# モジュール・テストツール
# ～Satsumas のご紹介

株式会社お台場システムズ

(C) 2005-2006 Odaiba Systems Corporation. All rights reserved

## 1　エビデンス

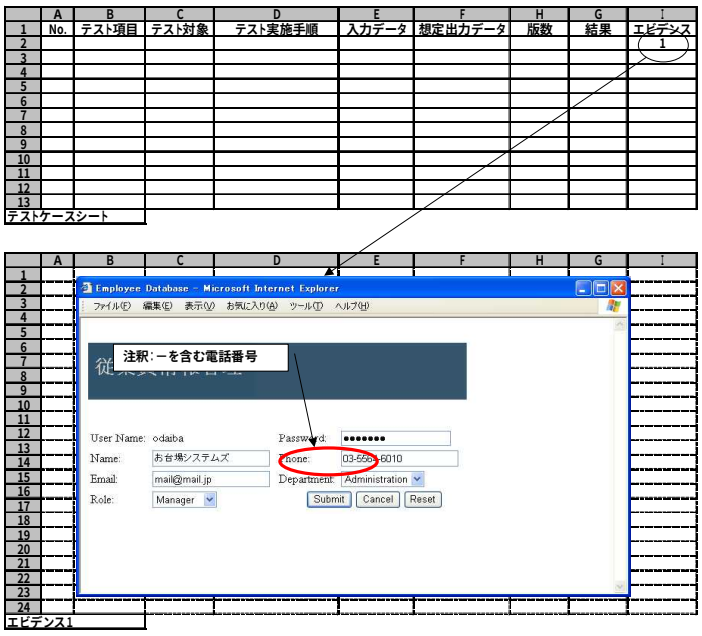
システム開発はますます低コストで短納期の傾向ですが、開発の品質管理については、より高品質を求める手間のかかる作業が増える傾向にあります。 ところで最近、“エビデンス(evidence)”って言葉をよく耳にするようになったと思いませんか。 もともと、“エビデンス”はセキュリティ関係の SE の間で、ハッカーによる不正アクセスや調査行為の有無の証拠（エビデンス）をとるなどの場合に使われる言葉でした。 ところが今、

表 1 これまでのテスト実施明細

| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| 1 | No. | テスト項目 | テスト対象 | テスト実施手順 | 入力データ | 想定出力データ | 版数 | 結果 |
| 2 | | | | | | | | |
| 3 | | | | | | | | |
| 4 | | | | | | | | |
| 5 | | | | | | | | |
| 6 | | | | | | | | |
| 7 | | | | | | | | |
| 8 | | | | | | | | |
| 9 | | | | | | | | |
| 10 | | | | | | | | |
| 11 | | | | | | | | |
| 12 | | | | | | | | |
| 13 | | | | | | | | |

テストケースシート

表 2 エビデンスありテスト実施明細




開発のそれもテストの現場でよく聞くようになりました。

つい数年前までのテストは、表計算ソフトにテストケース・テスト実施手順・想定出力とテスト実施結果が OK（PASS）だったのかＮＧ(FAIL)だったのか、○や×を記入したテストの実施明細を作成していました（表 1）。 ところが最近は、○×だけではなくて、テストを実施した際の証拠となるテストの画面コピーをエビデンスとして、テスト実施明細に貼り付ける事を義務づけする開発プロジェクトがでてきました。 エビデンスのとりかたも想定出力に対応する画面コピーだけでなく、テスト実施手順の画面コピーやその画面コピーに注釈を求めるプロジェクトもあります（表 2）。

## 2　開発工程とツール

ますます手間が増える一方のシステム開発の作業ですが、設計・実装・テストの開発工程の中でツールによって進化した工程があります。 それは、実装工程で、Eclipse に代表されるように統合開発環境(IDE)のツールを利用する事が当たり前になっています。 IDE ツールを使う以前は、コーディングとコンパイル作業は分かれている作業でした（C だと Make によるコンパイル作業）。 しかし、この分かれていた作業は IDE ツールの自動ビルド機能により、開発者がコーディングしている最中に裏でツールが自動でコンパイルしてくれるので、コーディング作業に集約されてひとつの作業になりました。 また、IDE ツールには、ソースコードを整形してプロジェクトのコーディングルールに合わせてコードのスタイルをフォーマットする機能を含め、エラー／警告機能もあり、コードレビューで見逃しがちな不要なコードや潜在的なバグに対して警告を出してくれます。 さらに、ソースコードの分岐やループによるコードの複雑さを数値化するメトリックス分析まで備えていたりします。 こうなってくると、コーディング以降のコードレビュー作業の作業内容も大きく変わってきます。 コードのスタイルや潜在バグなど IDE ツールが

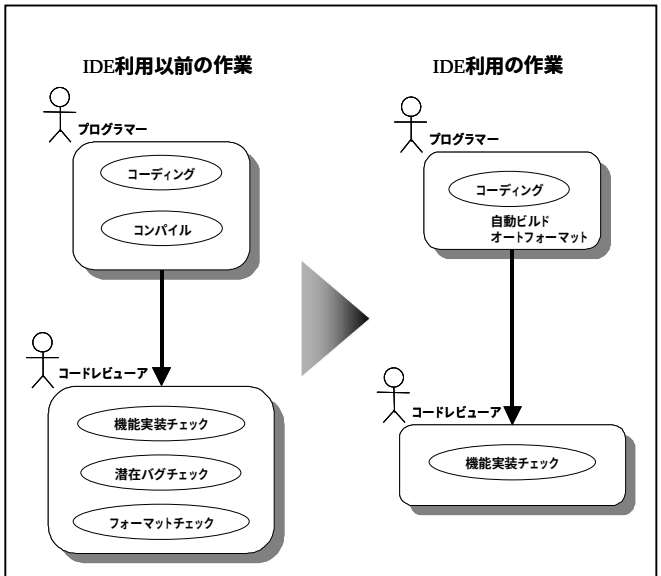



図 1 IDE利用前と利用後の作業の変化


(C) 2005-2006 Odaiba Systems Corporation. All rights reserved

自動で検査を行ってくれますからレビューアの役割は、設計仕様書の通りに確実に機能が実装されているかのレビューが中心となり、密度の濃いレビューを実施することができるようになりました（図 1）。

では、次工程のテストはどうなっているでしょうか。 最近は、テストファースト(テストありきで、まずテストからプログラムを書く)の eXtreme Programming 手法もでてきて xUnit によるテスティング・フレームワークを導入するところもありますが、 画面単体テストなどのモジュール・テストは、テスターがテスト仕様であるテストケースを作成⇒レビューアがテスト仕様をレビュー⇒テスターがテストを実施⇒再度レビューアがテストの実施を確認する昔ながらのプロセスです。

パフォーマンステスト、負荷テストやメモリリークテストなど人手でテストできない内容のものは専用のテストツールを使って行いますが、実装を検証する機能テストについてのテストツールは普及していません。 では、何故、機能テストツールは現場で普及していないのでしょうか。 いままで導入されてきた機能テストツールは、画面操作や実行後の出力結果と想定結果の突き合わせ手順を記録して、再度同じ手順でテストを実行する機能をもっていて、障害（バグ）や仕様変更があった場合に、再テスト（回帰テスト）をツールが自動実してくれるものです。 いままで導入されてきた機能テストツールによって作業の効率化は実現できましたが、実装工程で IDE ツール導入によるプロセスの変化が起こったような進化はありません。

機能テストツールは現場で普及しない理由を、設計・実装・テストの工程の作業特徴から検討してみます。 少々荒い表現ですが、設計工程は要求を具体化した設計書などの紙の資料を作成する作業であり、実装工程は紙の資料からコンピュータ言語にコード化

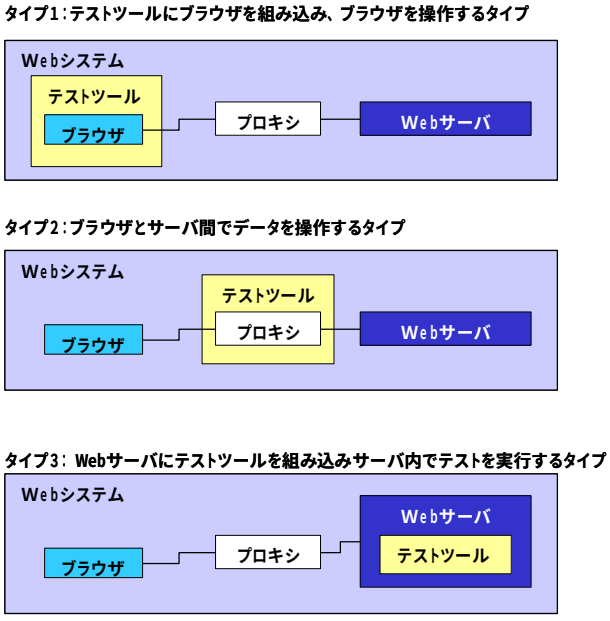



図 3 Web 系システムの機能テストツール

する作業となります。 IDE ツールは、厳密なコード化作業をサポートする自動化ツールで、厳密な作業を得意とするコンピュータのツールとしての持ち味が生きています。 では、テスト工程はどうかと

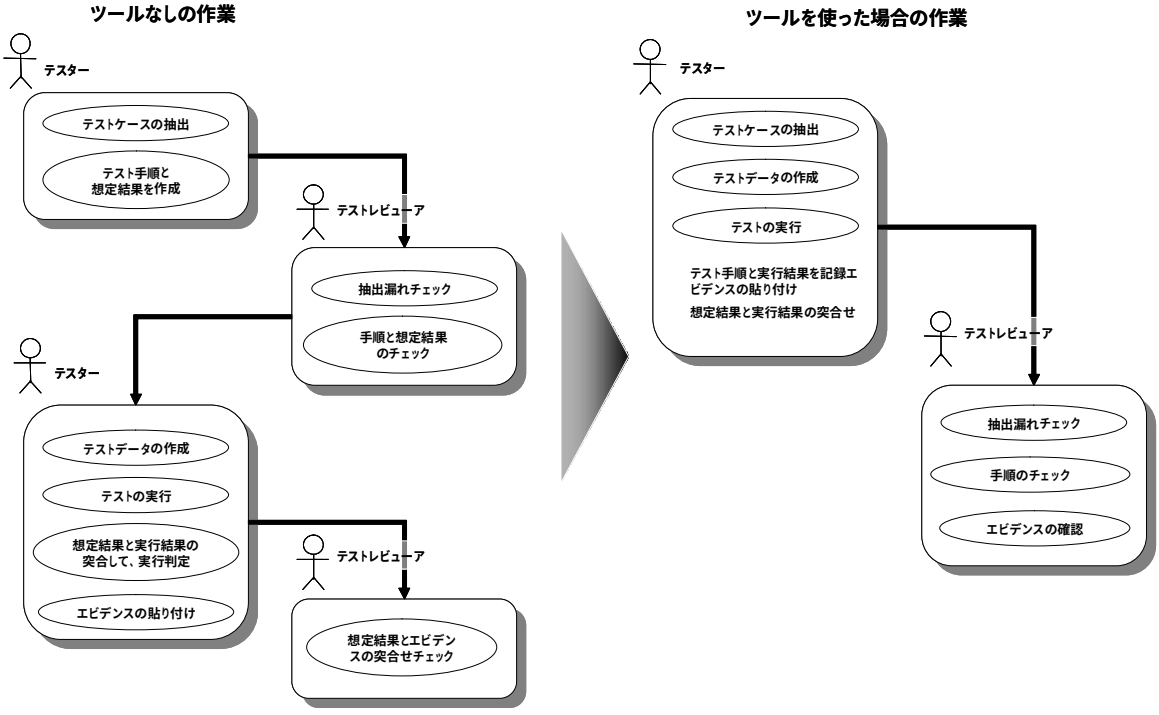



図 2 テストツールを使った場合のテスト作業


(C) 2005-2006 Odaiba Systems Corporation. All rights reserved

いうと、具体化された設計書の通りにシステムが動作してくれることを検証する作業となります。 テストの実行手順と想定した出力結果を作成するテストケースの作成作業、実際にテストを実施しシステムが出力する結果と想定結果の突き合わせ作業、また、その時の実行手順とシステムの出力結果をエビデンスとして保存する作業を通しで全てサポートしてくれる機能テストのツールがあれば、テスターの作業効率が格段に向上します。 そうなれば、従来テスターが行ってきた個々の作業が軽減され、IDEツールを導入した実装工程と同じように作業単位を集約することができ、テストのプロセスを変えることができます（図 2）。 また、テスターとレビューア間のパスも減り、作業の滞留を減らす効果も期待できます。

実際の作業でレビューアが一番困るケースは、テスターがテストを実施した際にバグが見つかり、ソースコードを修正して再テストを行い結果 OKなったテストケースが複数ある場合です。 テスト実施明細には、結果OKとなった際のソースコードの版数（Version）を記入してありますが、古い版のケースを再テストすべきか判断に困ります。 その時は、“怪しいものは、再テスト”に従って、全件再テストをテスターにリクエストします。 その時に、「テストツールがあれば遠慮なく再テストしてくださいと言えるのに」と思う方も多いではないでしょうか。 実装工程で IDE ツールが普及した次は、テスト工程の機能テストツールがブレイクするかもしれません。

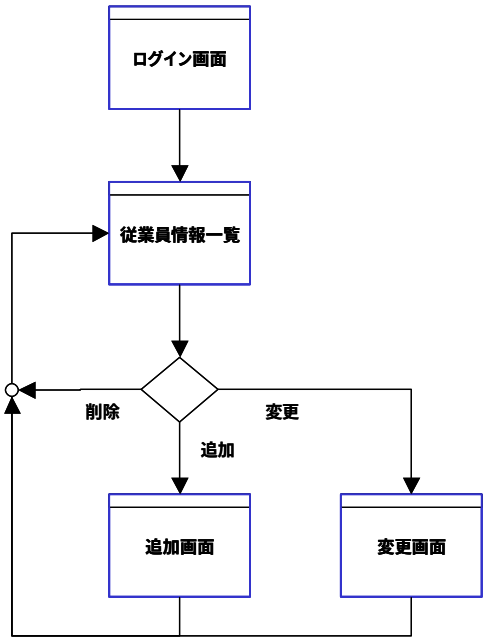



図 4 従業員情報管理システム

## 3 Satsumas とは

Satsumas は、当社お台場システムズが独自に開発した Web アプリケーション専用のテストツールです。 モジュール（単体）・テストにおいて、テスターがブラウザから入力する作業の手間と想定結果の確認とエビデンスの保存作業をサポートします。 この Satsumas は、パッケージ販売形式での提供方法とは異なり、Satsumas を搭載したノート PC のレンタル方式で提供しています。 機器レンタル方式は、テストツールのパッケージのインストール作業が不要になり、インストール作業での手間をかけずにテスト環境を準備することができます。 また、Satsumas は、複雑な GUI アプリケーションではありません。 マニュアルを読まなくても直感的にテストが実行できるように単純な Web アプリケーションですから導入までの教育や時間がかかりません。 さらに、テストに必要な期間だけの利用料金を支払うだけので、高価なテストツールのパッケージ購入に比べて、格段にコストを抑えることができます。

### 3.1 テストツールの種類

Satsumas 以外にも Web 系システムの機能テストツールは、フリー（Selenium,Solex など）や商用でいくつかあります。Web 系テストツールのタイプは3つのタイプがあります（図 3）。Satsumas は、タイプ2のブラウザとサーバの間でプロキシの機能を果たしながら HTTP データをキャプチャするキャプチャリ形式のテストツールで、キャプチャした HTTP データを操作・加工して、画面単位の入力テスト,機能テスト,脆弱性テストと複数の画面を遷移するシナリオテスト機能があります。

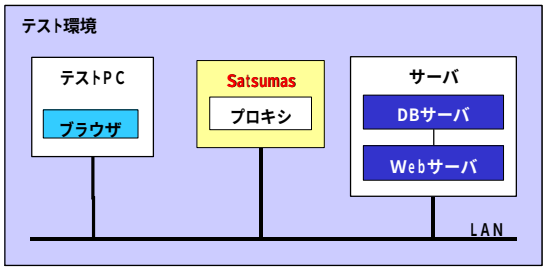



図 5 テスト環境の構築

## 4 Satsumas によるテスト

Satsumas によるテストの手順をわかりやすく説明するために、あるシステム開発のテストを例にします。 このシステムは、小規模な会社の従業員情報を管理することを目的とし、システム画面はログイン・従業員一覧・追加・更新の 4 画面で構成します（図 4）。 このシステムは社外からのアクセス利用を想定し


(C) 2005-2006 Odaiba Systems Corporation. All rights reserved

たシステムであり、セキュリティ性を考慮した、クライント上で動作する JavaScript などでのロジックを実装しないで全てサーバ側で処理する方針のシステムです。

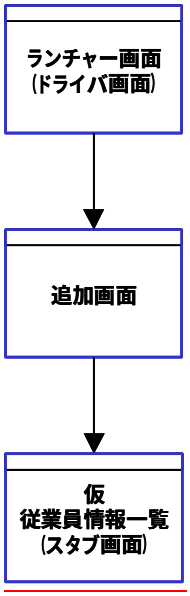



図 6 追加画面のテスト

モジュール(単体)・テストチームは、設計が終わった段階でテストケースを作成し、実装工程からあがってきたものを順にテストしていきます。 テストチームは、設計仕様書からシステム仕様や画面仕様の情報を入手し、実装チームから上がってくる画面のテストをするための、テスト準備をします。

### 4.1 スキャン

最初の作業は、テスト環境の構築です。 テスター用の PC とテスト対象のサーバにアクセス可能なネットワークに Satsumas を設置し、テストＰＣのブラウザの設定で、プロキシサーバ設定を Satsumas 経由にします(図 5)。


図 7 追加画面操作のスキャン


図 8 画面仕様の設定画面

表 3 追加画面の入力項目仕様

| No | 入力項目 | 必須 | 長さ(文字) | 文字種 | 半角・全角 | 値の範囲、例 | 補則 |
|---|---|---|---|---|---|---|---|
| 1 | ユーザID | ○ | 8～12 | 英数のみ | 半角 | | |
| 2 | パスワード | ○ | 6 | 英数のみ | 半角 | | |
| 3 | 従業員名 | ○ | 100 | 全ての文字 | 半角・全角 | | |
| 4 | 電話 | ○ | 15 | 数値と一部の記号 | 半角 | 0-9, - | |
| 5 | メールアドレス | ○ | 30 | 英数と一部の記号 | 半角 | a-z, .@_ | |
| 6 | 部署 | ○ | 1 | 数値 | | 管理部=1<br>経理部=2<br>営業部=3<br>技術部=4 | 管理部のみ 管理職あり |
| 7 | 役職 | ○ | 1 | 数値 | | 管理職=1<br>一般職=2 | |


(C) 2005-2006 Odaiba Systems Corporation. All rights reserved

一から行う開発を行うスクラッチ開発の場合、システムの全ての画面が実装された後でテストチームにプログラムが渡されることは稀です。 通常は、であがった画面から順次テストして、開発期間を節約します。 ここでは、最初にできあがってくる画面を追加画面とします。 テスト対象画面以外の画面が実装中の場合などは、テストドライバーのランチャー画面とテスト対象の次画面のスタブ画面 (ここでは、従業員情報一覧画面) をテストのために別途作成します (図 6)。 次に、追加画面を操作して、ブラウザと Web サーバ間の送受信データを Satsumas でスキャンします。 スキャンを実行するには、スキャン画面を表示して開始ボタンを押下します (図 7)。 このスキャン状態で画面を操作すると、Web サーバ間の送受信データを Satsumas がスキャンします。スキャンした情報は、「スキャンした画面の再表示」をクリックすると一覧を表示します。 追加画面の操作のリンクをクリックすると、追加画面の入力項目情報を表示します (図 8)。 表示された入力項目情報に対して、追加画面の入力項目の属性仕様書 (表 3) から入力項目の属性を登録します。

### 4.2 入力テスト

入力テストは、テスト対象の画面に入力項目がある場合には必ずテストしなければなりません。 システムを利用するユーザが入力項目に投入する値はありとあらゆる値があり、値の有効性処理が設計仕様通りに実装されていなければなりません。 もし、システム運用後にユーザ側で入力の不具合がみつかれば、システムを利用するユーザの信頼を一気に落とすことに成ります。 この入力テストは、漢字やカナなどの文字種に加えて半角･全角の区別や文字列長の組み合わせを考慮したテストケースを実行する必要があり、大変手間がかかる作業です。 この面倒な入力作業に対して、Satsumas の入力テスト機能は、1 つ1つの入力項目の属性情報からテストケースの作成～テストの実行～テストレポート作成を自動で行います。 自動で作成したテストケースは、文字列長の境界値の長さと半角･全角の漢字･カナ･記号･英字･数字の組み合わせとなります。

表 3 のユーザIDの項目では、文字列長が 8,9,11,12 文字でかつ半角の英字または数字の正常な入力のテストデータを自動生成して、Web サーバへ要求メッセージを送信します (図 10)。 さらに、文字列長が 0,7,13 ととても長い文字列長でかつ半角のカナか記号または全角文字の入力のテストデータを自動生成して、Web サーバへ要求メッセージを送信します。 テスト実施結果の明細は、Excel 形式のファイルで出力でき、テスト実施明細の作成作業を省力化できます。


図 9 機能テストケースの編集画面

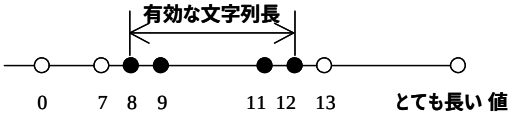



図 10 ユーザIDの文字列長の境界値テスト

その他に Sastumas の入力テスト機能は、基本的な脆弱性テストの機能があります。 脆弱性テストは、クロスサイトスクリプティング･SQL インジェクションや OS,Web サーバの製品名･バージョン情報･HELP またはサンプルの参照･強制的ブラウジングなど、技術雑誌や本などで解説されている基本的なセキュリティのテストを行います。 特にクロスサイトスクリプティング対


(C) 2005-2006 Odaiba Systems Corporation. All rights reserved

策でサニタイジングされていない画面表示の箇所を全て探す作業は人手ではかなり難しい作業ですが、これも自動で行います。

Satsumas による入力テストは、殆どが自動化されているので、例題の追加画面だと約 1 時間で全ての入力項目の入力テストを完了することができます。

### 4.3 機能テスト

機能テストは、ブラウザ上の送信（サブミット）ボタンやリンクのクリックによるイベント処理に対して、Web サーバの実行処理と処理後の応答画面の表示処理が設計仕様通りに実装されているかを検証するテストを行います。

ここも先の追加画面を例に説明します。 機能テストケースは、従業員情報が登録できる処理を検証するテストと登録できない処理を検証するテストを作成します。 登録できる処理を検証するテストは、新しい従業員を登録することにします。 登録できない処理を検証するテストは、すでに登録済みの従業員を再登録し、エラーメッセージを表示させることにします。 （データ制約・データオバーフロー・複数ユーザによる同時アクセスや機器障害など、機能テストで行うべきテスト項目についてはここでは省きます）

テストを実行するには、投入するテストデータやテストの動作環境を検討する必要があります。 登録できる処理を検証するためのテストの場合、テストを実行する前に予め登録する予定の従業員情報を DB 上から削除しておかないと、重複エラーとなり、登録できない処理を検証するテストになってしまいます。 そこで、テスト実行前の環境設定の手順に、予め登録する予定の従業員情報を DB 上から削除してからテストを開始するようにします。 次に、テストで投入する登録する予定の従業員情報を決めます。 最後に、テスト実行後の判定方法を決めます。 判定は、DB に従業員のレコードが登録されていることと次画面の従業員情報一覧画面が表示されることが OK の条件です。 ツールを利用するとテストの判定条件を厳密に定義しなければならないので、テストの目的と判定条件が明確化し、テストの質を高めることができます。

では、実際に従業員情報が登録できる処理を検証するテストケースを登録してみます。

テストケースの登録は、Satsumas の機能テストケースの編集画面から①テストケース名、②テスト目的を明記したテスト内容、③テスト環境設定、④テストデータ、⑤テスト判定方法の 5 つの項目を設定します。 ①テストケース名には、“E31 追加画面から管理部の管理者を登録”とします。 ②テスト内容には、テストレビュー者がテストケースをレビューし易いように以下の記述を設定します。

[1.テスト目的]

新しい従業員が登録できること。 管理部の管理者の登録。

[2.画面投入データ]

ユーザ ID=100、パスワード=password・・・部署=1・役職=1

[3.テスト環境]

テスト実行前は、DB 上にユーザ ID=100 のユーザが存在しないこと

[4.テスト判定方法]

(1)DB の従業員テーブルにレコードを追加

- 登録データの内容が各属性に登録される事
- 文字化け、文字落ち、余分な文字の付加がない事

(2)次画面で従業員情報一覧画面を表示

- 画面のタイトルが次画面である事

③テスト環境設定には、DB 上にユーザ ID=100 のユーザが存在しないように、“DB 上のユーザ ID=100 のユーザを削除す

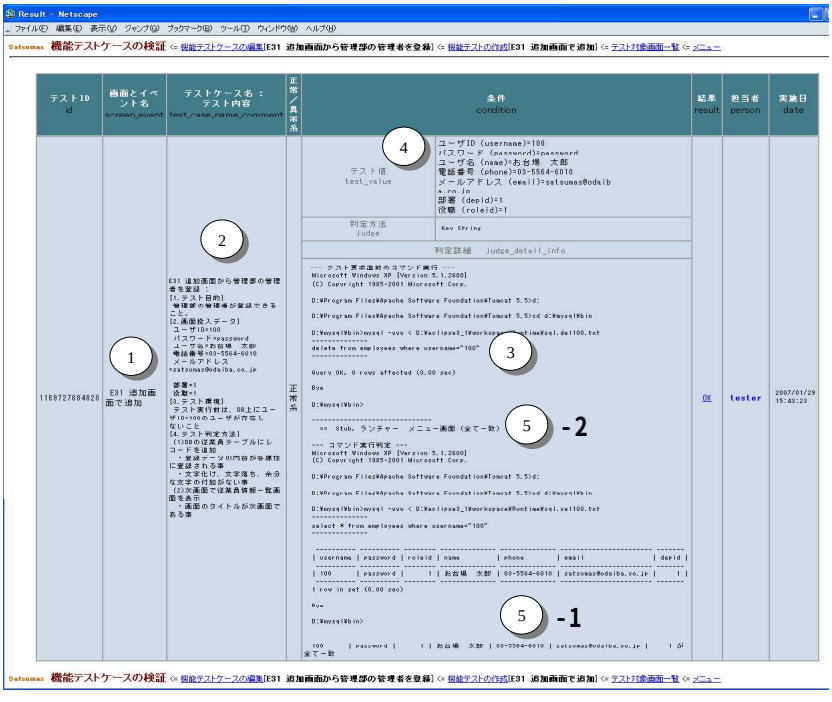



図 11 機能テストケースの検証画面


(C) 2005-2006 Odaiba Systems Corporation. All rights reserved

る DB 操作の処理スクリプト”を記述します。 Satsumas は、「**テストするためのプログラムは作成しない**」方針ですが、テスト環境設定や実行後の判定の際に DB 操作など汎用的なスクリプト操作を可能にするため、コマンドラインを実行できるようになっています。 ④テストデータには、スキャンした時の要求メッセージを元に表示された入力項目に対して設定していきます。ユーザ ID=100・パスワード=password・ユーザ名=お台場 太郎・電話番号=03-5564-6010・メールアドレス=satsumas@odaiba.co.jp・部署=1・役職=1 を設定します。 ⑤テスト判定方法には、”DB の従業員テーブルにレコードが追加されたことを確認する DB 操作の処理スクリプト⑤-1”を記述します。 さらに、次画面の表示となる応答メッセージの HTML のタイトルが従業員情報一覧画面になっているか設定します。 ここでは、次画面の従業員情報一覧画面の作りこみがまだという状況なので、仮従業員情報一覧（スタブ画面）の表示を暫定で確認することとします(⑤-2)（図 9）。

テストケースを登録したら確認のために、テストを実行します。機能テストケース実行後の検証画面にある結果 OK のリンクをクリックすると応答画面を別ウィンドウで表示し、テストデータを送信した結果の次画面の表示を確認することができます（図 11）。 機能テストの実行結果は、Excel ファイルに出力できるので、テスト結果の詳細な報告書を作成する手間を省力化できます。 また、エクスポートした Excel ファイルを直接編集し、テストケースの作成・加工が可能ですので、テストケースの作成効率だけでなくテストケースの管理工数を大幅に削減できます。

### 4.4 シナリオテスト

Satsumas のシナリオテスト機能は、テストシナリオに従って画面が遷移した事を検証することができます。 予め作成したシナリオに従って画面を操作すると、Satsumas がキャプチャしてシナリオテストとして登録します。 登録したシナリオテストは、シナリオテスト実行のエビデンスとすることができます。また、登録したシナリオテストをツールに実行させることで、障害対応や仕様変更後の回帰テスト作業を効率的に実行できます。 同じ様な画面遷移操作のテストを行う場合には、登録済みのシナリオを Excel ファイルにエクスポートし、Excel ファイルを直接修正した後にインポートさせることで、別のシナリオを簡単に作成できます。 また、あるシナリオの一部の操作繰り返しさせる時には Excel ファイルの内容を修正し、別のシナリオの一部をマージすることでシナリオの変更も容易にできます。 たとえば、登録部分の操作を 100 回繰り返すようなテストを作成するもの簡単にできます。

シナリオテストの登録は、①シナリオ名 ②シナリオテストの目的を明記したシナリオ詳細 ③テスト環境設定 ④各画面への遷移判定条件の 4 項目を設定します。 ここでは、管理者がログインして、一般社員を登録するテストシナリオを例に①～④の 4 項目の設定例を紹介します。

①シナリオ名には、“S21 ログインし追加画面で追加”とします。 ②シナリオ詳細には、テストレビュー者がテストケースをレビューし易いように以下の様に記述します。

[1.テスト目的]
管理者がログインし、新入社員の情報を登録
[2.画面投入データ]
管理者のユーザ ID=100
新入社員のユーザ ID=102
[3.テスト環境]
テスト実行前は、DB 上にユーザ ID=102 のユーザが存在しないこと
[4.テスト判定方法]
(1)ログイン画面の表示
(2)従業員一覧の表示
(3)追加画面の表示
(4)従業員一覧の表示
・従業員一覧に新入社員のユーザ ID が表示されること


(C) 2005-2006 Odaiba Systems Corporation. All rights reserved


図 13 シナリオテスト画面

念が一般に浸透していて、システム開発の予算縮小にも少なからず影響を及ぼしているように思われます。 だからと言って言い訳にはなりませんが、テストの飛ばしやテスト作業が不十分という状態が発生しているのではないでしょうか。最近では金融機関の合併にともなうシステム統合での問題発生など、「品質管理」や「テスト」という言葉が一般のニュースで流れています。 システムを発注する側も作る側もテストの重要性を再認識したと思います。 その中で、システム開発のテスト工程では、バグを減らして品質をより向上させ、さらに、時間と人を減らす方法としてテストツールの利用の検討が進んでいます。ここで、ご紹介した Satsumas は、モジュール(単体)・テストで利用するテストツールです。 バグを減らして品質をより向上させ、さらに、テストのプロセスをも変化させるツールを目指して開発しています。

・DB 上にユーザ ID=102 のユーザが存在する事

③テスト環境設定には、DB 上にログインする管理者ユーザ ID=10 のユーザが存在し、登録予定の一般社員ユーザ ID=102 が存在しないように、“DB 操作の処理スクリプト”を記述します。 ④各遷移画面の表示確認には、遷移した画面のタイトルが遷移予定画面のタイトルであるかを判定条件とする設定を行います。 さらに、新入社員がＤＢに追加されたかどうかを確認する DB 操作の処理スクリプトを記述します(図 13)。

シナリオテストのテスト実行結果は、設定したスクリプトや遷移画面の判定毎に OK/NG で表示します(図 12)。


図 12 シナリオテスト画面

## 5 さいごに

冒頭でも述べましたが、開発期間の短縮による短納期になっています。 特にインターネットの世界では、情報・コンテンツ・ソフトウェアなどがフリー＝タダという観


(C) 2005-2006 Odaiba Systems Corporation. All rights reserved