Borland®

Silk Performer 16.5

Web 負荷テスト チュートリアル

Borland Software Corporation
700 King Farm Blvd, Suite 400
Rockville, MD 20850

2015-09-10

目次

Web 負荷テスト チュートリアル

このチュートリアルは、Silk Performer を使用して、Web アプリケーションの負荷テストを実行し、できる限り迅速に稼働させるのに役立ちます。 Silk Performer の利点をフルに活用して、最先端の機能を簡単に使用できるようにします。

Web 負荷テストの概要

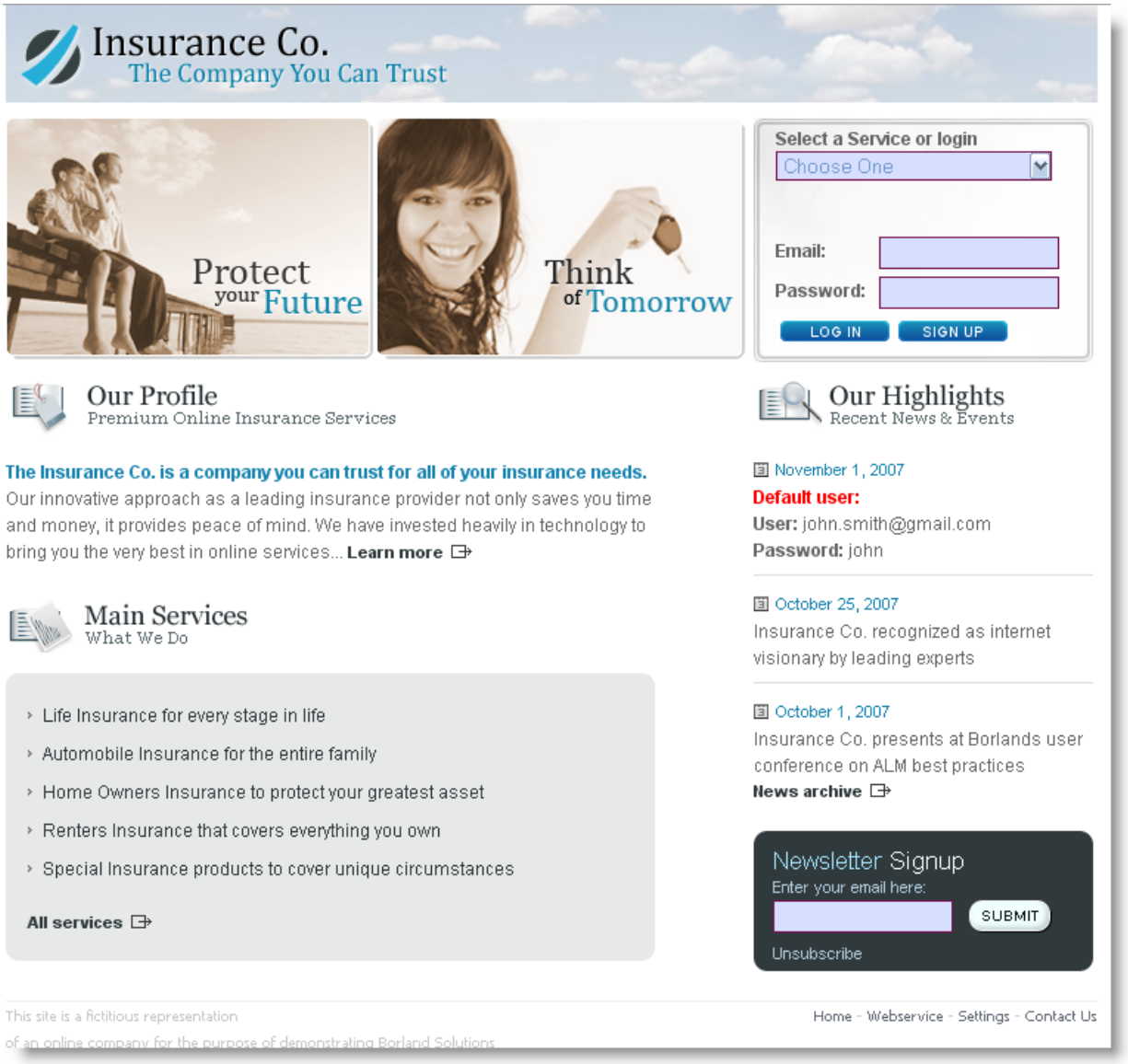
今日の最新の Web アプリケーションをテストする最も早くて簡単な方法は、プロトコル レベル (HTML/HTTP) でのテストを行うことです。 この方法では、高度な機能が組み込まれた簡単なスクリプトが生成されます。 Web アプリケーションを組み込み AJAX ロジックでテストする際に、プロトコル レベルでのテストが失敗であると判明した場合は、ブラウザ駆動型 Web 負荷テストを使用することをお勧めします。

Silk Performer を使用すると、今日の最新の Web アプリケーションのテストをプロトコル レベル (HTTP) で簡単に行えるだけでなく、実際の Web ブラウザ (Internet Explorer) を使用して負荷を生成することができます。 これにより、Web アプリケーションに組み込まれた AJAX ロジックを利用して、複雑な AJAX 動作をテスト中に正確にシミュレートすることができます。 この強力なテスト方法では、レンダリング時間やプロトコル レベルの統計など、エンド ユーザーによる実際のブラウザ動作を反映する結果が提供されます。

サンプル Web 2.0 アプリケーション

Silk Performer には、Web 2.0 アプリケーションのテストを習得するために使用できる最新のサンプル Web アプリケーションが備えられています。 InsuranceWeb サンプル Web アプリケーションは、ExtJS および JSF フレームワークに基づいて作成され、AJAX 技術を採用し、JSON と XML を介して通信します。

サンプル アプリケーションは、*http://demo.borland.com/InsuranceWebExtJS/* でホストされます。



Web 負荷テストプロジェクトを定義する

1. Silk Performer ワークフロー バーの **ここから開始する** をクリックします。

   注: 別のプロジェクトが既に開いている場合に、メニュー バーから **ファイル** > **プロジェクトの新規作成** を選択すると、現在開いているプロジェクトを閉じるように確認メッセージが表示されます。

   ワークフロー - プロジェクトの概要設定 ダイアログ ボックスが開きます。
2. **名前** テキスト ボックスに、プロジェクトの名前を入力します。
3. オプションのプロジェクト説明を **説明** に入力します。
4. **種類** メニュー ツリーで、**Web business transaction (HTML/HTTP)** を選択します。
5. **次へ** をクリックし、設定に基づいてプロジェクトを作成します。

ワークフロー - スクリプトの作成 ダイアログ ボックスが表示されます。

テスト スクリプトを作成する

テスト スクリプトを作成する最も簡単な方法は、Silk Performer Recorder を使用することです。Recorder は、トラフィックを測定して記録し、テスト スクリプトを生成する Silk Performer のエンジンです。

テスト スクリプトを記録する

1. ワークフロー バーの **スクリプトの作成** をクリックします。 **ワークフロー - スクリプトの作成** ダイアログ ボックスが表示されます。
2. 記録に使用するブラウザに応じて、**アプリケーション プロファイル** リストに一覧表示されたブラウザから 1 つを選択します。
3. **URL** フィールドに、記録する URL を入力します。

   注: InsuranceWeb サンプル Web 2.0 アプリケーションは、*http://demo.borland.com/InsuranceWebExtJS/* から入手できます。 **Select a Service or login** リストでは、Auto Quote および Agent Lookup サービスをテストに使用することができますが、リストに示されたそれ以外のサービスには機能がありません。
4. **記録の開始** をクリックします。 [Silk Performer Recorder - [<プロファイル名>]] ダイアログが最小化されて表示され、クライアント アプリケーションが起動されます。
5. 記録中に発生したアクションのレポートを表示するには、**GUI サイズの変更** をクリックして [SilkPerformer Recorder - [<プロファイル名>]] ダイアログを最大化します。[SilkPerformer Recorder - [<プロファイル名>]]ダイアログが最大化され、**アクション** ページに表示されます。
6. クライアント アプリケーションを使用して､テストでシミュレートするターゲット サーバーでの操作などを実行します。この操作を、Recorder が取得して、記録します。これらのアクションとダウンロードされたデータのレポートが、**アクション** ページに表示されます。
7. 記録を終了するには、**記録停止** をクリックします。
8. .bdf ファイルの名前を入力し、保存します。テスト スクリプトが問題なく生成された場合は、スクリプトが自動的に Workench に表示されます。追加の設定が必要な場合は、**キャプチャ ファイル** ページが表示されます。そして、キャプチャ ファイルからスクリプトを生成できます。

スクリプトの試行

テスト スクリプトを生成したら、スクリプトの試行を行って、スクリプトがエラーなしで動作するかどうかを確認します。 スクリプトの試行を行うことによって、Recorder で記録された操作がスクリプトで正確に再現されるかどうかがわかります。 また、スクリプトがエラーなしで動作できるようにするために、パラメータ化を必要とするコンテキスト固有のセッション処理がスクリプトに含まれているかどうかもわかります。

スクリプトの試行では、1 人の仮想ユーザーのみが実行され、スクリプト化されたアクション間の思考時間遅延が発生しない ストレス テスト オプションが有効になります。

テスト スクリプトを試行する

1. Silk Performer のワークフロー バーの **スクリプトの試行** をクリックします。 **ワークフロー – スクリプトの試行** ダイアログが表示されます。
2. **スクリプト** リスト ボックスでスクリプトを選択します。
3. **プロファイル** ボックスの一覧には、現在アクティブなプロファイルが選択されています (別のプロファイルを構成したのでなければ、これがデフォルトのプロファイル)。
   a) 選択したプロファイルのシミュレーション設定を構成するには、このリスト ボックスの右側にある **設定** をクリックします。

b) プロジェクト属性を構成するには、**プロジェクト属性** リンクを選択します。

4. ユーザー グループと仮想ユーザーの **ユーザーグループ** 一覧から、実行する仮想ユーザーのユーザー グループを選択します。

   スクリプトの試行なので、実行される仮想ユーザーは 1 ユーザーのみです。

5. スクリプトの試行中に Web サーバーからダウンロードされた実際のデータをリアルタイムに表示するには、**TrueLog Explorer によるアニメーション実行** チェック ボックスをオンにします。

   Web アプリケーション以外のテストを実行している場合は、このオプションを無効にしてください。

6. **実行** をクリックします。 スクリプトの試行が開始されます。

スクリプトの試行の実行中、記録された思考時間はすべて無視されます。 **監視** ウィンドウが開き、スクリプトの試行の実行の進行状況について詳細な情報が表示されます。 **アニメーション** オプションをオンにしている場合は、TrueLog Explorer が開きます。 ここには、スクリプトの試行の実行中にダウンロードされた実際のデータが表示されます。 スクリプトの試行の実行中にエラーが発生した場合は、TrueLog Explorer を利用して、すばやくエラーを検索して、セッション情報をカスタマイズできます。 TrueLog Explorer でスクリプトの評価とカスタマイズを完了すれば、このスクリプトはエラーなしで実行できます。

テスト スクリプトを分析する

Silk Performer には、スクリプトの試行の実行後にテスト スクリプトを評価するための手段がいくつか用意されています。

TrueLog Explorer による視覚的分析

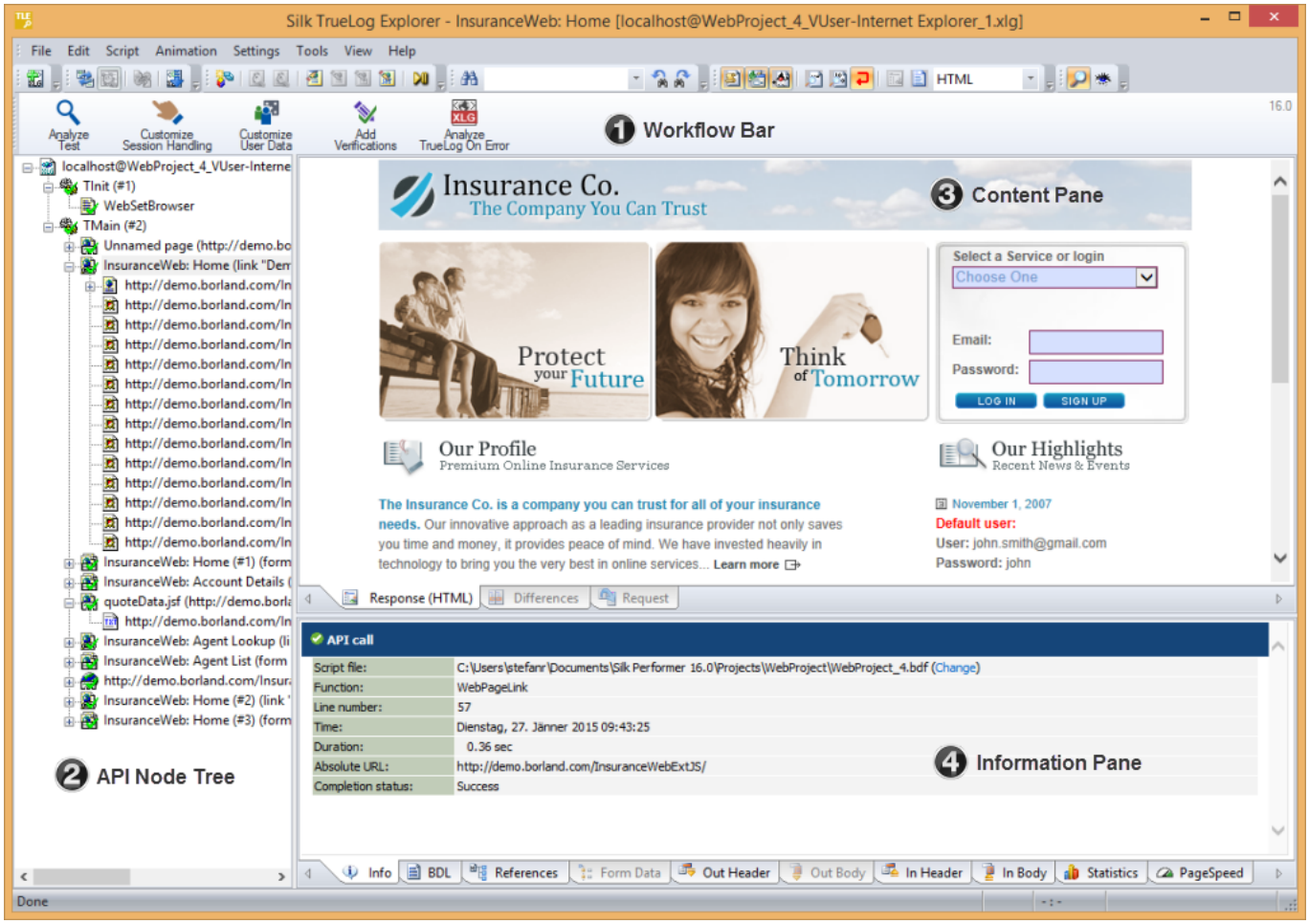
TrueLog Explorer の最も強力な機能の 1 つは、テスト中のアプリケーションによって表示された Web コンテンツを視覚的にレンダリングする機能です。 実際に、仮想ユーザーがアプリケーションとやり取りをするときに見るものが表示されます。

TrueLog Explorer のインターフェイスは、次のセクションから構成されています。

- **ワークフロー バー** は、TrueLog Explorer を扱う際の主要なインターフェイスとなります。 ワークフロー バーは、TrueLog Explorer に組み込まれているテスト方法論を元に、5 つの主要な作業をサポートしています。
- インターフェイスの左側にある **API ノード ツリー** メニューでは、負荷テスト中にダウンロードされた TrueLog データを展開または縮小できます。 ロードされている TrueLog ファイルはそれぞれ、関連するすべての API ノードへのリンクと共に、ここに表示されます。 ノードをクリックして、**画面** ペインにスクリーンショットを表示し、**情報** ビューに履歴詳細を表示することができます。
- **コンテンツ** ペインでは、受信したすべてのデータに対して複数のビューを利用できます。
- **情報** ペインには、テスト スクリプトおよびテスト実行に関するデータが表示されます。読み込まれた TrueLog ファイル、選択された API ノード、 BDL スクリプト、統計に関する一般情報が含まれます。

注: HTTP ヘッダー データは、現時点では使用できません。

注: Silk Performer から TrueLog Explorer を起動するには、**結果** > **TrueLog の検討** を選択します。



テスト実行を分析する

1. スクリプト試行によって TrueLog が TrueLog Explorer に読み込まれた状態で、ワークフロー バー の **テストの分析** ボタンをクリックします。
   テストの分析 ダイアログ ボックスが表示されます。
2. 次のいずれかのオプションで操作を続行します。
   - 仮想ユーザー要約レポートの表示
   - TrueLog 内のエラーの検索
   - 再生テスト実行と記録テスト実行の比較

仮想ユーザー要約レポートを表示する

仮想ユーザー要約レポートは、個々のスクリプト試行について要約したレポートで、基本情報とタイミング平均を含みます。 それぞれのレポートは 1 人の仮想ユーザーについて作成されます。 データは表形式で表示されます。

注: 仮想ユーザー要約レポートの処理にはかなりの時間がかかるので、デフォルトでは生成されない設定になっています。

アニメーションを指定したスクリプトの試行が終了したときや、メニュー ツリーで TrueLog ファイルのルート ノードをクリックしたときに、自動的に仮想ユーザーレポートが表示されるようにするには、**設定** > **オプション** > **ワークスペース** > **レポート** で **仮想ユーザー レポートを表示する** チェック ボックスをオンにしてください。

仮想ユーザー要約レポートには、以下に関する詳細情報が含まれています。

- 仮想ユーザー
- 検出されたエラー
- テスト スクリプトに定義されているトランザクションごとに追跡したレスポンス時間情報
- ダウンロードされた Web ページごとのページ タイマの測定値
- テスト スクリプトで宣言されている各 Web フォームに関連する測定値。POST、GET、および HEAD の各メソッドを使用したフォーム送信に対するレスポンス時間測定値とスループットを含みます。
- スクリプトで使用された個別のタイマおよびカウンタ (Measure 関数)
- IIOP、Web フォーム、TUXEDO、SAP などに関連する情報

TrueLog 内のエラーを検索する

TrueLog Explorer では、スクリプトの試行の後でエラーをすばやく検索できます。 そして、エラーとなったリクエストを調査して、必要なカスタマイズを行うことができます。

注: メニュー ツリーでは、再生エラーを含む API ノードは赤い「X」印が付いて表示されます。

1. 分析または変更したい TrueLog を開きます。
2. ワークフロー バーの **テストの分析** をクリックします。 **ワークフロー - テストの分析** ダイアログ ボックスが開きます。
3. **エラーの検索** リンクをクリックします。 **TrueLog のステップ スルー** ダイアログ ボックスが、**エラー** オプションが選択された状態で表示されます。
4. 一度に 1 つのエラーを検索しながら TrueLog 結果ファイル内を移動するには、**次を検索** をクリックします。

   別の方法で TrueLog 内を移動して、スクリプトが意図どおりに動作したかを視覚的に検証することもできます (**ページ全体**、**HTML ドキュメント**、**フォーム送信**、または **API 呼び出し**)。

ページ統計値を表示する

テスト実行が正確であることを検証した後、TrueLog Explorer では、**情報** ペインの **統計** タブを使って、"負荷のない" 状態でのアプリケーションのパフォーマンスを分析できます。 **概要** ページには、ページの総レスポンス時間、ドキュメント ダウンロード時間 (サーバー ビジー時間を含む)、埋め込みオブジェクトの受信に要した時間が表示されます。

詳細な Web ページ統計には、個々の Web ページ コンポーネントの正確なレスポンス時間が表示されます。 この詳細統計を使って、エラーやページ ダウンロードの遅延の根本原因を突き止めることができます。

Web ページの詳細な分析結果には、ページ コンポーネントごとに以下のデータが記載されています。

- DNS 検索時間
- 接続時間
- SSL ハンドシェイク時間
- リクエスト送信時間
- サーバー ビジー時間
- レスポンス受信時間
- キャッシュ統計値

概要ページを表示する

1. API ノードのツリー メニューから、統計を表示する API ノードを選択します。
2. **統計** タブをクリックして、**統計** ビューを開きます。
3. 詳細分析およびページの掘り下げを行うには、[URL] 列に一覧されている中から、特定のコンポーネントを選択します。

記録 Truelog と再生 Truelog を比較する

スクリプトの開発プロセス中に生成された TrueLog と、当初生成された TrueLog を比較することで、テスト スクリプトが正確に実行されたかどうかを確かめることができます。

1. ワークフロー バーの **テストの分析** ボタンをクリックします。 **ワークフロー - テストの分析** ダイアログ ボックスが表示されます。
2. **テスト実行の比較** をクリックします。
3. 対応する記録 TrueLog が比較ビューに開き、**TrueLog のステップ スルー** ダイアログ ボックスが、**ブラウザ ノード** オプションが選択された状態で表示されます。これにより、TrueLog をノードごとに比較することができます。
4. **次を検索** ボタンをクリックすると、TrueLog 結果ファイル内を 1 ページずつ移動することができます。

注: 再生時のコンテンツを表示しているウィンドウには、左上隅に緑色の三角マークが付いています。 アプリケーションの記録時に元々表示されていたコンテンツを表示しているウィンドウには、左上隅に赤い三角マークが付いています。

記録 TrueLog と再生 TrueLog を同期させる

比較モードでは、記録値と再生値の差分を識別できるように、再生 TrueLog と記録 TrueLog の間で対応する API ノードを同期することができます。

注: この機能は、TrueLog の自動同期が有効になっている場合は無効になります。

1. 次のいずれかを実行して比較モードを有効にします。
   - **表示** > **比較モード** を選択します。
   - ツールバーの **比較モード** ボタンをクリックします。
2. 対応する一組の記録/再生 TrueLog を開きます。
3. API ノードを右クリックし、**TrueLog の同期** を選択します。 TrueLog Explorer は、対応する TrueLog 内で、選択された API ノードと最も相関がある API ノードを見つけます。

テスト スクリプトをカスタマイズする

Silk Performer でテスト スクリプトを生成し、スクリプトの試行を実行したら、TrueLog Explorer を使用して次の方法などでスクリプトをカスタマイズすることができます。

- **入力データのパラメータ化**- ユーザー データのカスタマイズでは、記録された静的ユーザー入力データを、トランザクションごとに変化するパラメータ化された動的ユーザー データに置き換えることによって、現実に近いテスト スクリプトを作成できます。 このようなデータ駆動型テストを実行するために手動でスクリプトを作成する必要はありません。 この機能は、Web アプリケーション、データベース アプリケーション、XML アプリケーション、Citrix アプリケーション、SAPGUI アプリケーション、ターミナル エミュレーション アプリケーション、Oracle Forms アプリケーションで有効です。
- **テスト スクリプトへの検証の追加**- **検証の追加** ツールを使うと、テスト中にダウンロードされたデータの内容を確認することができ、サーバーの送信したコンテンツをクライアントが受信していることを検証できます。 検証は、システム開発後も、稼働中のパフォーマンス管理に役立ちます。 この機能は、Web アプリケーション、データベース アプリケーション、XML アプリケーション、SAPGUI アプリケーション、Citrix アプリケーション、ターミナル エミュレーション アプリケーション、Oracle Forms アプリケーションで有効です。 TrueLog Explorer は、Citrix サーバーをホストとするアプリケーションのビットマップとウィンドウの検証をサポートしています。

ユーザー入力データのカスタマイズ

ユーザー入力データのカスタマイズを行わないと、シミュレートするトランザクションはすべて同一のものになり、現実の環境でよく見られる変数を再現できません。

たとえば、**パラメータ ウィザード** を使用すると、テスト中にフォームに入力されるユーザー入力データをカスタマイズできます。**パラメータ ウィザード** を使用すると、テスト中にフォーム フィールドに入力される値を指定できます。そして、記録されたユーザー入力データをランダムなパラメータ化されたユーザー データに置き換えることによって、テスト スクリプトをより現実的なものにすることができます。

新しいパラメータを使用して HTML ユーザー データをカスタマイズする

読み進める前に、静的なセッション情報がすべてテスト スクリプトから削除されていることと、最も最近のスクリプトの試行で TrueLog Explorer に表示される TrueLog が生成されていることを確認してください。

HTML ベースのアプリケーションでは、ユーザー データのカスタマイズの目標は、フォーム フィールドに送信される値をカスタマイズすることです。

このタスクでは、ランダム変数に基づいてパラメータを作成する手順を説明します。

1. ワークフロー バーの **ユーザー データのカスタマイズ** をクリックします。**ワークフロー - ユーザー データのカスタマイズ** ダイアログ ボックスが開きます。
2. **HTML フォームにおけるユーザー入力データのカスタマイズ** リンクをクリックします。 これにより、TrueLog Explorer で以下のアクションが行われます。

   - メニュー ツリーで、最初の **WebPageSubmit** API 呼び出しノードを選択します。
   - **TrueLog のステップ スルー** ダイアログ ボックスを開きます (**フォーム送信** オプション ボタンが選択されています)。
   - **ポスト データ** ビューが表示されます。

     ポスト データ ビューには、選択されている WebPageSubmit 呼び出しによって送信された HTML フォームが含まれているページが表示されます。カーソルをフォーム コントロールの上に置くと、ツールチップに、そのコントロールの名前、初期値、および送信値が表示されます。オレンジ色の線は、下にある **フォーム データ** ビュー内の対応する BDL フォーム フィールド宣言を示します。
3. **TrueLog のステップ スルー** ダイアログ ボックスの **次を検索** または **前を検索** をクリックすると、TrueLog 内のすべての WebPageSubmit 呼び出しに目を通すことができます (これらは、ユーザー データのカスタマイズの候補となる呼び出しです)。

   注: ポスト データ ビューで強調表示されている HTML コントロールは、カスタマイズ可能なフォーム フィールドを表しています。
4. **ポスト データ** ページで、カスタマイズしたいフォーム コントロールを右クリックし、**値のカスタマイズ** を選択します。

   記録されている値を、さまざまな種類の入力データ（ファイルにあらかじめ定義した値や一般的なランダム値を含む）に置き換えることができます。また、記録された入力データをカスタマイズした値に置き換えるためのコードをテスト スクリプトに生成することもできます。

   パラメータ ウィザード が開きます。

   パラメータ ウィザード では、次の 2 つの方法でスクリプトの値を変更できます。

   - スクリプトの dclparam または dclrand のセクションで定義されている既存のパラメータを使用する
   - 新しい定数値、ランダム値、複数列データ ファイル内の値に基づいた、新しいパラメータを作成する

   新しいパラメータを作成すると、そのパラメータは既存パラメータ群に追加され、以降のカスタマイズで利用できるようになります。

5. **パラメータを新規に作成する** オプション ボタンをクリックし、それから **次へ** をクリックして新しいパラメータを作成します。 **パラメータの新規作成** ページが開きます。
6. **ランダム変数からパラメータを作成** オプション ボタンをクリックし、それから **次へ** をクリックします。 **ランダム変数** ページが開きます。
7. テスト スクリプトに挿入したいランダム変数の種類をリストボックスから選択し、**次へ** をクリックします。

   選択した変数の種類の簡単な説明が下部のウィンドウに表示されます。

   変数の名前と属性の指定 ページが開きます。
8. 変数の名前を **名前** テキスト ボックスに入力します。
9. 値を呼び出す順序として **ランダム** または **シーケンシャル** を指定します。

   ファイルからの文字列 というランダム変数タイプでは、指定されたファイルからランダムに選択または順次選択できるデータ文字列が生成されます。
10. **ファイル** グループ ボックスの **名前** リスト ボックスから構成済みのデータ ソースを選択して、**次へ** をクリックします。**使用法の選択** ページが表示されます。
11. 以下の選択肢のいずれかを選択して、新しいランダム変数の使用法を指定します。

    - **毎回**
    - **トランザクションごと**
    - **テストごと**
12. **終了** をクリックします。 これで、テスト スクリプトは、そのフォーム フィールドについて、記録された値ではなくランダム変数を使用するようになります。 新しいランダム変数関数が **BDL** ページに追加されます。

テスト スクリプトにランダム変数関数を追加した状態でスクリプトの試行を起動し、スクリプトがエラーなしに動作することを確かめます。

AJAX およびスクリプトのカスタマイズ

Silk Performer Recorder は、AJAX (Asynchronous JavaScript and XML) リクエストを利用した Web アプリケーションを記録して、それを再生できます。これが可能なのは、HTML レスポンス内の XML や JSON の形式で受信した AJAX 同期リクエスト/レスポンスを Silk Performer が認識しているからです。Silk Performer は、WebPageUrl 呼び出し時に発生する AJAX リクエストを記述します。

Silk Performer および TrueLog Explorer は、AJAX リクエスト内の値へのアクセスをサポートしています。これによって、スクリプト カスタマイズ機能 (入力データのパラメータ化、検証、解析、およびセッション情報のカスタマイズなど) を利用できます。

ヒント: InsuranceWeb デモ アプリケーション (*http://demo.borland.com/InsuranceWebExtJS/* で利用可能) には、テスト目的で JSON と XML シリアル化メソッドを切り替える機能が備わっています。**Select a Service or login** リストから、JSON の場合は Agent Lookup を、XML の場合は Agent Lookup (XML) を選択します。

JSON データと XML データの書式整形

JSON および XML は、AJAX アプリケーションや REST 技術などの環境で一般に使われるデータ構造形式です。Silk Performer は、BDF スクリプト内の XML 形式や JSON 形式のバイト ストリームを、書式整形して表示する機能をサポートしています。レンダリングされた JSON 形式のデータに対して、TrueLog Explorer のカスタマイズ機能を利用して、文字列値のカスタマイズがさらに簡単にできます。

JSON 形式のデータを BDF スクリプトに記録または挿入すると、Silk Performer は JSON バイト ストリームをそのまま表示します。JSON 形式で表示されると、Silk Performer のパラメータ ウィザードを使用して XML のカスタマイズを簡単に行うことができます。

Silk Performer には、JSON データを、書式整形された JSON 用のレンダリング ビューで表示するか、JSON のバイト ストリームをそのまま表示するかを選択するオプションがあります。

JSON の理解

JSON.org (www.json.org) によると、“JSON (JavaScript Object Notation) は、軽量なデータ交換形式です。人にとっては読み書きがしやすく、マシンにとっては解析および生成がしやすくなっています。JSON は、JavaScript プログラミング言語のサブセットに基づいています...""JSON はテキスト形式であり、言語を完全に気にせずに使用できますが、C、C++、C#、Java、JavaScript、Perl、Python などその他多くの C 言語ファミリーのプログラマによく知られている規約を使用します。このような特性により、JSON は理想的なデータ交換言語となります。”

TrueLog Explorer で JSON および XML 表示の書式整形を有効化する

TrueLog Explorer が JSON データや XML データを検出できない場合には、**レンダリング** ページで JSON および XML を表示する際の書式整形を有効化します。 TrueLog Explorer が JSON データや XML データを検出すると、**レンダリング** ページは自動的に書式整形されます。

1. TrueLog メニュー ツリーで JSON 形式または XML 形式のデータを含むノード (HTTP Post コマンド ノードなど) を選択します。
2. **レンダリング** タブをクリックします。
3. ツール バーの **自動表示** をクリックします。 JSON データおよび XML データが自動的に書式整形されます。
4. JSON データまたは XML データを右クリックし、 **ファイルの種類を指定してレンダリング** > **JSON** または **ファイルの種類を指定してレンダリング** > **XML** を選択してデータを書式整形します。

Silk Performer で JSON および XML 表示の書式整形を有効化する

1. JSON 形式または XML 形式のデータを含む BDL スクリプト内で、表示する画面データを右クリックします。
2. コンテキスト メニューから望ましい表示形式を選択します。
   - データを拡張 JSON 形式にするには、 **記述形式** > **JSON** を選択します。
   - データを XML 形式にするには、 **記述形式** > **XML** を選択します。

   コンテキスト メニューから **記述形式** > **自動判定** を選択すると、Silk Performer が画面データの種類に最適な形式を判断するようにできます。 デフォルトでは、JSON および XML データは、BDF スクリプトで書式整形されます。
3. 書式整形されたデータ文字列を右クリックして、Silk Performer の標準の文字列カスタマイズ コマンドにアクセスします。

   注: 書式整形された JSON データは、Silk Performer コード エディタに統合されるので、JSON 形式は、ツールバーの **元に戻す**/**やり直し** ボタンを使用して取り消し/やり直しできます。

   注: 書式整形された JSON および XML データの表示は、TrueLog Explorer でも利用できます。

検証

多くの場合、アプリケーション エラーが原因で誤った HTTP レスポンスが発生することはありません。 その代わり、アプリケーションは間違ったデータ値や、HTML コンテンツに組み込まれたエラー メッセージ (A Servlet Exception occurred... や Server Too Busy... など) を返します。 HTTP ステータス コードの簡単なチェックでは、この種のエラーは検出されません。 そのため、アプリケーション チェックをテスト スクリプトに組み込まない場合、アプリケーション エラーはしばしば見過ごされます。 検証関数は、単純な標準 HTTP エラー以外のアプリケーション エラーをチェックするのに役立ちます。

検証をテスト スクリプトに組み込むと、テストは単純な負荷テストから発展して、負荷テストと機能テストが組み合わさったものになります。 大規模な負荷テスト シナリオの際にも、パフォーマンス上の大きな不利益を被らずにこれらのスクリプトを使用することができます。 そのため、これらは新しい種類のアプリケーション エラー、つまり、負荷がかかったときにのみ発生し、付加的な検証チェックを行わない簡単な負荷テスト スクリプトでは見過ごされる可能性が高いエラーを検出できるようになります。

Silk Performer には、テスト スクリプトを検証機能で強化する方法が以下の 3 とおりあります。

- 記録セッション時に検証関数を自動的に Recorder に生成させます。
- 検証関数のコードを手動で作成することにより、スクリプトを直接に機能強化します。
- TrueLog Explorer を通じて検証を視覚的に適用します。BDL コードを 1 行も書く必要はありません。TrueLog Explorer が検証関数をスクリプト内に自動的に生成します。

ヒント: 詳細については、TrueLog Explorer のヘルプを参照してください。

記録中に自動的に検証を生成する

Silk Performer スクリプトによる記録中に、自動的に検証を生成できるようにするには:

1. Silk Performer 内から、**設定** > **アクティブ プロファイル** を選択します。 **プロファイル - [<プロファイル名>] - シミュレーション** ダイアログが表示されます。
2. ショートカット リストから **記録** > **Web** を選択し、**検証** タブをクリックします。
3. **記録** セクションで、**タイトル検証を記録する** および **ダイジェスト検証を記録する** チェック ボックスをオンにします。
4. **OK** をクリックします。

タイトル検証

タイトル検証は、アプリケーションが正しい Web ページを返すかどうかを確かめる簡単かつ自動的な手段になります。HTML ドキュメントを取得する際に、Recorder は、ページ レベルの Web API 呼び出しまたは低レベルの Web API 呼び出しごとにタイトル検証関数を自動的に生成できます。

注: HTML ページ タイトル (<TITLE HTML> タグの内容) が設定されていない場合や共通の名前を共用している場合には、タイトル検証は機能しません。

タイトル検証

```
transaction  TMain
begin
   ...
   WebVerifyHtmlTitle("ShopIt  -  Greetings",
WEB_FLAG_IGNORE_WHITE_SPACE  |
      WEB_FLAG_EQUAL  |  WEB_FLAG_CASE_SENSITIVE,  1,
      SEVERITY_ERROR,  bVerifyTitleSuccess1);
   WebPageUrl("http://myHost/shopit");
```

ダイジェスト検証

ダイジェスト検証は、比較的変更がない HTML ページのコンテンツを検証する場合に便利です。ダイジェスト検証は、動的な HTML ページ、つまりハイパーリンク、埋め込まれたオブジェクト、または非表示のフォーム フィールドにセッション情報を含んでいるために呼び出しごとに変化する HTML ページには役に立ちません。

この関数は、WebVerifyDataDigest チェックおよび WebVerifyHtmlDigest チェックを呼び出して、Silk Performer が再生中に受け取ったデータと、Recorder が対応する記録セッション中に取得したデータが異なるかどうかを検証します。これらの関数呼び出しによって、Silk Performer はすべての文字の出現に関する情報を含むダイジェストを生成し、それらの結果と Recorder によって生成されたダイジェストを比較します。この関数により、Silk Performer が、記録中に取得したデータと同じデータを再生中に取得するかどうかを検証できます。Recorder は、ダイジェスト検証に指定されたコンテンツ タイプ (デフォルトのコンテンツ タイプは、text/html) のドキュメントを受け取る各ページ レベルの Web API 呼び出しおよび低レベルの Web API 呼び出しに対するダイジェスト検証関数を自動的に生成します。

ダイジェスト検証

```
const
   gsVerDigest_ShopIt_Greetings  :=  "¥h014200000B..."
                                          "¥h016000500A..."
                                          "¥h030001000A..."
                                          "¥h0900410020...";

transaction  TMain
begin
   ...
   WebVerifyDataDigest(gsVerDigest_ShopIt_Greetings,  162);
   WebPageUrl("http://myHost/shopit",  "ShopIt  -  Greetings");
   ...
end  TMain;
```

再生中に検証チェックを有効にする

スクリプト再生中に有効または無効にできる検証チェックのリストに対する検証設定オプションを確認します。

ヒント: スクリプトのプロファイル設定は、WebSetOption という BDL 関数を使って上書きできます。

1. Silk Performer 内から、**設定** > **アクティブ プロファイル** を選択します。 **プロファイル - [<プロファイル名>] - シミュレーション** ダイアログが表示されます。
2. ショートカット リストから **再生** > **Web** を選択し、**検証** タブをクリックします。
3. **HTML/XML** と **データ** 領域で、再生中に有効化する検証チェック対象のチェック ボックスをオンにします。

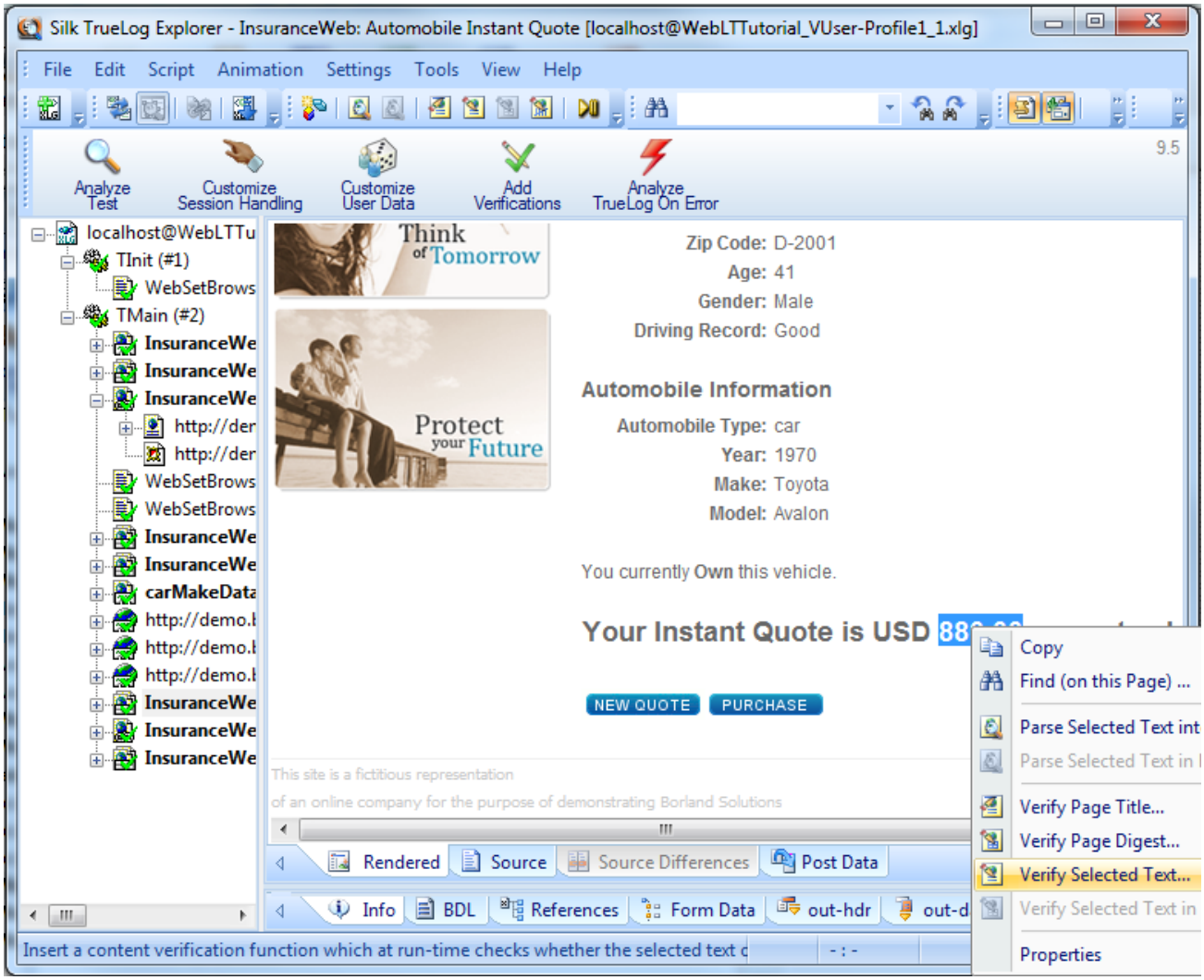
コンテンツ検証関数を挿入する

1. 分析または変更したい TrueLog を開きます。
2. 検証したいコンテンツ (テキストや画像など) を含む TrueLog API ノードを選択します。
3. 検証したいコンテンツを **ソース** ページで選択します。

   注: ページ タイトル検証関数およびページ ダイジェスト検証関数の場合は、このステップは不要です。
4. ワークフロー バーの **検証の追加** をクリックします。**ワークフロー - 検証の追加** ダイアログ ボックスが開きます。
5. 有効になっている検証を以下から 1 つ選択します。
   - ページ タイトルの検証
   - 選択したテキストの検証
   - HTML テーブル内で選択したテキストの検証
   - ダイジェストの検証
6. 表示されるダイアログ ボックスに必要事項を入力します。
   検証関数を BDL スクリプトにどう挿入するかを指定します。

   注: [左境界] と [右境界] は、自動的に識別されます。
7. BDL スクリプトに追加したい検証それぞれについて、この作業を繰り返します。
8. **ワークフロー - 検証の追加** ダイアログ ボックスで **はい** をクリックします。スクリプトの試行が開始します。
9. 検証が正常に合格したことを確認します。

検証を含む API ノードには、青い "V" 印が表示されます。



解析関数

解析関数は、検証関数と似ていますが、Web サーバーから返された内容を解析し、その値がテスト基準を満たすかどうかを確認するために使用されます。 検証関数が、基本的に指定された入力値の出現をチェックするのとは異なり、解析関数は、サーバー レスポンスの内容を解析し、解析された値を BDL 変数で返します。 通常、解析関数は以下のタスクで使用します。

- Web アプリケーションの状態情報を管理するために、セッション ID を解析して、スクリプト内の静的なセッション ID を解析された動的なセッション ID に置き換えます。 これは、解析関数の主な適用領域の 1 つです。
- 解析関数を使用して、検証関数では実現できない機能強化された検証をスクリプトに作成します。 たとえば、HTML テーブルの 3 行目 2 列目の値が、1 行目 2 列目の値と 2 行目 2 列目の値の和に等しいかどうかを検証できます。 スクリプトで解析関数を使用することによって、3 つの値を解析し、それらを比較することによって、この機能強化された検証チェックを作成できます。
- サーバーから返されたデータに基づいて、テスト スクリプトの一部を条件実行します。 たとえば、HTTP リクエストが、Results: <nnn> items found を含む HTML ページを返すとします。 <nnn> の値に基づいて、異なるアクションを実行します。 たとえば、<nnn> = 0 の場合は、トランザクションを終了し、<nnn> = 1 の場合は、リンク **詳細** に移動し、<nnn> is greater than 0 の場合

は、リンク **次へ** に移動します。 これを実現するには、 HTML ページの <nnn> の値を解析し、解析した値に基づいたアクションをスクリプトに記述する必要があります。

Silk Performer は、テスト スクリプトを解析機能で拡張するための 2 つの方法を提供します。

- 検証関数を手動でコーディングすることによって、スクリプトを直接機能強化します。
- TrueLog Explorer で解析関数を視覚的に適用します。BDL コードは、1 行も記述する必要はありません。 TrueLog Explorer は、スクリプトに解析関数を自動的に生成します。

ヒント: 詳細については、TrueLog Explorer のヘルプを参照してください。

HTML コンテンツの解析

HTML コンテンツの解析関数は、レンダリングで表示された HTML コンテンツの部分を解析できます。 HTML コンテンツの検証関数と解析関数では、ブラウザで表示されるテキスト コンテンツを検証や解析できます。 TrueLog Explorer の **HTML ビュー** にある、次の HTML コンテンツの解析関数を適用します。

- WebParseHtmlBound
- WebParseHtmlTitle
- WebParseTable
- WebParseResponseTag
- WebParseResponseTagContent

レスポンス データの解析

データ解析関数は、サーバーから返されたレスポンス データの一部を解析します。 HTML ドキュメントがサーバーから返された場合、HTML ドキュメントのソース コード全体が解析されます。 TrueLog Explorer の **ソース ビュー** にデータ検証関数を適用します。これには、以下の関数が含まれます。

- WebParseDataBound (この関数は、廃止された WebParseResponseData 関数を置き換えます)
- WebParseResponseHeader

セッション処理

Web アプリケーションは、Web サーバーが以降のクライアントのリクエストを処理できるように、一意の Web セッション ID を多くの場合使用します。

詳細については、TrueLog Explorer のヘルプを参照してください。

セッション処理のカスタマイズ

Web サーバー アプリケーションが、今後のクライアント リクエストを識別するために必要な情報を実行時に生成することがよくあります。 サーバーは、ブラウザへのレスポンスに、一意の文字列 (セッション ID と呼ばれる) を含めます。 この文字列は後続の各リクエストの一部としてブラウザからサーバーに返ります。これによりサーバーはリクエストが一部になっている固有な Web セッションを識別できます。 一般に、セッション ID はメソッドを参照し、Web サーバー アプリケーションは個別のユーザーを識別するために、該当するユーザーに対して以前にアプリケーションが持っていたユーザーのセッション情報の状態とこの ID を関連付けるために使用します。

セッション ID は、さまざまな方法でクライアントに送信されます。 クッキーにセッション ID を含めることはよくあります。またはハイパー リンクで使用される、URL の一部として内部 HTML、または埋め込みオブジェクト、非表示の HTML フォーム フィールドに含まれます。 セッション ID は、クッキー、URL、および HTTP のポスト データでサーバーに送り返されます。

クッキー内のセッション情報

サーバーから:

```
Set-Cookie: SessionID=LGIJALLCGEBMIBIMFKOEJIMM; path=/
```

サーバーへ:

```
Cookie: SessionID=LGIJALLCGEBMIBIMFKOEJIMM
```

HTML リンクの URL にあるセッション情報

サーバーから:

```
<html>
  ...
  <a href="/ShopIt/acknowledge.asp?
SessionID=LGIJALLCGEBMIBIMFKOEJIMM" >
  Enter Shop
  </a>
 ...
</html>
```

サーバーへ:

```
GET /ShopIt/acknowledge.asp? SessionID = LGIJALLCGEBMIBIMFKOEJIMM
HTTP/1.1
```

セッション情報が、非表示のフォーム フィールドに含まれる場合

サーバーから:

```
<html>
  ...
  <form action="kindofpayment.asp" method="post" >
  Currently we only accept Credit Cards
  <input type="hidden" name="SessionID"
value="LGIJALLCGEBMIBIMFKOEJIMM">
  <input type="text" name="name" value="Jack " >
  <input type="submit" name="paymentButton" value="Submit">
  </form>
  ...
</html>
```

サーバーへ:

```
POST /ShopIt/kindofpayment.asp HTTP/1.1
...
SessionId=LGIJALLCGEBMIBIMFKOEJIMM&name=Jack&paymentButton=Submi
t
```

カスタマイズしたセッション処理を使用すべき場合

スクリプト内の WebPageUrl 呼び出しで、クエリ文字列の一部にセッション ID が含まれる URL を使用しているとしましょう。このスクリプトを再生すると、ハードコードされたこの静的なセッション ID がサーバーに送信されます。セッション ID では再生したセッションが正しく識別されないため (それでも記録セッションは識別されます)、再生は正しく機能しません。スクリプト内の静的なセッション ID を実行時に生成された動的な値に置き換えなければ、Web アプリケーションでは通常、We are sorry, your session has expired.Please return to the login screen and try again (セッションが期限切れです。ログイン画面に戻ってやり直してください) などのエラーが発生します。

幸いなことに、Silk Performer ではセッションのカスタマイズが必要になることは多くなく、仮にセッションの手動カスタマイズが必要な場合でも、Silk Performer の TrueLog Explorer の指示に従って手順を進めることができます。

ほとんどの場合、スクリプトを記録する際には、セッション処理をカスタマイズするために必要な変更は何もなく、そのまま動作します。Silk Performer では、そのため、以下のようにハードコードされたセッション ID をユーザーが手動で処理しないですむように、非常に高度な方法を複数用いています。

- クッキー管理：サーバーがセッション情報の交換にクッキーを使用する場合、Silk Performer では動的なセッション ID 値を自動的に処理します。Silk Performer はブラウザのクッキー管理を正確にエミュレートしているため、ブラウザがサーバーに送信するのと同じようにクッキーを送信します。状態管理に手動操作は不要です。
- ページ レベルの Web API：記録の際にページレベル API を使用する (これがデフォルト設定) と、WebPageLink や WebPageSubmit などの、コンテキストのある Web API 関数呼び出しを主に生成するスクリプトが生成されます。コンテキストのある Web API 呼び出しは、HTTP レベルではなく HTML レベルで動作するため、URL をパラメータとして使用しません。コンテキストのあるすべての API 呼び出しでは、セッションの手動カスタマイズは必要ありません。ページレベル API が使用されるのは、**ワークフロー - プロジェクトの概要設定** ダイアログでアプリケーションの種類として Web business transaction (HTML/HTTP) を選択した場合です。そのため、低レベルの Web API ではなく Silk Performer のページレベル API を使用することを強くお勧めします。

HTML の動的生成にクライアント側 Java Script を多用した結果、Silk Performer Recorder が HTML コンテキストを見失い、コンテキストのない Web API 呼び出しを記述する場合があります。WebPageUrl や WebPageForm などの、コンテキストのない Web API 呼び出しには、URL がパラメータとして含まれています。これらのまれなケースでは、ハードコードされたセッション ID がスクリプトに含まれていることがあります。それらが書かれている箇所は、Web API 呼び出しの URL パラメータや、スクリプトの dclform セクションに宣言されているフォーム フィールドです。

コンテキストのあるスクリプト (カスタマイズ不要)

```
transaction TMain
  begin
    WebPageUrl("http://lab38/ShopIt/"); // first call is always context-less
    WebPageLink("Join the experience!");
    WebPageSubmit("Enter", SHOPIT_MAIN_ASP001);
    WebPageLink("Products");
  end TMain;

 dclform
 SHOPIT_MAIN_ASP001:
    "SessionID" := "" <USE_HTML_VAL>, // hidden value:
    "LGIJALLCGEBMIBIMFKOEJIMM"
    // recognized as a hidden form
    // field, the value is taken from
    // the actual HTML form field.
    "name" := "Jack", // changed
    "New-Name-Button" := "" <USE_HTML_VAL> ; //unchanged
    value: "Enter"
```

コンテキストのない関数を用いたスクリプト (カスタマイズの必要な静的なセッションデータが DCLFORM セクションに含まれている)

```
transaction TMain
  begin
    WebPageUrl("http://lab38/ShopIt/"); // first call is always context-less
    WebPageUrl("http://lab38/ShopIt/main.asp", NULL,
SHOPIT_MAIN_ASP001);
    WebPageForm("http://lab38/ShopIt/main.asp", SHOPIT_MAIN_ASP002);
    WebPageUrl("http://lab38/ShopIt/products.asp");
  end TMain;

 dclform
```

```
SHOPIT_MAIN_ASP001:
    "from"  :=  "welcome";

    SHOPIT_MAIN_ASP002:
  "SessionID"           :=  "LGIJALLCGEBMIBIMFKOEJIMM",
    "name"                  :=  "Jack",
    "New-Name-Button"  :=  "Enter";
```

セッション処理のカスタマイズ

次のステップはセッション処理をカスタマイズするために必要です。

詳細については、TrueLog Explorer のヘルプを参照してください。

1. カスタマイズするために必要なセッション ID を識別する。
2. そのセッション ID がサーバーからクライアントへ送信された最初のレスポンス (スクリプト内の Web API 呼び出し) を検索する。
3. Web API 呼び出しのレスポンスから見つかったセッション ID の解析結果を変数に入力する。
4. スクリプト内の、ハードコードされているセッション ID の出現箇所すべてをその変数に置換する。

ユーザー プロファイル

ブラウザ タイプ

テストで用いられる仮想ユーザーは、幅広い選択肢の中から選んだ任意の Web ブラウザとそこに装備されている任意の機能を使用するようにカスタマイズすることができます。 今日最もよく使用されているブラウザをエミュレートすることができます。 また、デバイス ユーザーにサービスを提供するブラウザなど、あまり有名でないブラウザも使用可能です。 さまざまなバージョンのスレッド処理技術および HTTP を用いて、カスタム ブラウザを定義することができます。

帯域幅

モデムは、ホーム ユーザーがインターネットへのアクセスに通常使用する手段です。 ユーザーのモデムはネットワーク通信チェーンで最も遅いリンクになることがあるため、モデムを負荷テストでシミュレートすることができます。

仮想ユーザーは、今日の消費者の間で広く使用されている主要な接続タイプの帯域幅を使用するようにカスタマイズすることができます。 カスタム設定では、接続が下り方向 (サーバーからクライアントへ) のトラフィックと上り方向 (クライアントからサーバーへ) のトラフィックに異なる帯域幅設定を使用するように定義することができます。

プロファイルを追加する

1. **プロジェクト** > **プロファイルの新規作成** を選択します。 **新規のプロファイル** ダイアログが開きます。

2. **新しいプロファイルの名前を入力する** に名前を入力して **OK** をクリックします。 **プロジェクト** メニュー ツリーで **プロファイル** フォルダが展開し、新しいプロファイルが利用可能になります。

ブラウザ設定を構成する

1. Silk Performer の **プロジェクト** ツリーで **プロファイル** ノードを展開します。
2. 構成対象のプロファイルを右クリックして、**プロファイルの編集** を選択します。

   ヒント: あるいは、メニュー バーから **設定** > **アクティブ プロファイル** を選択することもできます。

プロファイル - [<プロファイル名>] - シミュレーション ダイアログ ボックスが開きます。左側のショートカット リストに **再生** カテゴリが表示されます。

3. ショートカット リストで **Web** アイコンをクリックします。
4. **ブラウザ** タブをクリックします。

   ブラウザの種類 領域を使用して、ブラウザ固有の設定を行います。
5. **ブラウザ** リスト ボックスから、シミュレーションに使用する Web ブラウザを選択します。

   この選択により、HTTP リクエストに含まれるヘッダー情報の形式と、シミュレーションに使用するスレッド モデルが決まります。

   注: モバイル Web アプリケーション テスト (iPhone、iPad、Android、Windows Phone または Blackberry) では、記録に使用するユーザー エージェント文字列を変更できます。
6. **OK** をクリックして、設定を保存します。

ブラウザ シミュレーション設定を構成する

1. Silk Performer の **プロジェクト** ツリーで **プロファイル** ノードを展開します。
2. 構成対象のプロファイルを右クリックして、**プロファイルの編集** を選択します。

   ヒント: あるいは、メニュー バーから **設定** > **アクティブ プロファイル** を選択することもできます。

   プロファイル - [<プロファイル名>] - シミュレーション ダイアログ ボックスが開きます。左側のショートカット リストに **再生** カテゴリが表示されます。
3. ショートカット リストで **Web** アイコンをクリックします。
4. **シミュレーション** タブをクリックします。

   シミュレーション 領域を使用して、Web サイトを訪問するユーザーの現実的なシミュレーションのためのオプションを設定します。
5. トランザクションのたびに、仮想ユーザーにブラウザ エミュレーションをリセットさせるには、**トランザクションごとにユーザーの動作をシミュレートする** チェック ボックスをオンにします。

   選択した追加オプションに応じて、Silk Performer は、Web サイトを初めて訪問したユーザー、またはそのサイトを再訪問したユーザーのいずれかをシミュレートします。 サイトを初めて訪問したユーザーの場合は、永続クッキーが設定されておらず、ドキュメントもキャッシュされていません。一方、サイトを再訪問したユーザーの場合は、通常、再訪問までの間にブラウザを閉じたとしても、ドキュメントがキャッシュされ、永続クッキーが設定されています。 このオプションを無効にすると、仮想ユーザーは、テストが終了するまで閉じられることのない Web ブラウザをエミュレートします。したがって、複数のトランザクションにわたってキャッシュされた情報が再利用されます。
6. 初めて Web サイトを訪問したユーザーを実際にシミュレートするには、**初めてのユーザー** オプション ボタンをクリックします。

   トランザクションごとに、永続的な接続は閉じられ、Web ブラウザのエミュレーションがリセットされます。また、ドキュメント キャッシュ、ドキュメント履歴、クッキー データベース、認証データベース、および SSL コンテキスト キャッシュもクリアされます。 このオプションを選択した場合、Silk Performer は、サーバーから全サイト(すべてのファイルを含む)をダウンロードします。
7. Web サイトを再訪問したユーザーを実際にシミュレートするには、**2 回目以降のユーザー** オプション ボタンをクリックします。

   トランザクションごとに、永続的な接続は閉じられ、ドキュメント履歴、非永続クッキー データベース、認証データベース、および SSL コンテキスト キャッシュがクリアされます。 このオプションを選択した場合、ユーザーはドキュメント キャッシュをクリアしません。 詳細については、『The BDL Function Reference』(英語) の WebSetUserBehavior 関数を参照してください。

   ユーザーの許容度をシミュレートするための詳細なオプションを調整するには、**ユーザーの許容度** 領域を使用します。
8. **OK** クリックして、**プロファイル - [<プロファイル名>]** ダイアログ ボックスの設定を保存します。

監視を構成する

テストを実行する前に、Performance Explorer (Silk Performer サーバー監視ツール) がそのテストで使用するローカルおよびリモート サーバーをどのように監視するかを定義する必要があります。 サーバーを監視すると、サーバーのボトルネックが明らかになり、その場所を特定して解決できるため、オペレーティング システムとアプリケーション サーバーのパフォーマンスを調べることができます。

次の 3 つの監視オプションが使用できます。

- **デフォルトの監視** - このオプションは、テスト対象のアプリケーション タイプに基づいて、推奨されるデータ ソースのセットを監視するよう Performance Explorer に指示します。 これは、Performance Explorer ワークスペースの **自動的に監視を開始する** と **デフォルト監視テンプレートを使用する** を有効にするのと同じです (**設定** > **アクティブ プロファイル** > **再生** > **監視** > **デフォルト監視テンプレートを使用する**)。
- **カスタム監視** - このオプションでは Performance Explorer が監視モードで開き、**データ ソース ウィザード - システムの選択** ダイアログ ボックスが表示され、手動でデータ ソースを設定できます。 Performance Explorer 監視プロジェクト設定は、Silk Performer プロジェクト設定とともに保存されます。
- **監視なし** - このオプションを使用すると、ローカルまたはリモート サーバーを監視せずにテストを実行できます。 このオプションでは、**自動的に監視を開始する** 設定が無効になります (**設定** > **アクティブ プロファイル** > **再生** > **監視** > **デフォルト監視テンプレートを使用する**)。

監視オプションを定義する

1. ワークフロー バーの **監視の設定** をクリックします。 **ワークフロー - 監視の設定** ダイアログ ボックスが表示されます。
2. 次のいずれかのオプションを選択して、**次へ** をクリックします。

   - **デフォルトの監視** - このオプションは、テスト対象のアプリケーション タイプに基づいて、推奨されるデータ ソースのセットを監視するよう Performance Explorer に指示します。 これは、Performance Explorer ワークスペースの **自動的に監視を開始する** と **デフォルト監視テンプレートを使用する** を有効にするのと同じです (**設定** > **アクティブ プロファイル** > **再生** > **監視** > **デフォルト監視テンプレートを使用する**)。
   - **カスタム監視** - このオプションでは Performance Explorer が監視モードで開き、**データ ソース ウィザード - システムの選択** ダイアログ ボックスが表示され、手動でデータ ソースを設定できます。 Performance Explorer 監視プロジェクト設定は、Silk Performer プロジェクト設定とともに保存されます。
   - **監視なし** - このオプションを使用すると、ローカルまたはリモート サーバーを監視せずにテストを実行できます。 このオプションでは、**自動的に監視を開始する** 設定が無効になります (**設定** > **アクティブ プロファイル** > **再生** > **監視** > **デフォルト監視テンプレートを使用する**)。

   (デフォルト監視およびカスタム監視のみ) ログを有効にしていると、確認ダイアログ ボックスにより通知されます。 ログがテスト結果に悪影響を及ぼすことがあります。
3. **OK** をクリックしてログ設定を承認するか、**キャンセル** をクリックしてログ オプションを調整します (**設定** > **アクティブ プロファイル** > **結果** > **ログ**)。
4. *(カスタム監視のみ)* Performance Explorer が開始され、**データ ソース ウィザード** が開きます。 ウィザードに従って手順を実行します。
5. **ワークフロー - ワークロードの設定** ダイアログ ボックスが表示されます。 **OK** をクリックして監視設定を承認します。

ワークロードの調整

ワークロードの設定は、負荷テストを実行するプロセスの一部です。 Silk Performer には、負荷テストの基本として使用するさまざまなワークロード モデルが用意されています。 ワークロードを構成する前に、ニーズに最も適したモデルを選択する必要があります。

複数のワークロード モデルを負荷テスト プロジェクトで定義して、あとで利用するために保存できますが、一度にアクティブにできるワークロード モデルは 1 つだけです。 承認済みのベースラインの結果はワークロード モデルに関連付けられています。 ワークロード モデルをコピーまたは名前変更すると、対応して承認済みのベースラインの結果もコピーまたは名前変更されます。

ワークロード モデル

Silk Performer には、次のワークロード モデルが用意されています。

- **増加** – 負荷テストの開始時は、Silk Performer は定義されているユーザー総数ではなく、 その指定された一部のみをシミュレートします。 ワークロードは、ユーザー リストで指定されたすべてのユーザーが実行されるまで段階的に増加します。

  このワークロード モデルは、システムがクラッシュしたりレスポンス時間やエラーしきい値の許容範囲内で応答しなくなる負荷レベルを調べる場合に特に有効です。
- **定常状態** – このモデルでは、テスト全体を通して同じ数の仮想ユーザーが使用されます。 負荷テスト スクリプトで定義されているトランザクションを各仮想ユーザーで実行します。 仮想ユーザーは、作業が終了すると再度トランザクションを実行します。 トランザクション間の遅延はありません。指定されたシミュレーション時間に達すると、テストは終了します。

  このワークロード モデルは、特定の負荷レベルでのテスト対象システムの動作を調べる場合に特に有効です。
- **動的** – 仮想ユーザー数をテストの実行中に手動で変更できます。 仮想ユーザーの最大数を設定した後は、テスト中いつでもその範囲内で仮想ユーザー数を増減できます。 シミュレーション時間は指定されません。 テストを手動で終了させる必要があります。

  このワークロード モデルは、さまざまな負荷レベルで実験したり、負荷テスト時に負荷レベルを調節できるようにする場合に特に有効です。
- **終日** – このワークロード モデルでは、柔軟な方法で負荷の分散を定義できます。 負荷テストの任意の期間に、異なる数の仮想ユーザーを割り当てることができます。ユーザー タイプごとに異なる負荷分散を用いることができます。 そのため、就業日ワークロードや週単位ワークロードなどの複雑なワークロード シナリオを設計できます。 実行が開始されていない期間について、負荷テスト時に負荷レベルを調整することもできます。

  このワークロード モデルは、複雑で持続的なワークロード シナリオをできるだけ現実的な方法でモデル化する場合に特に有効です。
- **キュー** – このモデルでは、トランザクションは定められた出現率に従ってスケジューリングされます。この比率は、シミュレーション時間とスクリプトの dcluser セクションで指定したトランザクション数から計算される平均間隔に基づいたランダム値です。 すべての仮想ユーザーが定められたタスクを完了すると、負荷テストは終了します。

  注: このモデルでは、出現率はランダムに設定されるため、テストは指定されたシミュレーション時間より長くなる場合があります。 たとえば、シミュレーション時間を 3,000 秒に指定して、100 個のトランザクションを実行する場合は、平均のトランザクション出現率は 30 秒になります。

  このワークロード モデルは、キュー メカニズムを用いて複数の同時リクエストを処理するワークロードをシミュレートする場合に特に有効です。エンド ユーザーではなく Web サーバーからリクエストを受け取るアプリケーション サーバー (たとえば、サーブレット エンジンやトランザクション サーバーなど) は通常、このようなキュー モデルを用いて正確にテストすることができます。

- **検証** – 検証テストの実行は、拡張検証機能と組み合わせた場合に特に役立ちます。 この組み合わせは、Web ベース アプリケーションの回帰テストにも用いることができます。 検証テストでは常に、指定されたエージェント コンピュータで特定ユーザー タイプの単一のユーザーを実行します。

  このワークロードは、Web アプリケーションの検証を自動化する場合や、検証テストをコマンド ライン インターフェイスから開始する場合に特に有効です。

ワークロードの定義

負荷テストを実行する前に、使用するワークロード モデルを指定して、ワークロード設定を構成する必要があります。 ワークロード モデルを選択するには、ワークフロー バーで **ワークロードの調整** をクリックします。

1. ワークフロー バーの **ワークロードの調整** をクリックします。 **ワークフロー - ワークロードの選択と調整** ダイアログ ボックスが表示されます。
2. 以下のワークロード モデルのいずれかを選択します。
   - 増加
   - 定常状態
   - 動的
   - 終日
   - キュー
   - 検証
3. **次へ** をクリックします。
4. ヘルプにある手順に基づいて、特定のワークロード タイプを構成します。

負荷テストを実行する

負荷テストでは、負荷がサーバーのパフォーマンスに与える影響を調査するために、テスト スクリプトを利用して、ターゲット サーバーに対して複数の仮想ユーザーが実行されます。 大規模な負荷テストの場合は、ローカル エリア ネットワーク上に適切なテスト環境をセットアップする必要があります。このようなテスト環境には、仮想ユーザーのホストとなるエージェント コンピュータも含まれます。

タスクを完了するには、以下のことが不可欠です。

- 実行対象のテストに適したタイプのオプションを設定する。
- 必須のワークロードを正確に定義する。
- サーバーのパフォーマンスを評価するために、テスト結果の生成を有効にする。

負荷テスト中に結果セット一式のログを有効にするのは避けてください。負荷テスト結果と干渉する場合があります。 ただし、**エラー時に TrueLog を生成する** ログ オプションを利用すると、エラーが発生した場合に、必要なログ ファイルがディスクに書き込まれます。TrueLog On Error を利用すると、再生エラーをビジュアルに調査できます。

負荷テストを実行する

1. ワークフロー バー の **テストの実行** をクリックします。 **ワークフロー - ワークロードの設定** ダイアログ ボックスが表示されます。
2. 負荷テストを開始するには、**実行** をクリックします。
3. **結果ファイル サブディレクトリの新規作成** ダイアログ ボックスで **OK** をクリックします。

   *(*省略可能*)* 結果サブディレクトリの名前を指定するには、**一意なサブディレクトリを自動的に生成する** チェック ボックスをオフにし、**結果ファイルのサブディレクトリを指定する** テキスト ボックスに新しいサブディレクトリの名前を入力します。

Silk Performer の表形式の監視ビューと Performance Explorer のグラフィカルな監視ビューを表示することによって、テストの進行状況とサーバーの活動状況を監視します。

負荷テストを監視する

Silk Performer の 負荷テスト実行中に、詳細なリアルタイム情報を利用できます。 クライアント側、サーバー側の両方のアクティビティの報告がグラフィック表示とフルテキストで行われるので、テストの進捗状況をその推移に従って直観的に監視できます。

テストを行うワークベンチから直接、そのテスト中のエージェント コンピュータと仮想ユーザーの総合的な概要情報を参照できます。 表示する情報の詳細レベルは、テスト中のすべてのエージェント コンピュータの進捗状況を概観するレベルから、各仮想ユーザーによって実行されるトランザクションの詳細を網羅して表示するレベルに至るまで制御できます。 各エージェントと各ユーザーの進捗情報は、複数のカテゴリに分けられています。 ユーザーごとのランタイム情報には、トランザクション、タイマ、関数、エラーに関する色分けされた見出し (カスタマイズ可能) が含まれています。

また、ターゲット サーバーのパフォーマンスのリアルタイム監視は、グラフィック形式で利用できます。現在広く使用されているすべてのオペレーティング システムで動作する各種の Web サーバー、アプリケーション サーバー、およびデータベース サーバーからの最も関連のあるパフォーマンス関連情報がグラフで表示されます。 同時に複数のグラフを表示することもできます。これらのグラフを並べて、最も関連のあるものを比較し、対比させることができます。 メニュー ツリー エディタを利用すると、任意のデータソースから得られる要素をグラフ上で組み合わせることができます。 クライアント アプリケーションから得られるレスポンス時間その他のパフォーマンス情報は、サーバーからのパフォーマンス データと同じグラフに表示できます。 この機能によって、直接ビジュアルな比較が可能になり、サーバーの問題点がクライアントの動作に与える影響を特定することができます。

負荷テスト中にエージェントを監視する

監視 ウィンドウを使用して、負荷テスト実行中の進捗状況を表示します。 このウィンドウの上部には、エージェント コンピュータとユーザー グループの進捗情報が表示されます。

数多くの情報オプションの中から、次の情報を参照できます。

- 特定のエージェントのステータス
- エージェント上で完了したテストの割合
- 実行されたトランザクション数

負荷テスト中に特定のエージェントを監視する

監視 ウィンドウの上部で、監視対象の特定のエージェントを選択します。

選択したエージェント上で実行中の仮想ユーザーについて、以下の情報が **監視** ウィンドウの下部に表示されます。

- ステータス
- 現在のトランザクション名
- 完了した作業の割合
- 実行されたトランザクション数

負荷テスト中に特定の仮想ユーザーを監視する

監視 ウィンドウの下部で、監視対象の仮想ユーザーを右クリックし、**[仮想ユーザー名] の出力を表示** を選択します。

Silk Performer は **仮想ユーザー** ウィンドウに、選択したユーザーに関する詳細なランタイム情報 (そのユーザーが実行したトランザクションと関数、そのユーザーがサーバーとの間で送受信したデータなど) を表示します。

ヒント: 仮想ユーザー領域を右クリックして **列の選択** を選択すると、表示する列を選択できます。

サーバー パフォーマンスの監視にグラフを使用する

1. ワークフロー バーの **ベースラインの確認** をクリックします。 **ワークフロー - ベースラインの確認** ダイアログ ボックスが開きます。
2. **監視オプションの定義** をクリックして、オンライン パフォーマンス データを受信するための設定を行います。 **プロファイル - [プロファイル名] - 結果** ダイアログ ボックスが開きます。
3. **プロファイル - [プロファイル名] - 結果** ダイアログ ボックスで、負荷テストの実行中に **自動的に監視を開始する** チェック ボックスをオンにし、以下のオプションのいずれかを選択します。
   - **デフォルト監視テンプレートを使用する** オプション ボタンをクリックします。
   - **カスタム監視テンプレートを使用する** オプション ボタンをクリックして、カスタム監視テンプレートを作成します。
4. **カスタム監視テンプレートの編集** をクリックします。 Performance Explorer が表示されます。
5. 現在使用中のすべての監視ウィンドウを閉じます。
6. Performance Explorer ワークフロー バーの **サーバーの監視** をクリックします。
   または、メニュー バーから **結果** > **サーバーの監視** を選択します。
   データ ソース ウィザード/データ ソースの選択 ダイアログ ボックスが開きます。
7. 次のいずれか 1 つのステップを行います：
   - サーバーが提供するデータ ソースがわかっている場合、**定義済みのデータ ソースから選択する** オプション ボタンをクリックして、一覧からあらかじめ定義されているデータ ソースを選択します。
   - サーバーが提供するデータ ソースがはっきりしない場合、**データ ソースを検出させる** オプション ボタンをクリックして、Performance Explorer がサーバーをスキャンして、利用可能なデータ ソースを検出します。
8. **次へ** をクリックします。
   メニュー ツリーで、サーバーとアプリケーションが実行されているオペレーティング システムに対応するフォルダを展開します。
9. 表示されたリストから、監視するサーバー アプリケーションを選択します。
   たとえば、オペレーティング システムを監視するには、**System** を選択します。
10. **次へ** をクリックします。 **接続パラメータ** ダイアログ ボックスが開きます。
11. **接続パラメータ** 領域で、該当するサーバー システムのホスト名または IP アドレス、ポート番号、その他のデータなど、データ ソースへの接続に必要な接続パラメータを指定します。
    指定するデータは、監視対象のコンピュータで稼働しているオペレーティング システムによって異なります。
12. **次へ** をクリックします。 **表示する測定値の選択** ダイアログ ボックスが開きます。
13. メニュー ツリーを展開して、監視する要素を選択します。
14. **終了** をクリックします。 監視グラフには、指定した要素のサーバー パフォーマンスがリアルタイムで色分けされて表示されます。 グラフの下には、グラフに含まれている要素のリスト、色分け表示、および各要素のパフォーマンス情報が表示されます。

テスト結果の検討

Silk Performer にはテスト結果の表示、レポート、および分析に利用できるアプローチがいくつか用意されています。 定義しておいた測定をテスト中に行い、さまざまなグラフィック形式および表形式で表示することができます。 オプションは、以下のとおりです。

- **Performance Explorer**：テスト結果を表示するための主要なツールです。 グラフィック機能を幅広く備えており、いくつでも必要な数の要素を持つ結果を、ユーザー定義によるグラフで表示します。 異

なるテストの結果を比較することができます。 サーバー監視のための幅広い機能があります。 ユーザー タイプ統計値を時系列テスト結果情報と組み合わせた包括的な HTML ベースの概要レポートも利用できます。

- **TrueLog On Error : Silk Performer** は、さまざまなアプリケーションの種類について、負荷時の視覚的検証機能を提供します。 負荷テスト中に、詳細なコンテンツ検証チェックと完全なエラーの詳細分析と組み合わせることができます。
- **仮想ユーザー レポート ファイル :** 有効になっていると、これらのファイルにはユーザーごとのシミュレーション結果が格納されます。 各ユーザーの詳細な測定値が、表形式で表示されます。
- **仮想ユーザー出力ファイル :** 有効になっていると、これらのファイルにはテスト スクリプトに使用される write 文の出力が格納されます。
- **ベースライン レポート :** XML/XSL ベースの詳細なレポートであり、要約テーブル、トランザクション レスポンス時間の詳細、アクセスされたすべての HTML ページのタイマ、Web フォーム、発生したエラーなどが提供されます。 この情報はベースライン テストに関係のあるすべてのユーザー タイプで利用できます。
- **Silk Central レポート : Silk Performer** プロジェクトは、Silk Central (Silk Central) テスト計画に統合され、Silk Central から直接実行されます。 これにより、強力なテスト結果分析とレポート作成が可能になります。 Silk Central レポート作成の詳細については、『 *Silk Central* ヘルプ』を参照してください。

TrueLog On Error

Silk Performer TrueLog 技術を使用すると、アプリケーションに重い負荷がかかっている場合に、ユーザーのサブセットにのみ通常発生するエラーを見つけ出すことができます。 大半のアプリケーションでは、このような種類の負荷は、アプリケーションが実際に配置されてから経験することになる場合がほとんどです。 典型的なエラーとしては、Web ページのテキストの誤り、計算値や表示値の誤り、「サーブレット エラー」や「サーバーがビジー状態です」 といったアプリケーション関連のメッセージなどがあります。 これらはシステムレベルのエラーではなく、Web ページには、ステータス コード「HTTP 200」で表示されるものです。

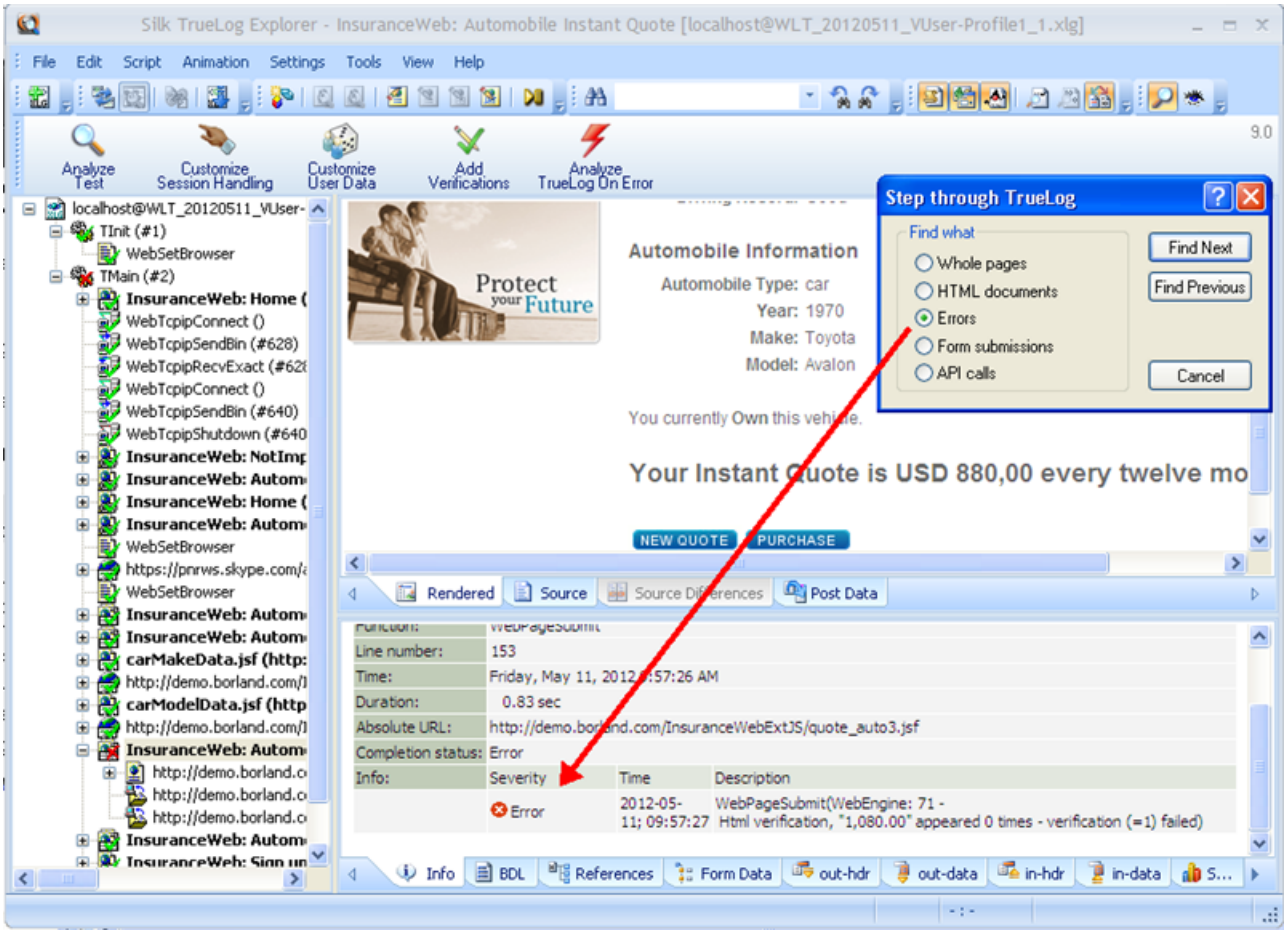
TrueLog Explorer には、次の機能を使用した、負荷時の Silk Performer 検証機能のビューが備わっています。

- ビジュアル コンテンツ検証では、検証するコンテンツを視覚的に定義できます。
- TrueLog On Error 生成と TrueLog On Error 分析の機能により、エラーを視覚的に分析してその根本原因を特定することができます。

TrueLog Explorer でエラーを表示する

1. 負荷テスト中にエラーが発生した場合、ワークフロー バーの **結果の検討** ボタンをクリックすると、TrueLog の内容を視覚化して表示することができます。 **ワークフロー - 結果の検討** ダイアログが開きます。
2. Silk TrueLog Explorer リンクをクリックします。 TrueLog Explorer が開き、**TrueLog のステップ スルー** ダイアログ ボックスがアクティブになります。 **エラー** を選択します。
3. エラーのオカレンスを 1 つずつたどっていきます。

ヒント: エラーの履歴を表示するには、メニュー ツリーから 1 つ前の API ノードをクリックします。



Performance Explorer の概要

Performance Explorer には 2 種類の主要な機能があります。

- 結果の分析
- リアルタイム監視

結果の分析

Performance Explorer の結果の分析機能を使用すると、完了した負荷テストの結果をグラフ、表、レポートを利用して分析できます。グラフを使用すると、負荷テスト中に収集したデータを可視化でき、レポートや表は、重要なデータをまとめたり見つけたりするのに役立ちます。すべてのグラフ、表、レポートの基礎をなすのが .tsd ファイルのデータです。各負荷テスト中に、Silk Performer は大量のデータをキャプチャし、いくつかの時系列データ (.tsd) ファイルに保存します。負荷テストが完了すると、Performance Explorer でデータ (.tsd ファイル) を読み込み、要求に従って編集できます。すべてのグラフが完全にカスタマイズ可能で、必要な数の測定値を含めることができます。1 つまたは複数のテストからの測定値を使用して、複数のグラフを開き、類似性や差異を表示できます。 Performance Explorer は、グラフ、表、レポートのさまざまなテンプレートを提供します。さらに、クライアントとサーバー パフォーマンスに関する情報を 1 つのグラフに置くことができるので、サーバー パフォーマンスがクライアントの動作に与える影響を直接見ることができます。変更をテンプレートとして保存すれば、個々の設定を再利用できます。

リアルタイム監視

Performance Explorer のリアルタイム監視機能を使用すると、システムのパフォーマンスをリアルタイムに表示するグラフを作成、設定することにより、幅広いシステムを監視できます。同時に複数のグラフを開き、複数のシステム (Web サーバーのパフォーマンスとオペレーティング システムのパフォーマンスなど) のパフォーマンスを同時に監視することができます。グラフに測定値を追加することは、

Performance Explorer では直感的で簡単です。追加する測定値 (複数でも可) を、ツリーからグラフにドラックするだけです。

概要レポートの測定値

概要レポートは、次のセクションで構成されます。

- 全般情報
- 要約テーブル
- ユーザー タイプ
- カスタム グラフ
- カスタム テーブル
- 詳細グラフ

全般情報

全般情報セクションには、管理情報が表形式で表示され、負荷テストの重要な結果がグラフ形式で表示されます。

管理情報には、プロジェクト名、プロジェクトの説明、負荷テスト番号、負荷テストの説明、負荷テストの日付、負荷テストの期間、使用されたエージェント コンピュータの数、および実行していた仮想ユーザーの数が含まれます。

グラフには、アクティブな仮想ユーザーの数、トランザクションのレスポンス時間の測定値、および負荷テスト中に発生したエラーの数が表示されます。 トランザクションのレスポンス時間は、正常に実行されたトランザクション、失敗したトランザクション、および取り消されたトランザクションについて測定されます。

負荷テストを実行するプロジェクトのタイプと関係のある、要約測定値が、追加のグラフに表示されます。たとえば、Web アプリケーションをテストしている場合は、Web ページのレスポンス時間測定値がグラフに表示されます。

要約テーブル

このセクションには、要約測定値 (すべての仮想ユーザーの測定値の集約値) が表形式で表示されます。 最初のテーブルには、実行されたトランザクションの数や発生したエラーの数などの、全般情報が表示されます。 それ以降のテーブルには、テストしたアプリケーションの種類に関連する要約情報が含まれます。

ユーザー タイプ

このセクションには、ユーザー グループごとに、詳細な測定値が表形式で表示されます。 測定値には、トランザクションのレスポンス時間、個々のタイマ、カウンタ、およびテストしているアプリケーションのタイプ (Web、データベース、CORBA、または TUXEDO) と関係のあるレスポンス時間とスループットの測定値が含まれます。 また、すべてのユーザー グループのエラーおよび警告が、一覧表示されます。

カスタム グラフ

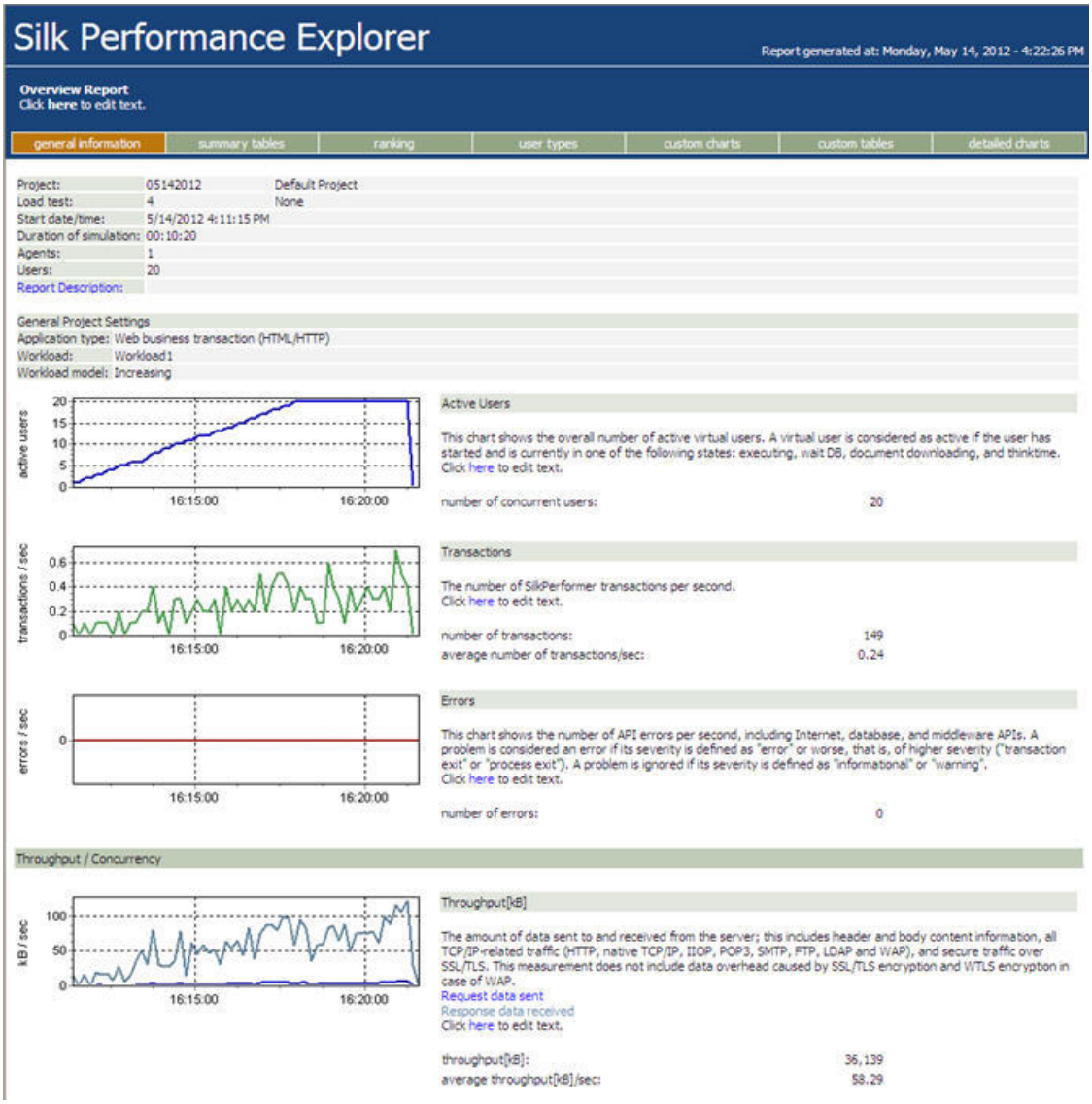
このセクションには、手動で追加したグラフが表示されます。 いつでもこのセクションに、グラフを追加したり、削除できます。 変更をテンプレートとして保存して、すべての要約レポートで表示できます。

カスタム テーブル

このセクションには、手動で追加したテーブルが表示されます。 いつでもこのセクションに、テーブルを追加したり、削除できます。 変更をテンプレートとして保存して、すべての要約レポートで表示できます。

詳細グラフ

このセクションには、レポートに含まれるグラフを拡大したグラフを提供します。 拡大されたグラフにジャンプするには、縮小されたグラフをクリックします。逆に、縮小されたグラフにジャンプするには、拡大されたグラフをクリックします。



概要レポートを表示する

1. ワークフロー バーの **結果の検討** ボタンをクリックします。 **ワークフロー - 結果の検討** ダイアログが表示されます。
2. **Silk Performance Explorer** ボタンまたはリンクをクリックします。 **設定 > オプション > レポート作成** ダイアログで **概要レポートを自動的に生成する** チェック ボックスをオンにしてある場合、Performance Explorer が開き、最新の負荷テストの概要要約レポートが表示されます。 また、このダイアログでは、概要レポートの生成用に以前格納したテンプレートを使用することも選択できます。 この設定はグローバル設定であり、使用しているワークロード プロジェクトに関係なく Performance Explorer で使用されます。
3. 概要レポートが自動的に表示されない場合は、ワークフロー バーの **概要レポート** ボタンをクリックします。
   a) 作業する負荷テストのディレクトリに移動し、対応する .tsd ファイルを選択し、**開く** をクリックします。
   b) **次へ** をクリックします。
   c) 必要に応じて、**テンプレート** フィールドで参照 (**...**) ボタンをクリックし、使用するテンプレート ファイルを探します。
   d) **終了** をクリックします。

e) 選択した負荷テストによっては、関係のあるすべてのファイルをプロジェクトにロードするのか確認するメッセージが表示される場合があります。

概要レポートが開きます。

索引

A

C

J

P

T

W

X

え

か

き

く

け

れ

わ