を開発する必要がある．以下に述べる設計書は，マイクロソフト社のExcelによって作成し，設計項目の抽出はExcelマクロプログラムによって実装している．

4.1. 画面フロー図

図4の画面フロー図は，画面を枠付きのテキストボックスで，画面間の遷移を枠なしのテキストボックスで表現し，テ

キストボックス間をコネクターで結合している．コネクターで結合しているテキストボックス間の構造を解析することによってトレース関連の定義に必要な設計項目を抽出することができる． 図11は，図4の画面フロー図からの設計項目の抽出結果である．なお，抽出処理では，外部画面設計書のボタン名および処理機能設計書の遷移先画面名との整合性を保つため，図4の ［ボタン:処理結果］における“処理結果”を 遷移先画面の［画面名←処理結果］ に付け替えている．

0	画面フロー図
1	ブラウザ画面
1.1	URL指定
1.1.1	ログオン画面
2	ログオン画面
2.1	ログオン
2.1.1	ログオン失敗画面←失敗
2.1.2	アプリケーション初期画面←成功
2.1.3	期限切れ警告画面←期限切れ
2.2	クリア
2.2.1	ログオン画面
3	パスワード変更失敗画面
3.1	戻る
3.1.1	パスワード変更画面
4	ログオン失敗画面
4.1	戻る
4.1.1	ログオン画面
5	期限切れ警告画面
5.1	ログオン
5.1.1	アプリケーション初期画面
5.2	パスワード変更
5.2.1	パスワード変更画面
6	パスワード変更画面
6.1	パスワード変更
6.1.1	パスワード変更失敗画面←失敗
6.2	クリア
6.2.1	パスワード変更画面
6.3	パスワード変更
6.3.1	パスワード変更成功画面←成功
7	パスワード変更成功画面
7.1	次へ
7.1.1	SQM-Trace初期画面

図11 画面フロー図からの設計項目抽出
Fig.11 Design item from Screen flow diagram

4.2. 外部画面設計書

図12は，図5の外部画面設計書から抽出した設計項目である．設計項目は，【画面名，【出力，【入力，【ボタン，【機能 をキーワードとして抽出している．なお，項目番号は，設計項目の出現順に付与している．

4.3. 内部画面設計書

本研究では，下記のルールに基づき，内部画面設計書の HTML 文を解析し，設計項目を抽出している．設計項目の構造は，外部画面設計書と同様であるので，抽出結果は省略する．

0	外部画面
1	URL入力画面
1.1	出力データ群
1.2	入力データ群
1.3	ボタン群
1.3.1	ボタン機能1
1.3.1.1	URL指定
1.3.1.2	ログオン画面表示処理
2	ログオン画面
2.1	出力データ群
2.1.1	ユーザ名
2.1.2	パスワード
2.1.3	プロジェクト名
2.2	入力データ群
2.2.1	ユーザ名
2.2.2	パスワード
2.2.3	プロジェクト名
2.3	ボタン群
2.3.1	ボタン機能1
2.3.1.1	ログオン
2.3.1.2	ログオン処理
2.3.2	ボタン機能2
2.3.2.1	クリア
2.3.2.2	クリア処理

図12 外部画面設計書からの設計項目抽出
Fig.12 Design item from external screen design

・title タグで囲まれた文字列を画面名とする．
・input タグの type=″text″ および ″password″ の name に対応する値を入力データ名とする．
・input タグの type=″submit″ または ″reset″ の value に対応する値をボタン名とする．
・上記のタグで囲まれていない文字列を出力データ名とする．

4.4. 処理機能設計書

図13は，図8の外部画面設計書から抽出した設計項目である．処理機能設計書は，実装のフレームワークである Struts の実装方式を反映し，機能 ID, Action Form, Action Servlet と遷移先を定義している．設計項目の構造は，処理機能設計書の項目とほぼ1対1に対応している．SQL 文については構文を解析し，使用しているテーブル名とデータ項目名を抽出している．

機能 ID は，外部画面設計書の機能名と対応している（ただし，用語の変換が必要）．Action Form と Action Servlet に関するIDは，ソースプログラムとのトレース関連を想定し，内部処理設計書で新たに導入した設計項目であり，外部設計書に対応する設計項目はない．遷移先は，Action Servlet の処理結果によって分岐するため，[画面ID←処理結果] という形式で遷移先の画面を列記している．この遷移先は，画面フロー図の ［ボタン:処理結果］に対応するものである．

ER図，テーブル定義書からの設計（データ）項目の抽出は，単純な抽出処理で実装できることから，本文では省略する．

0	内部処理
1	LogOnDispAction
1.1	LogOnDispForm
1.2	LogOnDispProcess
1.2.1	テーブル名
1.2.1.1	R_Project
1.2.2	データ項目名
1.2.2.1	ProjectID
1.3	遷移先
1.3.1	LogOn.jsp←right
2	LogOnAction
2.1	LogOnForm
2.2	LogOnProcess
2.2.1	テーブル名
2.2.1.1	R_User
2.2.1.2	R_Project
2.2.2	データ項目名
2.2.2.1	Password
2.2.2.2	DateUp
2.3	遷移先
2.3.1	Trace.do←Success
2.3.2	LogOnWarning.jsp←Warning
2.3.3	LogOnFail.jsp←Fail

図13 内部画面設計書からの設計項目抽出
Fig.13 Design item from internal screen design

5. トレース関連の一括定義

5.1. 設計項目の識別

設計項目間のトレーサビリティを管理するためには，設計項目を一意的に識別することが前提になる．CMMI などでは，設計項目に対し，一定の体系に基づく識別 ID を付与することを推奨している．しかし，識別 ID の付与は，目視で行われ，現実的には無視できない工数を要する．

本文では，外部設計における名称と内部設計における ID の対応表を作成し，この対応表に基づいた名称と ID のパターンマッチによってトレース関連を一括定義する方法を述べる．

5.2. トレース関連の一括定義

4章で述べたように，設計項目を階層的に定義しているため，同等の役割を果たす設計項目であっても，設計書によって出現する階層が異なる．

例えば，画面に配置される「ボタン」は，外部画面設計書では，第2階層が “3” で第4階層が ”1” の項目に編集している．すなわち，”#” で任意の番号を表すとき，“#.3.#.1” の位置に編集している．一方，「ボタン」は，画面フロー図では，第2階層目 (“#.#“) に編集している．ただし，上記の対応は特定の画面に限定したものであるから，正確には，画面名称が一致していることが前提になる．すなわち，外部画面設計書から画面フロー図への「ボタン」に関するトレース関連は，画面名が一致し，かつ，ボタン名が一致する下記の条件で定義する．

外部画面設計書の画面名 (“#”) ＝画面フロー図の画面名 (“#”)
外部画面設計書のボタン名 (“#.3.#.1”) ＝
画面フロー図のボタン名 (“#.#”)

5.3. 用語の変換

外部設計は利用者の視点，内部設計は開発者の視点で作成されるため，同じ設計対象でも異なる名称 (ID) が使用される．したがって，外部設計から内部設計へのトレース関連を定義するためには，用語の読み替えが必要になる．一般に，用語は使用されている文脈に依存する．例えば，外部設計で「ユーザ名」は，入力データとしても，出力データとしても使われている．したがって，用語を変換する場合，用語そのものだけでなく，その用語が使われている文脈も考慮する必要がある．文脈としては，設計書の種類と設計項目の役割 (抽出した設計項目階層での位置) を指定する必要がある．

図14は，内部設計の用語と文脈をキーにして外部設計の用語に変換するための用語の対応表の例である．

R_TERM : テーブル

ID	TNAME	TProcess	TC_No	SNAME	SProcess
1	ユーザ名:	内部画面	#1#	ユーザ名	外部画面
2	パスワード:	内部画面	#1#	パスワード	外部画面
3	プロジェクト名:	内部画面	#1#	プロジェクト名	外部画面
4	LogOnAction	内部画面	#3#2	ログオン処理	外部画面
5	HTML	内部画面	#3#2	クリア処理	外部画面
6	UID	内部画面	#2#	ユーザ名	外部画面
7	PWD	内部画面	#2#	パスワード	外部画面
8	PID	内部画面	#2#	プロジェクト名	外部画面
9	ログオン	内部画面	#3#1	ログオン	外部画面
10	キャンセル	内部画面	#3#1	クリア	外部画面
11	TraceDB←Success	内部処理	#3#	アプリケーション初期画面←成功	画面フロー図
12	LogOnWarning.jsp←	内部処理	#3#	期限切れ警告画面←期限切れ	画面フロー図
13	LogOnFail.jsp←Fail	内部処理	#3#	ログオン失敗画面←失敗	画面フロー図
14	LogOn.jsp	内部処理	#3#	ログオン画面	画面フロー図

図14 用語の対応表の例
Fig.14 An example of term mapping table

5.4. 実装結果

トレース関連を一元管理するための Web システム SQM-Trace (Systematic Quality Management by Trace -relation) を開発した．図15は，トレース関連の一括登録操作画面の例である．なお，5.2 で述べた設計項目の階層は，変換テーブルにより，一般的な用語に変換しているが，詳細は省略する (例：外部設計書における “#.3.#.1” を「ボタン名」に変換)．

図2に示したように，トレース関連は2つの木構造の任意の要素間に定義されるものであり，その表示方法としては，マトリックス，木構造と要素間の矢印，要素の対応一覧によるものがある．ここでは，トレース関連を簡潔かつ効率的に表示できることから，要素の対応一覧による表示について述べる．

要素の対応一覧は，次の手順で作成することができる．

① 基準となる木構造 (以降，基準木) を深さ優先で探索し，要素一覧を作成する．
② 基準木の全ての要素に対し，トレース関連で定義されている相手側 (基準木がソース木であればターゲット木) の要素を基準木の要素に紐付けてリストする．
③ 基準木の要素で，下位の要素の全てがトレース関連定義済みの場合，その要素をトレース関連定義済みとする．

図16は，画面フロー図をソース側の基準木（設計書）として，トレース関連のターゲット側の要素を一覧表示したものである（前方探索）．“>” 印が付いた設計項目は，その設計項目をソースとするトレース関連が定義されていることを示している．”∧” 印は，下位の全ての設計項目にトレース関連が定義されているために，（実質的に）トレース関連が定義されている設計項目を示している．

図17は，画面フロー図をターゲット側の基準木（設計書）としてトレース関連のソース側の要素を一覧表示したものである（後方探索）．“<” 印が付いた設計項目は，その設計項目をターゲットとするトレース関連が定義されていることを示している．

前方探索と後方探索を組み合わせることにより，設計項目間の来歴を効率的に確認することができる．

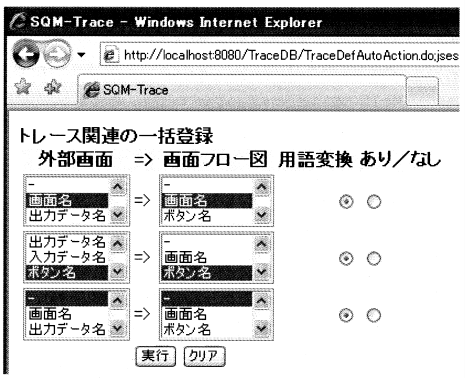

図15 一括登録の操作画面の例
Fig.15 An example of bulk definition

基準設計書	前方設計項目
∧ 0 画面フロー図	-
∧ 1 ブラウザ画面	-
∧ 1.1 URL指定	-
> 1.1.1 ログオン画面	内部処理 # 1.3.1 LogOn.jsp
∧ 2 ログオン画面	-
∧ 2.1 ログオン	-
> 2.1.1 ログオン失敗画面←失敗	内部処理 # 2.3.3 LogOnFail.jsp←Fail
> 2.1.2 アプリケーション初期画面←成功	内部処理 # 2.3.1 TraceDB←Success
> 2.1.3 期限切れ警告画面←期限切れ	内部処理 # 2.3.2 LogOnWarning.jsp←Warning
∧ 2.2 クリア	-
> 2.2.1 ログオン画面	内部処理 # 1.3.1 LogOn.jsp

図16 前方探索の例
Fig.16 An example of forward trace

後方設計項目	基準設計書
-	0 画面フロー図
-	1 ブラウザ画面
外部画面 # 1.3.1.1 URL指定	< 1.1 URL指定
-	1.1.1 ログオン画面
外部画面 # 2 ログオン画面	< 2 ログオン画面
外部画面 # 2.3.1.1 ログオン	< 2.1 ログオン
-	2.1.1 ログオン失敗画面←失敗
-	2.1.2 アプリケーション初期画面←成功
-	2.1.3 期限切れ警告画面←期限切れ
外部画面 # 2.3.2.1 クリア	< 2.2 クリア
-	2.2.1 ログオン画面

図17 後方探索の例
Fig.17 An example of backward trace

6. おわりに

品質向上の手法として設計項目の来歴管理がある．来歴管理は，必要性が認められているにもかかわらず，数百以上の設計項目に加え，設計書の表記上の多様性のために実務での実践が困難な状況にある．本文では，各種の設計書から，設計項目を木構造として網羅的に抽出する方法を論じた．さらに，設計項目間のトレース関連を一括して定義する方法とトレース関連管理システム（SQM-Trace）による実装結果を示した．

今後は，テスト工程における来歴管理について研究を進める計画である．

参考文献

[1] レフィングウェル，ウィドリング著，石塚，荒川監訳：ソフトウェア要求管理，ピアソン・エデュケーション，2002.

[2] 宇田川：ウォーターフォールモデルに基づく情報システム開発における成果物の品質管理手法について，情報処理学会論文誌，Vol.49，pp922-929，2008.

[3] Understanding and Leveraging a Supplier's CMMI Efforts -- A Guidebook for Acquirers CMMI Guidebook for Acquirers Team: Technical Report CMU/SEI-2007-TR-004, Carnegie Mellon University, 2007.

[4] ISO 9001-2000: ISO (International Organization for Standardization) ISO/TC 176 (2000).

[5] 実践的アプローチに基づく要求仕様の発注者ビュー検討会：失敗しない外部設計，日経BP社，2008.