Vol.2009-IS-110 No.3
2009/12/1

情報システム開発おける トレーサビリティ管理へのアプローチ

宇田川　佳久[†]

情報システム開発における品質向上の有効な手法として設計項目のトレーサビリティ管理があり，CMMI などの開発標準では，開発工程を跨る設計項目のトレーサビリティ管理を義務付けている．本文は，Web アプリケーションの要件定義，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手法と，設計項目間のトレーサビリティを管理するデータディクショナリーについて論じている．トレーサビリティ管理のモデルとしては著者らが先に提唱した "トレース関連" 採用しており，設計項目の網羅性を確認する手段を提供している．本研究の特徴は，トレース関連の定義対象である設計書と設計項目自体をデータディクショナリーで管理するため，柔軟なトレース関連の定義を可能にしていることである．プロトタイプの実装により，トレース関連に基づくトレーサビリティ管理の基本機能を検証している．

An approach to traceability management in information system development

Yosihisa Udagawa[†]

Managing traceability on design items among development processes is effective to improve the quality of information systems developed, and thus it is mandatory in system development standards such as CMMI. This paper describes techniques for extracting design items exhaustively in a hierarchy from requirement specifications and design documents on a Web application. We discuss the structure of a data dictionary to store design items and traceability information. As a model for traceability, we adopt "trace-relation" which is a binary relationship between design items and provides a method to verify that design items are exhaustively related. Since the data dictionary involves meta data on trace-relation, it allows us to flexibly define documents and design items on which trace-relations are defined. Prototype implementation shows that a set of essential functions to manage the trace-relation is successfully realized.

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，オフショア開発など，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．

品質低下への対策として，設計項目のトレーサビリティを管理することが有効であり，米国カーネギーメロン大学ソフトウェア工学研究所の能力成熟度モデル統合 (CMMI) 2) や米国カリフォルニア州交通局の ITS システム構築ガイドブック 1) などでは，設計項目のトレーサビリティ管理を義務付けている．ただし，CMMI，ITS システム構築ガイドブックともトレーサビリティ管理の具体的な方法については言及しておらず，個別のシステム開発プロジェクトに一任している．

著者らは，トレーサビリティの管理が現実の開発プロジェクトで十分に実施されていない主な原因は，下記があると考えている．

(1) 設計書の記述方法について共通の方法が確立していないため 3)4)，設計項目を抽出する基準や方法が確立していない．
(2) トレーサビリティに関するモデルが確立していない 6)7)．
(3) トレーサビリティ管理に関する実装事例が少ない 6)7)．

このような背景から，著者らは下記の方針でトレーサビリティに関する研究を推進している．(1) に対しては，多くの開発プロジェクトで使われている表計算ツールの使用を前提とした設計書の記述方法を定め，設計項目を網羅的に抽出する方法についてプロトタイプを実装し，実現性を検証する．(2)に対しては，設計書に共通するものとして設計項目の目次が木構造として構成されることに着目し，トレーサビリティを 2 つの木構造の関連として定義する "トレース関連" を提唱した 8)．(3)に対しては，データディクショナリーを使った "トレース関連" のプロトタイプを実装し，トレーサビリティ管理に関する実装事例を提供する．

本文は，上記の(1)(3)に関する研究結果について論じている．2 章では，トレーサビリティ管理の課題と技術標準 CMMI における扱いについて述べる．3 章では，トレース関連を管理するデータディクショナリーの構成とトレース関連の構造について論じている．4 章では，Web アプリケーションにおけるログオン処理に関する要件定義書，外部設計書，内部設計書の記述方法と各設計書から設計項目を網羅的に抽出する方法について論じている．5 章では，プロトタイプの実装結果を示し，トレース関連に基づくトレーサビリティ管理の基本機能を検証した．6 章では課題と今後の研究分野について述べている．

[†] 三菱電機インフォメーションシステムズ(株)
Mitsubishi Electric Information Systems Corp.

2. トレーサビリティ管理の課題と CMMI

2.1 トレーサビリティ管理の課題

1970 年代から続く研究から,トレーサビリティ管理を困難にする課題として下記が指摘されている 6)7).

(1) 成果物が異なる言語（自然言語，プログラミング言語）で記述されている．
(2) 成果物が異なる抽象レベル（要求，設計，実装）で記述されており，表現方法が異なる．
(3) 要求変更，設計変更に対し，トレーサビリティの変更および再定義が発生する．

これらの課題解決には，成果物の記述方法のモデル化を推し進めるアプローチがあり，本研究もこのアプローチに位置づけられるものである．

2.2 技術標準 CMMI におけるトレーサビリティ管理

CMMI 2) は，米国カーネギーメロン大学ソフトウェア工学研究所（SEI: Software Engineering Institute）がソフトウェアの開発能力を客観的に評価すための指標として定めたものであり，開発のプロセスに注目した評価を行う．なお，CMMI の規格書は SEI のホームページから無償でダウンロードすることができる．

CMMI は，ソフトウェアの開発能力を以下の 5 段階で評価している．

レベル 1　初期：プロセスがメンバーに依存し，組織としては確立していない段階
レベル 2　管理：反復してプロセスが実行できる段階
レベル 3　定義：プロセスが組織で定義され共有されている段階
レベル 4　定量的管理：品質測定基準に基づきプロセスが管理されている段階
レベル 5　最適化: 品質測定基準に基づき継続的にプロセスを最適化している段階

レベル 1 は，組織としてのプロセスが確立されていない段階であり，実質的にはレベル 2 から 5 が，企業や組織が達成すべきレベルである．

現時点で最新である CMMI バージョン 1.2（以降，単に CMMI と表記）は，要求管理, プロジェクト計画策定, 構成管理など 22 個のプロセス領域で構成されていて，レベルに応じてサポートすべきプロセス領域が定められている．このうちトレーサビリティに関する記述があるのは，表-1 に示した 10 個のプロセス領域である．なお，トレーサビリティが関連するプロセス領域のレベルは 2 から 4 であり, CMMI においてトレーサビリティが基本条件となっていることが分かる．

トレーサビリティの要求事項が顕著な要求管理プロセス(Requirements Management)において，トレーサビリティは，要求変更によって発生するプロジェクト計画や成果物への影響を把握する手段として特に重要である．効果としては，原要求と詳細化された要求の間の双方向トレーサビリティにより，原要求が網羅的に詳細化されていることを確認することができる．また，中間成果物と最終成果物，要求変更の内容，要求とテスト計画の対応についても探索可能になる．

表 1 トレーサビリティに関連する CMMI Ver.1.2 のプロセス領域
Tab.1 Process areas in CMMI Ver. 1.2 concerning traceability

No	英語名称	日本語名称	レベル
1	Configuration Management	構成管理	2
2	Decision Analysis and Resolution	決定分析と解決	3
3	Integrated Project Management	統合プロジェクト管理	3
4	Measurement and Analysis	測定と分析	2
5	Quantitative Project Management	定量的プロジェクト管理	4
6	Requirements Development	要求開発	3
7	Requirements Management	要求管理	2
8	Supplier Agreement Management	供給者合意管理	2
9	Technical Solution	技術解	3
10	Verification	検証	3

3. トレース関連とデータディクショナリー

3.1 トレーサビリティとトレース関連

著者らは，一般にシステムエンジニアリングでは，システムを機能または構成要素の木構造として捉えて設計，実装すること，ならびに，成果物としての設計書における記載項目は一般的な書籍に付けられている目次(木構造)が存在することに着目し，トレーサビリティを 2 つの木構造間の関連（以降，トレース関連と呼ぶ）として定義した．図 1 は，トレース関連の概要を図示している．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．ソース木（前工程）の要素から，トレース関連が定義されている要素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要素を検出できる．例えば，図 1 の場合，要素 S_{23} が後工程に引き継がれていないことを検出できる．

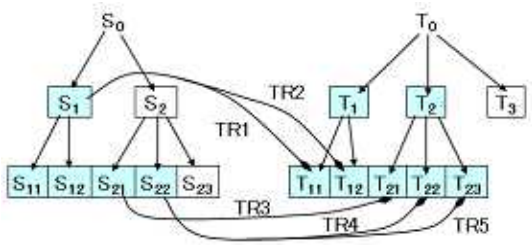

図 1 設計工程のトレース関連の例
Fig.1 An example of trace-relation in design

3.2 設計書とデータディクショナリー

本節では，設計工程（要件定義，外部設計と内部設計）における設計項目のトレース関連を管理するための SQM-Trace　(Systematic Quality Management by Trace-relation) と称するデータディクショナリーについて述べる．以降，記述を簡便にするため，要件定義書，外部設計書（外部画面設計書，画面フロー図など），内部設計書（内部画面

設計書，処理機能設計書など）を単に設計書という用語で表現するものとする．

図 2 は，設計書と SQM-Trace データディクショナリーとの関連を概念的に図示したものである．本文の 1 章で述べたように，情報システム開発では，設計書の記述方法に関する標準化が十分でなく，設計書は個々のプロジェクトごとに定めた基準に則して作成されているのが現状である 3)4)．したがって，設計書ごとに，設計書の内容を解析しトレース関連の定義対象となる項目を抽出し，データディクショナリーに登録する処理を個別に開発する必要がある．この処理の詳細については 4 章で論じている．

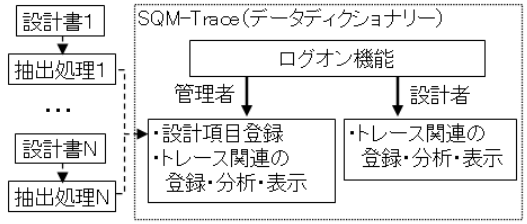

図 2 設計書とデータディクショナリー
Fig.2 Design documents and the data dictionary

3.3 SQM-Trace データディクショナリーの構造

SQM-Trace データディクショナリーに関する要件を下記とする．

(1) 要件定義書および設計書にはバージョンがあり，ベースライン管理の対象とする．
(2) 要件定義書および設計書に記載された項目は，章節項で番号付けされた木構造を成し，一般的な書籍における目次が存在する．なお，章節項は無作為に作られるのもではなく，記述対象とするテーマが決められ，項目名が付けられているものとする．例えば，外部画面設計書の各章では画面を定義し，章のタイトルは画面名を付与するといった基準が存在する．
(3) 設計書に記載された項目間のトレース関連を管理する．なお，本文では，トレース関連のタイプとして設計の詳細化のみを対象とする．
(4) トレース関連の定義対象である設計書と設計項目，および，トレース関連を定義する方法（自動または手動）自体をデータディクショナリーで管理する．

図 3 は上記の要件を満たすデータディクショナリーのテーブル構造を示している．なお，"PK"は主キー，"FK"は外部キーを示す．

テーブル R_Baseline は，ベースラインを構成する設計書タイプとバージョンに関する情報を記憶している．主キーは，ベースライン番号(Baseline_ID)と設計書タイプ(DocType)であり，ベースライン番号と設計書タイプが決まると，バージョン番号(Ver_ID)が一意に決まる．本文では，設計書群として要件定義書，外部画面設計書，画面フロー図，内部画面設計書と処理機能設計書を作成するものとし，以降，テーブル R_Baseline の設計書タイプ (DocType) に登録済みであるものとする．

テーブル R_Title_Tree は，設計書を構成する項目の木構造を表現するもので，主キーは，設計書タイプ(DocType)，バージョン番号(Ver_ID)，項目タイトル(C_Title)と項目番号(C_No)である．設計書タイプ(DocType)には，テーブル R_Baseline からの外部キー制約がある．

テーブル R_Trace は，トレース関連を記憶するためのものである．テーブル R_Trace の主キーは，ソース木の要素を示す項目(SDocType, SVer_ID, SC_Title, SC_No)とターゲット木の要素を示す項目(TDocType，TVer_ID，TC_Title，TC_No）である．これらの項目には，テーブル R_Title_Tree からの外部キー制約がある．

テーブル R_Trace_Schema は，トレース関連の定義対象となる設計書と項目，および，トレース関連を定義する方法(自動または手動）を記憶するためのものである．テーブル R_Trace_Schema の主キーは，ソースとなる設計書タイプ(SDocType)とバージョン番号 (SVer_ID) と項目パターン(SC_NoPT)，および，ターゲットとなる設計書タイプ(TDocType)とバージョン番号 (TVer_ID) と項目パターン(TC_NoPT)である．項目パターンは，ワイルドカード“#”と章節項を表す番号を使って表記する．具体的な記述例は本文 5 章で述べている．属性 Flag は，トレース関連を自動で定義するか手動で定義するかを示すフラッグ(Flag)である（図 16 参照）．

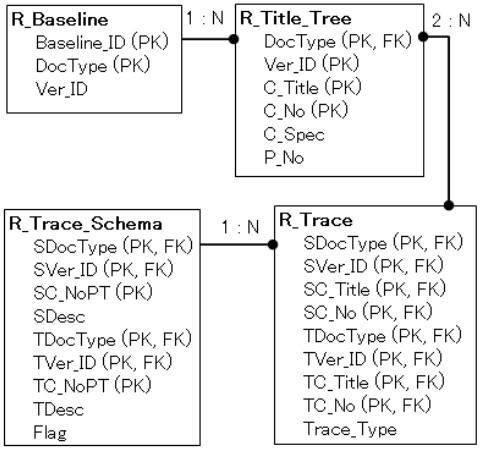

図 3 データディクショナリーのテーブル構造
Fig.3 Table structure of the data dictionary

3.4 設計書の構成とトレース関連

本文では，下記に示した基準で設計書を作成するものとする．

(1) 要件定義では，要件定義書を作成し，画面ごとに画面名と要件を記述する．要件定義書で定める画面名は外部設計に引き継がれること．
(2) 外部設計では，各画面で入出力するデータ項目，ボタン，ボタン操作で起動される機能（外部画面設計書），および，画面の遷移（画面フロー図）を設計する．これらの設計項目は，内部設計に引き継がれること．
(3) 内部設計では，各画面の実装方法（内部画面設計書）とボタン操作によって起動される機能（処理機能設計書）を設計する．

図 4 は設計書の項目の木構造と項目間のトレース関連を図示したものである．図 4 における項目のインデントは，木構造の階層の深さを表している．以下，図 4 の内容

を説明する．

要件定義書では画面名を章（第 1 階層）のタイトルとして記述する．要件定義書から外部画面設計書へのトレース関連は，要件定義書で定める画面名が外部設計に引き継がれることを示している．

外部画面設計書は，画面ごとに出力データ，入力データ，ボタンおよびボタン操作で起動される機能を記述する．出力データ，入力データ，ボタンは 1 つの画面に複数存在する可能性があることから，それぞれ "出力データ群"，"入力データ群"，"ボタン群" ，"ボタン機能 N" という項目を設け，その下位に具体値を定義するものとする．

画面フロー図は，遷移元の画面名，遷移の契機となるボタン名および遷移先の画面名を定義するものとする．外部画面設計書から画面フロー図へのトレース関連は，画面フロー図で定義する画面名とボタン名が外部画面設計書で定義されていることを前提とすることを示している．

内部画面設計書は，画面の実装方式を定義するものであり，画面は実装で使われる識別 ID を使って定義されるものとする．一般に，実装で使われる識別 ID は，外部設計で使われる項目名称とは異なるため，識別 ID と項目名称の同一性を機械的に判断することは困難である．本研究では，外部画面設計書の設計項目を内部画面設計書の設計項目に関連つけるためには設計者の判断に基づく対応付けを行っている．この対応付け方法については本文 5 章で述べる．

処理機能設計書も，実装方式を反映した方法で記述する．本研究では，実装のフレームワークとして Jakarta プロジェクト (http://www.jakarta.org/) の Struts ®1.2 を採用していることから，Struts® 1.2 の実装方式を反映して，機能 ID, Action Form, Action Servlet および遷移先を定義するものとする．

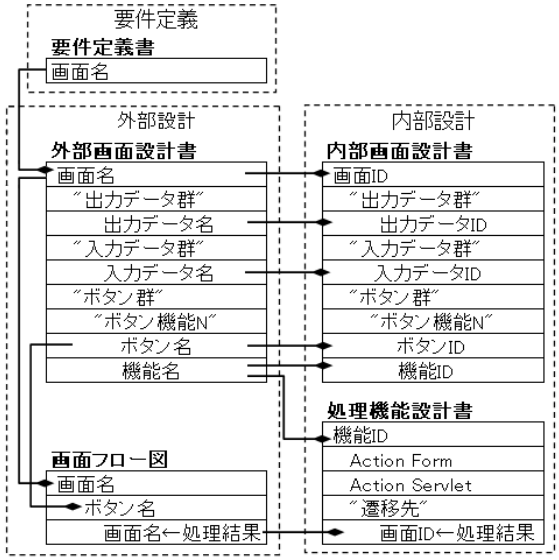

図 4 設計項目の構造とトレース関連
Fig.4 Design item structure and Trace-relation

4. 設計書の記述と設計項目の抽出

4.1 要件定義書からの項目抽出

情報システム開発で作成する設計書の記述方法は，開発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 3)4)．ここでは，論点を具体的にするため，図 2 に示した SQM-Trace のログオン機能についての設計書を対象とする．

要件定義書の記述方法として画面単位で要件を定義するものとする．図 5 は，ログオン機能の要件を記述したものである．要件定義書は 1 枚のシートに記述し，第 1 行目にはバージョン番号(Ver_ID)と設計書タイプ(DocType)，第 2 行目以降の第 1 カラムには要件の項目番号と項目名，次の行の第 2 カラムには要件の内容を記述するものとする．

図 6 は，図 5 の要件定義からトレース関連の定義に必要な項目を抽出し，SQM-Trace データディクショナリーのテーブル R_Titel_Tree に登録する処理の主要部を示している（構文は Visual Basic 風）．図 6 の 1～7 行目は，変数の宣言とデータベースへの接続処理である．8 行目はバージョン番号，9 行目は設計書タイプを抽出する処理である．10，11 行目と 26，27 行目でシートのすべての記述に対する処理ループを構成している．12～16 行目は，要件定義書の第 2 行目以降の第 1 カラムに記載された要件の項目番号と項目名を抽出している．17～19 行目は，項目の内容(C_Spec)と上位の項目番号(P_No)を設定している．20～24 行目は抽出した項目をテーブル R_Titel_Tree に登録する処理である．図 6 の処理により，「第 2 行目以降の第 1 カラムに要件の項目番号と項目名，次の行に要件の内容を記述する」というルールが守られていれば，要件の項目番号と項目名を網羅的に

0	要件定義
1. ログオン画面	
	・ログイン画面では，ユーザID，パスワード，プロジェクト名を入力する． ・ログオンが成功した場合，SQM-Trace初期画面に遷移する．
2. ログオン失敗画面	
	・ログオンが失敗した場合，ログオン失敗画面を表示し，ログイン画面への遷移を可能にする．
3. 期限切れ警告画面	
	・パスワードの期限切れ10日前から警告画面を表示する． ・警告画面からは，パスワード変更画面に遷移するか，SQM-Trace初期画面に遷移することをユーザが指定する．
4. パスワード変更画面	
	・新パスワードを2回入力する．
5. パスワード変更成功画面	
	・2つのパスワードが一致した場合，パスワードを更新し，パスワード変更が成功した画面を表示する． ・SQM-Trace初期画面への遷移を可能にする．
6. パスワード変更失敗画面	
	・パスワード変更に失敗した場合は，パスワード変更失敗画面を表示し，パスワード変更画面への遷移を可能にする．
7. SQM-Trace初期画面	
	・この画面についての詳細は別途定める．

図 5 要件定義書
Fig.5 Requirements

```
Dim Ver_ID,DocType,C_Title,C_No,C_Spec As String
Dim P_No, strSql, insSql, S1, S2 As String
Dim i, L As Integer
Dim adoCn As ADODB.Connection
Set adoCn = New ADODB.Connection
adoCn.Provider = "Microsoft.Jet.OLEDB.4.0"
adoCn.Open "Accessデータベースファイル名"
Ver_ID = Cells(1, 1)  'バージョンIDを取得
DocType = Cells(1, 2) '設計書の種別を取得
i = 2
Do
  S1= Cells(i, 1)
  If S1 <> "" Then
    L = InStr(S1, ".")
    C_No = Trim(Mid(S1, 1, L - 1)) '章の番号
    C_Title = Trim(Mid(S1, L + 1)) '章のタイトル
    i = i + 1
    C_Spec = Cells(i, 2)
    P_No = "0"
    insSql = "insert into R_Title_Tree " & _
      " values ('" & DocType & "','" & Ver_ID & _
      "','" & C_Title & "','" & C_No & "','" & _
      C_Spec & "','" & P_No & "')"
    adoCn.Execute insSql, , adCmdText
  End If
  i = i + 1
Loop While (Cells(i, 1)<>"" Or Cells(i, 2)<>"")
adoCn.Close
```

図 6 要件項目の抽出プログラム
Fig.6 Requirement item extraction program

抽出し，データディクショナリーに登録することができる．

4.2 外部設計書からの項目抽出

4.2.1 外部画面設計書からの項目抽出

個々の画面は外部画面設計書で定義する．外部画面設計書では，画面ごとに画面定義を 1 枚の Excel シートに記述するものとする．図 7 は，外部画面設計書の例であり，図 5 の「ログオン画面」に対応するものである．同様の記述方法で，図 5 に示したすべての画面を定義するが，紙面の都合で省略する．

外部画面設計書の目的は，各画面を構成する要素を定義することである．本研究では，画面を構成する要素として，出力データ，入力データ，ボタン名とボタン操作で実行される機能名

【画面名｜ログオン画面】

項目名	タイプ	項目の内容
【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)	“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、アプリ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 7 外部画面設計書　Fig.7 External screen design

対象とする．外部画面設計書の記述方法としては，設計項目を【<設計項目の種類>｜<設計項目名> 】でタグ付けして記述するものとする．設計項目の種類としては，画面名，出力，入力，ボタンおよび機能とする．例えば，画面名の場合，【画面名｜<画面名> 】でタグ付けする．

```
Dim J, K, LP1, LP2, LP3, LP4, MP, RP As Integer
Dim S1, S2, S3, S4, ZName As String
Dim OutD(20), InpD(20), BtnD(20), KinoD(20) As String
Dim OutDp, InpDp, BtnDp As Integer
For J = 1 To Sheets.Count
    OutDp = 0
    InpDp = 0
    BtnDp = 0
    Sheets(J).Select
    Cells(1, 1).CurrentRegion.Select
    RC = Selection.Rows.Count 'データの範囲を取得
    For K = 1 To RC
        S1 = Cells(K, 1).Value
        LP1 = InStr(S1, "【画面名｜")
        LP2 = InStr(S1, "【出力｜")
        LP3 = InStr(S1, "【入力｜")
        LP4 = InStr(S1, "【ボタン｜")
        MP = InStr(S1, "｜")
        RP = InStr(S1, "】")
        If LP1 > 0 Then
            ZName = Mid(S1, MP + 1, RP - MP - 1)
            ZName = Trim(ZName)  '画面名を取得
        ElseIf LP2 > 0 Then
            S4 = Mid(S1, MP + 1, RP - MP - 1)
            OutDp = OutDp + 1
            OutD(OutDp) = Trim(S4) '出力データ項目を取得
        ElseIf LP3 > 0 Then
            S4 = Mid(S1, MP + 1, RP - MP - 1)
            InpDp = InpDp + 1
            InpD(InpDp) = Trim(S4) '入力データ項目を取得
        ElseIf LP4 > 0 Then
            S4 = Mid(S1, MP + 1, RP - MP - 1)
            BtnDp = BtnDp + 1
            BtnD(BtnDp) = Trim(S5) 'ボタン項目を取得
            S3 = Cells(K, 4).Value
            LP5 = InStr(S3, "【機能｜")
            MP = InStr(S3, "｜")
            RP = InStr(S3, "】")
            S4 = Mid(S1, MP + 1, RP - MP - 1)
            KinoD(BtnDp) = Trim(S4) '機能項目を取得
        End If
    Next K
Next J
```

図 8 外部画面の設計項目抽出プログラム
Fig.8 External screen design item extraction program

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 9 設計項目抽出中間結果
Fig.9 Design items extracted

図 8 は，外部画面設計書から設計項目を抽出するプログラムの概要を示している．5～9 行目と 43 行目ですべてのシートを対象とした処理ループを定義している．10，11 行目では，シートに記載されているデータの範囲を取得している．12～42 行目までが，設計項目である画面名，出力データ，入力データ，ボタン名と機能名を抽出する処理

の概要である．一般に出力データ，入力データ，ボタン名と機能名は，複数定義されるため，出力データを配列 OutD，入力データを配列 InpD，ボタン名を配列 BtnD，機能名を配列 KinoD に記憶している．

図 9 は，図 7 の外部画面設計書に図 8 の処理を適用して抽出した項目に，章節項番号を付与したものである．このようにして抽出した設計項目を整形してテーブル R_Title_Tree に挿入し（図 6 の 20～24 行目と同様の処理），外部画面設計書からの設計項目の抽出処理を終了する．

4.2.2 画面フロー図からの項目抽出

SQM-Trace のログオン機能では，ボタン操作によって他の画面への遷移を行うものとする．図10は，図5の要件定義に対応する画面フロー図である．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10では，ボタン操作の結果に応じた分岐が発生する場合は，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．

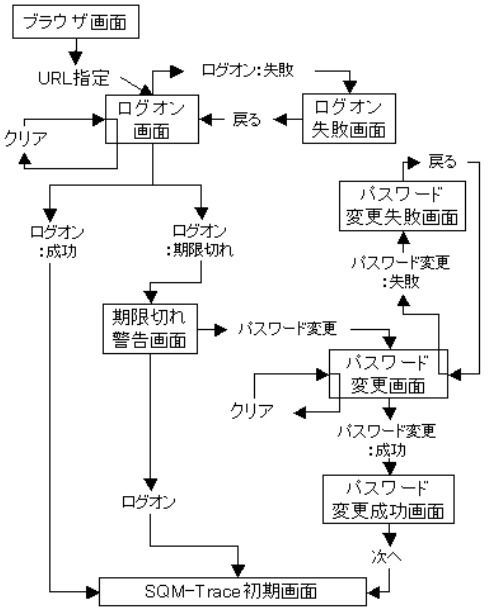

図 10 画面フロー図
Fig.10 Screen flow diagram

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	クリア
2.1.1	ログオン画面
2.2	ログオン
2.2.1	期限切れ警告画面←期限切れ
2.2.2	ログオン失敗画面←失敗
2.2.3	SQM-Trace初期画面←成功
3	ログオン失敗画面
3.1	戻る
3.1.1	ログオン画面
4	期限切れ警告画面
4.1	パスワード変更
4.1.1	パスワード変更画面
4.2	ログオン
4.2.1	SQM-Trace初期画面
5	パスワード変更画面
5.1	クリア
5.1.1	パスワード変更画面
5.2	パスワード変更
5.2.1	パスワード変更失敗画面←失敗
5.2.2	パスワード変更成功画面←成功
6	パスワード変更失敗画面
6.1	戻る
6.1.1	パスワード変更画面
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 設計項目抽出結果
Fig.11 Design items extracted

図 10 の画面フロー図は，画面を枠付きのテキストボックスで，また，ボタン名と処理結果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オートシェープのコネクターで結合している．従って，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．抽出プログラムは紙面の都合で省略する．

図 11 は，図 10 の画面フロー図から抽出した項目に，章節項番号を付与したものである．なお，図 11 では，後述する処理機能設計書の遷移先との整合性を保つため，図 10 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．このようにして抽出した設計項目を整形して，テーブル R_Title_Tree に挿入し（図 6 の 20～24 行目と同様の処理），画面フロー図からの設計項目の抽出処理を終了する．

4.3 内部設計書からの項目抽出

4.3.1 内部画面設計書からの項目抽出

図 12 は，図 7 に示した外部画面設計書に対応する内部画面設計書であり，フォームの定義，出力データ，入力データ，ボタンなどの主要な画面構成を HTML 言語によって記述したものである．図 13 は，図 12 の内部画面設計書から抽出した項目に，章節項番号を付与したものである．抽出プログラムは紙面の都合で省略する．

```
<title>ログオン画面</title>
<form method="post" action="/LogOnAction">
ユーザ名: <input type="text" name="UID" size="12">
パスワード: <input type="password" name="PWD" size="12">
プロジェクト名:<select name="PID">
[ <option value=%DB検索結果1%> %DB検索結果2% ]
</select><p>
<input type="submit" value="ログオン">
<input type="reset" value="クリア">
</form>
```

図 12 内部画面設計書
Fig.12 Internal screen design

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	UID
2.2.2	PWD
2.2.3	PID
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	/LogOnAction
2.3.2	ボタン機能2
2.3.2.1	クリア
2.3.2.2	HTML

図 13 設計項目抽出結果
Fig.13 Design items extracted

4.3.2 処理機能設計書からの項目抽出

図 14 は，図 7 に示した「ログオン」ボタンを操作することで起動される「ログオン処理」に対応する処理機能設計書である．この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．機能 ID を章に，Action Form を 1 節に，Action Servlet を 2 節に，遷移先を 3 節の項番号に割り当てている．図 15 は図 14 から抽出した項目に，章節項番号を付与したものである．

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。
・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にログインする。
・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"Scess"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14 処理機能設計書
Fig.14 Functional design

2	LogOnAction
2.1	LogOnForm
2.2	LogOnProcess
2.3	遷移先
2.3.1	Trace.do←Success
2.3.2	LogOnWarning.jsp←Warning
2.3.3	LogOnFail.jsp←Fail

図 15 設計項目抽出結果
Fig.15 Design items extracted

5. プロトタイプの実装

5.1 設計項目の木構造とトレース関連の定義

図 16 は，図 4 に示したトレース関連を図 3 のテーブル R_Trace_Schema に実装したものである．SC_NoPT と TC_NoPT カラムに記載されている"#" は任意の数字を示すワイルドカードであり，"#" は章番号を，"#.#" は章節番号を表している．図 4 で定義している 10 個のトレース関連を，テーブル R_Trace_Schema の 10 行のレコードで実装している．

Flag カラムは，トレース関連の定義方法を表現している．Flag カラムが "0" の場合は設計項目を表す文字列の一致に基づいてトレース関連を自動で定義することが可能であることを示している．トレース関連の自動設定は，トレース関連を定義しようとしている 2 つの設計書で同じ用語集に基づいて記述されている場合に可能であり，本文 4 章の事例では要件定義書，外部画面設計書および画面フロー図間のトレース関連が該当する．

Flag カラムが "1" の場合は手動でトレース関連を定義することを示している．本文 4 章の事例では外部設計と内部設計に関する設計項目間のトレース関連が手動による定義に該当する．一般に，外部設計と内部設計では，同じ設計対象に対し異なる用語が使われるため，設計者の判断に基づいてトレース関連を手動で定義する．

5.2 トレース関連の定義

5.2.1 トレース関連の自動定義

図 17 は，トレース関連を自動で定義する処理の概要を示している(構文は Java 言語風)．2 つの設計書間に定義されるトレース関連が 1 個の場合は，項目名称の文字列の一致でトレース関連を定義することができるが，図 4 の外部画面設計書から画面フロー図へのトレース関連のように，同一の設計書間のトレース関連が 2 個以上存在する場合は，上位のトレース関連の範囲内で下位のトレース関連を定義する必要がある．

SDocType	SVer_ID	SC_NoPT	SDesc	TDocType	TVer_ID	TC_NoPT	TDesc	Flag
要件定義	0	#	画面名	外部画面	0	#	画面名	0
外部画面	0	#	画面名	画面フロー図	0	#	画面名	0
外部画面	0	#.3.#.1	ボタン名	画面フロー図	0	#.#	ボタン名	0
外部画面	0	#	画面名	内部画面	0	#	画面ID	1
外部画面	0	#.1.#	出力データ名	内部画面	0	#.1.#	出力データID	1
外部画面	0	#.2.#	入力データ名	内部画面	0	#.2.#	入力データID	1
外部画面	0	#.3.#.1	ボタン名	内部画面	0	#.3.#.1	ボタンID	1
外部画面	0	#.3.#.2	機能名	内部画面	0	#.3.#.2	機能ID	1
外部画面	0	#.3.#.2	機能名	処理機能	0	#.3.#	機能ID	1
画面フロー図	0	###	画面名←処理結果	処理機能	0	#.3.#	画面ID←処理結果	1

図 16 テーブル R_Trace_Schema のレコード
Fig.16 Records of table R_Trace_Schema

```
1  変数 SDType, SVID, TDType, TVID(処理中の設計書名とバージョン)にNullを設定する;
2  変数 SNoPT, TNoPT(処理中の設計項目の検索パターン)にNullを設定する;
3
4  for(テーブルR_Trace_Schema の各レコード R について )
5  {
6    if( R.Flag の値が"0"である R.SDocType, R.SVer_ID, R.SC_NoPT,
7        R.TDocType, R.TVer_ID, R.TC_NoPT を検索する;)
8    { if(R.SDocType, R.SVer_ID, R.TDocType, R.TVer_ID が
9        SDType, SVID, TDType, TVID と一致する)
10     { SNoPTと R.SC_NoPT とをマージし,項目番号の検索パターンSNoPT を作る;
11       TNoPTと R.TC_NoPT とをマージし,項目番号の検索パターンTNoPT を作る;
12     } else {
13       SNoPTに R.SC_NoPTを, TNoPTに R.TC_NoPTを設定する;
14     }
15     for(R.SDocType, R.SVer_ID, SNoPTを満たすレコードRS をテーブル
16         R_Title_Tree から検索する)
17     { for(R.TDocType, R.TVer_ID, TNoPTを満たすレコードRT をテーブル
18         R_Title_Tree から検索する)
19       {  if(RS.C_Title == RT.C_Title)
20          { テーブルR_Trace に RS.DocType, RS.Ver_ID, RS.C_Title, RS.C_No,
21            RT.DocType, RT.Ver_ID, RT.C_Title, RT.C_No を登録する;
22          }
23       }
24     }
25   }
26 }
```

図17 トレース関連の自動定義プログラムの概要
Fig.17 Overview of automatic Trace-relation definition program

図 17 の 1,2 行目は処理中の設計書名とバージョン ID を記憶している変数を初期化している．4～7 行目はテーブル R_Trace_Schema に定義されているすべてのトレース関連の定義に対し，自動定義するものを検索している．8～11 行目はトレース関連の定義が同じ設計書名とバージョン番号に対し 2 個以上存在する場合の処理であり，上位の検索パターンをマージして，処理対象とする項目番号の検索パターンを生成している．12～14 行目は，新規の設計書名とバージョン ID に対してトレース関連が定義される場合の処理である．15～24 行目は，テーブル R_Title_Tree から設計書名，バージョン ID，項目

番号の検索パターンと設計項目名が一致するものを網羅的に検索し，テーブル R_Trace に登録する処理である．

5.2.2 トレース関連の手動定義

トレース関連の手動定義は，トレース関連を定義する 2 つの設計書を構成する項目を一覧表示した画面で実施する．図 18 は，外部画面設計書と内部画面設計書を構成する項目一覧を 2 個表示した画面である．この画面で，トレース関連を定義する設計項目を選択し，実行ボタンを押下することでレース関連を定義することができる．

5.3 トレース関連の表示

本研究では，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．本文で実装した要素の対応一覧は，次の手順で作成している．

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 19 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 19 では，「画面フロー図」に"∧" 印が付いていることから，画面フロー図に関するトレース関連がすべて定義されていることを示している．同様の方法でトレース関連の後方探索を実装できる．

図18 トレース関連の手動定義の例
Fig.18 Manual Trace-relation definition

図 19 前方探索の例
Fig.19 An example of forward trace

6. おわりに

トレーサビリティ管理は，品質向上に有効であるにもかかわらず，設計書の記述方法の多様性や大量の設計項目のために実務レベルでは定着していない状況にある．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手法と，トレース関連情報を管理するデータディクショナリーについて論じ，プロトタイプの実装によって，トレース関連の定義と表示に関する基本機能を検証した．

トレース関連の定義は設計項目名の一致を基準といているが，外部設計と内部設計のように，工程によっては同じ設計項目に対し異なる用語が使われることがあり，トレース関連を効率的に定義することを阻害する要因になっている．今後は，設計項目名の同一性を扱うための用語辞書に関する研究，テスト工程におけるトレーサビリティの管理方式などに関する研究を計画している．

参考文献

1) California Department of Transportation: Systems Engineering Guidebook for ITS, Ver. 2.0 , http://www.fhwa.dot.gov/cadiv/segb/index.htm (2007).
2) CMMI Product Team: CMMI for Development, Ver. 1.2 --- Improving processes for better products, CMU/SEI -2006-TR-008, Software Engineering Institute, Carnegie Mellon University, http://www.sei.cmu.edu/ (2006).
3) 平田: 良い外部設計書を書いていると自信を持って言えますか，日経 IT プロ, http://itpro.nikkeibp. co.jp/article/COLUMN/20080514/301678/ (2008).
4) 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).
5) 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).
6) Rilling J., Charland P., and Witte R.: Traceability in Software Engineering - Past, Present and Future, CASCON (Centre for Advanced Studies Conference) 2007 Workshop Report (2007).
7) Spanoudakis, G. and Zisman A.: Software Traceability: A Roadmap, Handbook of Software Engineering and Knowledge Engineering. World Scientific Publishing, pp.395-428 (2005).
8) 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).