Microsoft® Visual Studio® Team Foundation Server 2010

Team Foundation Server による ソースコード管理入門

著作権

このドキュメントに記載されている情報は、このドキュメントの発行時点におけるマイクロソフトの見解を反映したものです。マイクロソフトは市場の変化に対応する必要があるため、このドキュメントの内容に関する責任を問われないものとします。また、発行日以降に発表される情報の正確性を保証できません。
このホワイトペーパーは情報提供のみを目的としています。明示、黙示、または法令に基づく規定に関わらず、これらの情報についてマイクロソフトはいかなる責任も負わないものとします。
この文書およびソフトウェアを使用する場合は、適用されるすべての著作権関連の法律に従っていただくものとします。このドキュメントのいかなる部分も、米国 Microsoft Corporation の書面による許諾を受けることなく、その目的を問わず、どのような形態であっても、複製または譲渡することは禁じられています。ここでいう形態とは、複写や記録など、電子的な、または物理的なすべての手段を含みます。ただしこれは、著作権法上のお客様の権利を制限するものではありません。
マイクロソフトは、この文書に記載されている事項に関して、特許、申請中特許、商標、著作権、および他の知的財産権を所有する場合があります。別途マイクロソフトのライセンス契約上に明示の規定のない限り、このドキュメントはこれらの特許、商標、著作権、またはその他の知的財産権に関する権利をお客様に許諾するものではありません。
別途記載されていない場合、このドキュメントで使用している会社、組織、製品、ドメイン名、電子メール アドレス、ロゴ、人物、場所、出来事などの名称は架空のものです。実在する商品名、団体名、個人名などとは一切関係ありません。

目次

■これからのソース管理ツール

現在、あなたが携わっている開発では、ソース コードの管理はどのように行っているでしょうか? ファイル サーバーに日々のフォルダーを作成して管理していますか? 1 人で開発しており、自端末のフォルダーにしかソースが存在しないでしょうか? もしくは Visual SourceSafe (以下、VSS) などのソース管理ツールを利用した管理を行っているでしょうか?

そもそもなぜソース管理ツールの導入が必要なのでしょうか。ファイル サーバーにソースを格納しているだけの場合、作業日ごとにフォルダーを分けるなどして管理していることが多いのではないでしょうか。これでも問題ないかもしれませんが、ファイルを格納する際に他の人の変更内容を上書きしてしまって、せっかく修正した内容が消えてしまったり、そもそも最新のファイルがどこにあるのかわからなくなってしまったりした経験はありませんか? また、自端末にしかソースが存在していない状況で HDD がトラブルに見舞われてソースが消えてしまったことはありませんか? このような事態を防止し、無駄な作業を発生させないようにするために、最新のソースの場所を一元管理し、だれがどのように変更したかの履歴を管理し、データを安全にバックアップしておく、ソース管理ツールが必要です。

●Visual SourceSafe から Team Foundation Server へ

VSS はこれからソース管理ツールを導入しようというユーザーにはハードルも低く、簡単に利用を始めることができるため、根強く利用されてきました。しかし一方で、管理がファイル サーバーであるためのパフォーマンスの問題や、リポジトリ容量が約 4 GB に制限されてしまうという問題も抱えていました。そこで、マイクロソフトでは VS 2005 をリリースしたときから、VSS に代わる新しいソース管理ツールとして Team Foundation Server (以下、TFS) をリリースしています。簡単な比較表を表 1 に示しますが、TFS は VSS が抱える問題をすべて解決してくれるだけのポテンシャルを持っていたため、かなりの期待を持たれてきたのは間違いありません。

項目	TFS	VSS
対応範囲	ソース管理を含むソフトウェア ライフサイクル全般の統合	ソース管理のみ
データ管理方法	SQL Server を利用	ファイル サーバーを利用
容量制限	なし	約 4 GB
同時利用ユーザー数	数千名規模を想定	20 名前後を想定
セキュリティ	Windows ユーザー認証に統合された多重防御	独自ユーザーによるリポジトリ アクセス権設定のみ
リモート アクセス	最適化された Web サービス 遠隔地用にキャッシュを利用可能	HTTP 経由の簡易的なアクセス

表 1: TFS と VSS の比較

しかし、TFS はソース管理以外にもさまざまなことができる製品であるため、利用が難しいという問題を含んでいま

した。また、インストールから導入までが VSS と比較すると難しい、価格も高いといったことがあり、これらも利用を妨げる原因となっていました。このため、今まではなかなか利用されずに見送られてきたのが現実です。
しかし、次バージョンである TFS 2010 は違います。詳細は後述しますが、MSDN Subscription で提供されたり、クライアント OS にインストールすることができるなど導入の難易度は VSS と同様になりました。本稿では、TFS を使ったソース コード管理の実践方法を詳細にわたって解説していくので、ぜひご活用ください。

なお、本稿の内容は Visual Studio 2010 日本語版 β2 および Team Foundation Server 2010 日本語版 β2 を基にしているため、スクリーンショットの一部に英語表記が含まれることがありますが、製品版では日本語表示になるのでご了承ください。

●TFS を使ったソース管理

先述のとおり、TFS にはさまざまな機能が用意されていますが、本稿の解説はソース管理を主軸にしています。ことソース管理に限ってしまえば、実はそんなに複雑なことはありません。今回はひとまずソース管理に限ってその全容をつかんでおきましょう。TFS 2005 や 2008 を利用したことがある場合にはおさらいとなりますが、利用したことがない場合には、この機会に把握しておいてください。

まずは図 1 をご覧ください。

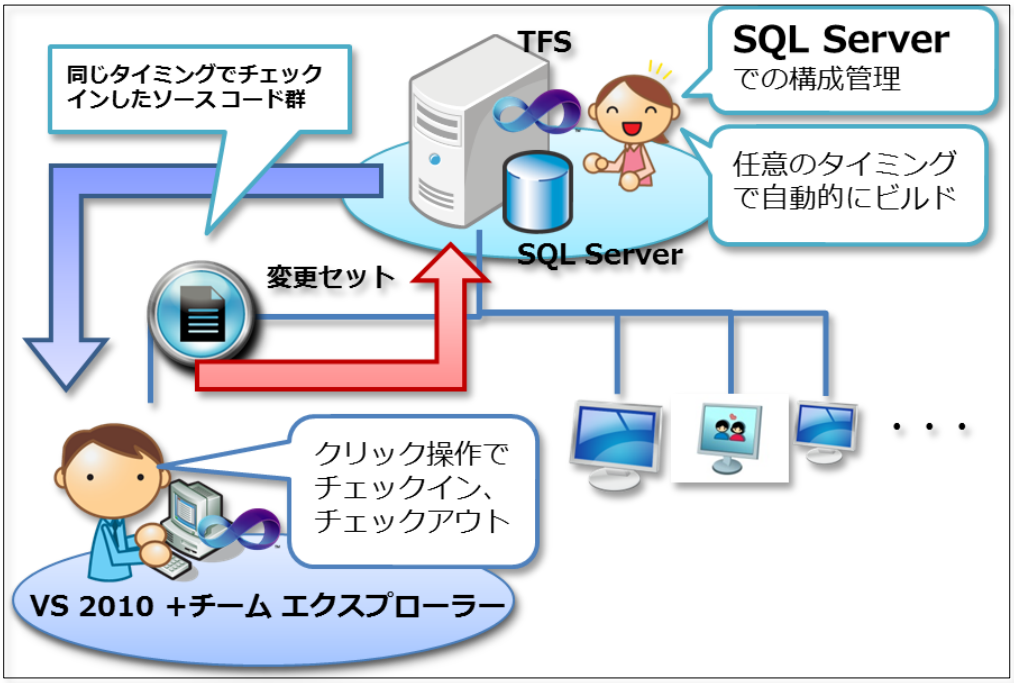


図 1: TFS によるソース管理の全体像

ソース管理を利用するユーザーは、VS 2010 とチーム エクスプローラーを使って TFS に接続しています。チェックイン、チェックアウトはマウス操作だけでできます。チェックインは、変更セットという単位で実行されます。変更セットについては後述しますが、簡単に言うとチェックインされたファイル群を管理する 1 つの固まりとなっているものです。リポジトリは前述の表 1 のとおり、ファイル サーバーではなく SQL Server が利用されています。さらにもう 1 つ、任意のタイミングで自動的にビルドという説明があります。これについても後述しますが、TFS では設定に基づいて任意のタイミングでビルドを実行させることもできます。

いくつか聞き慣れないものも出てきてはいますが、既に何かしらのソース管理ツールを使っているのであれば、今

までと大差ないことがわかります。ファイル サーバーで管理されているのであれば、いちいち Windows エクスプローラーを開かずとも VS 上からすべての操作が行える便利さに興味を持っていただけるはずです。では、ここから先の章では、この図に示した方法を実践していく方法を確認していくことにしましょう。

[コラム] チェックインとチェックアウトとは

チェックアウトとは、ソース管理サーバーに登録されているファイルを自端末で編集するために、編集許可を求める操作のことです。

チェックインとは、チェックアウトして編集したファイルを、編集が終わったため最新の状態でソース管理サーバーに登録する操作のことです。

■ソース管理の日々の使い方 (TFS 利用者のために)

本来、TFS によるソース管理を使い始めるためには、環境の準備が必要となります。しかし、その点は「ソース管理を始めるには (TFS 管理者のために)」で後述することにします。ひとまず利用可能な環境が整っていることを前提として、1 人の利用者という視点で日々の作業を行っていく方法を覚えてしまいましょう。すべての方法はステップバイ ステップで説明していくので、この章の内容を覚えてしまえば、もう TFS での管理もお手の物となるはずです。

●事前設定

まずは、TFS をソース管理サーバーとして利用するための VS の設定を確認しておきましょう。VS の [ツール] メニューの [オプション] をクリックしてください。図 2 のような [オプション] 画面が開くので、左のツリー メニューから [ソース管理] – [プラグインの選択] を選択します。ここで [現在のソース管理プラグイン] に Visual Studio Team Foundation Server が選択されている状態にします。特に VSS など TFS 以外のソース管理ツールを利用している場合には選択肢がいくつか表示されるので注意してください。

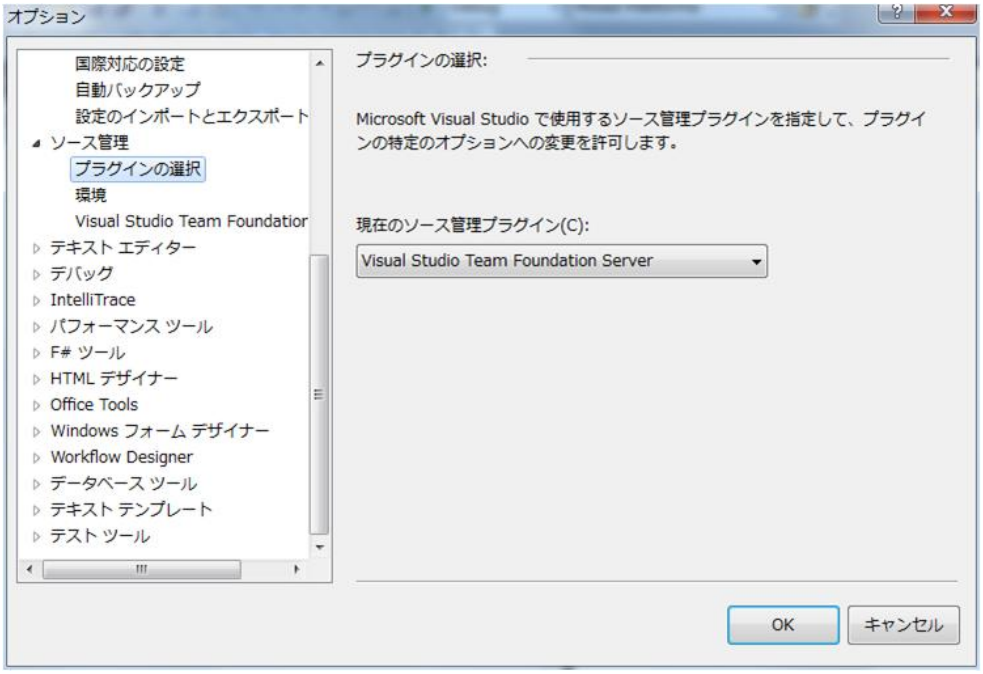


図 2: ソース管理プラグインの設定

プラグインを設定すると、左側のツリー メニューに [ソース管理] – [Visual Studio Team Foundation Server] という項目が表示されるようになるので、これを選択します。[ファイルのダウンロードにプロキシ サーバーを使用する] という項目のチェック ボックスがオフになっていれば、その他の項目は任意設定でかまいませんが、[チェックアウト時に項目の最新バージョンを取得する] という項目のチェック ボックスはオンにしておくと利便性が向上するので設定しておくことをお勧めします (図 3)。

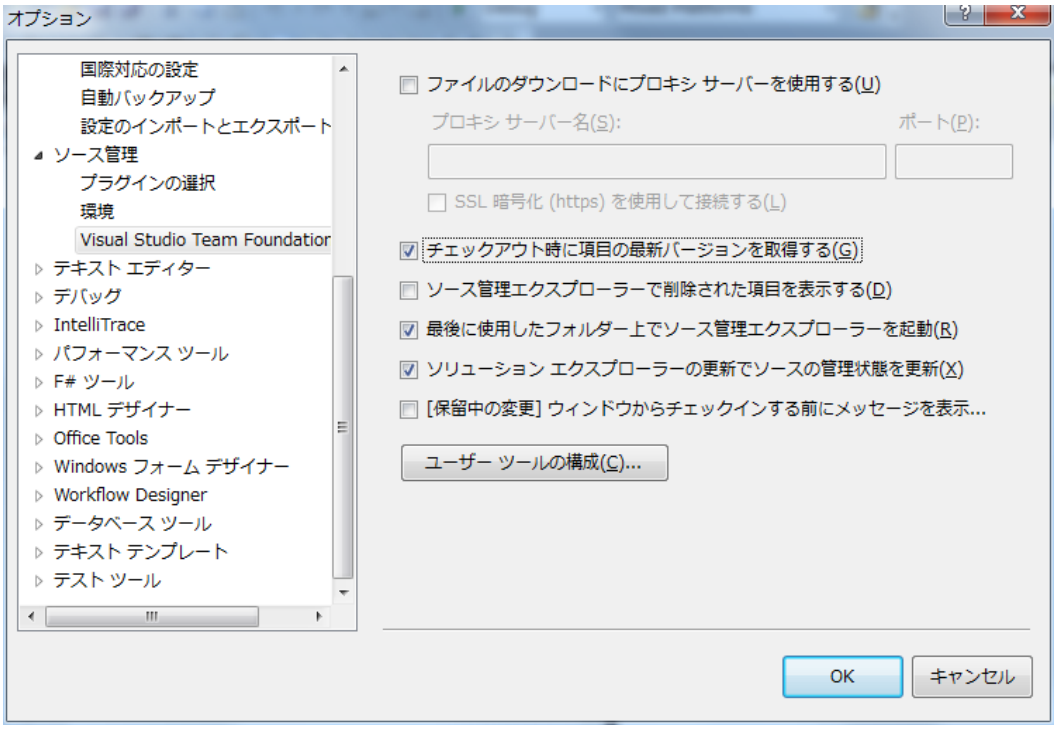


図 3: Visual Studio Team Foundation Server プラグインの設定

[コラム] 最新バージョンの取得の設定について

この設定は、後述するチーム プロジェクトのソース管理の設定で、チェックアウト時の最新バージョンの取得が強制されていない場合に有効となります。もし、最新バージョンの取得が強制されている場合には、この設定にかかわらず、チェックアウト時には最新バージョンが取得されることになります。

●チーム プロジェクトへの接続

TFS を使うための設定が完了したら、今後 VS 起動後にまず行うべきことはチーム プロジェクトというプロジェクト管理のための領域に接続することです。一度設定してしまえば次からは VS の起動と同時に行われるようになりますが、これから毎日使うものなので、しっかり手順を押さえておきましょう。

VS を起動したら、[チーム] メニューの [Team Foundation Server に接続] をクリックします。図 4 のように [チーム プロジェクトへ接続] というウィンドウが表示されます。

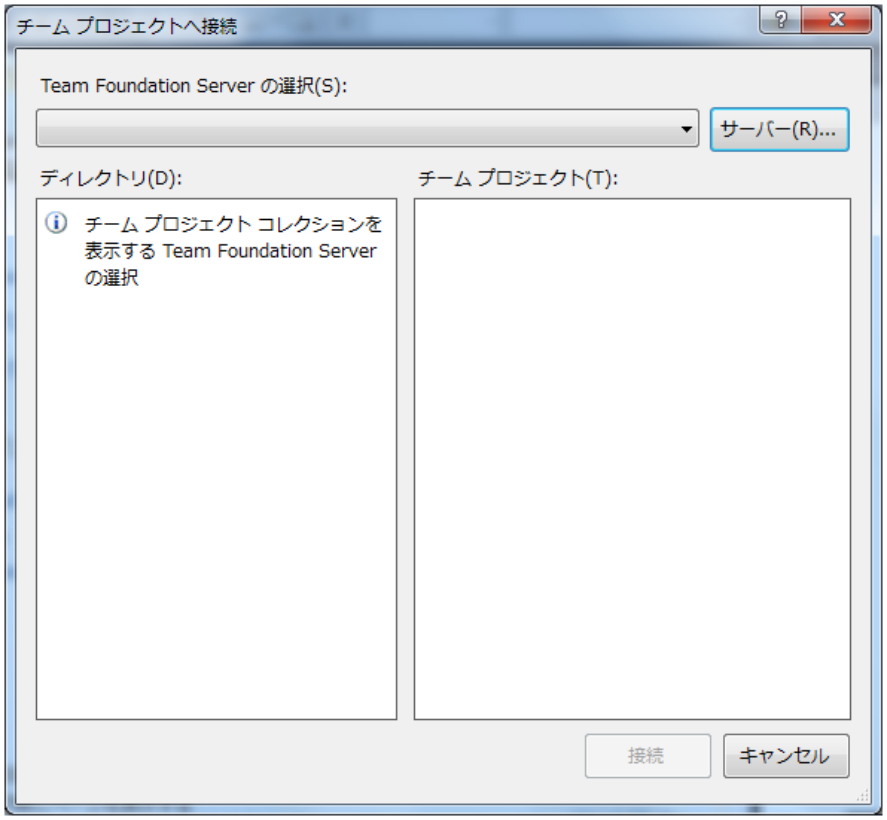


図 4: チーム プロジェクトへの接続ウィンドウ

ここで [サーバー] をクリックしてください。[Team Foundation Server の追加および削除] ウィンドウが表示されるので、[追加] をクリックします。[Team Foundation Server の追加] ウィンドウが表示されるので、[Team Foundation Server の名前または URL] ボックスに TFS サーバー名を入力します。入力を行うと、たとえば図 5 のような状態になります。

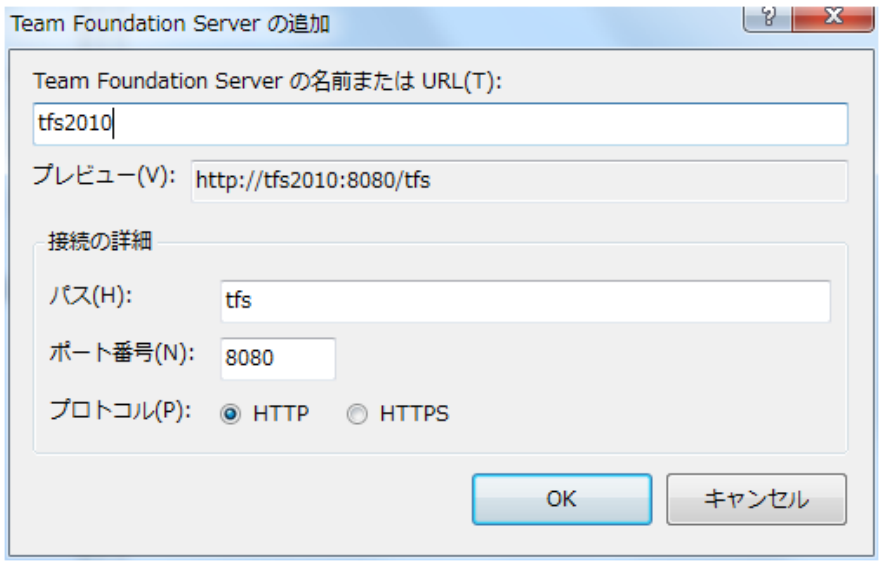


図 5: TFS 名の入力

他にも設定できる項目はありますが、ひとまずはこのままでかまいません。入力が終わったら [OK]、戻った画面では [閉じる] をクリックし、[チーム プロジェクトへ接続] の画面まで戻ります。

戻ってきた画面では、図 6 に示すようにドロップダウン リストに追加した TFS サーバーが設定された状態になっています。

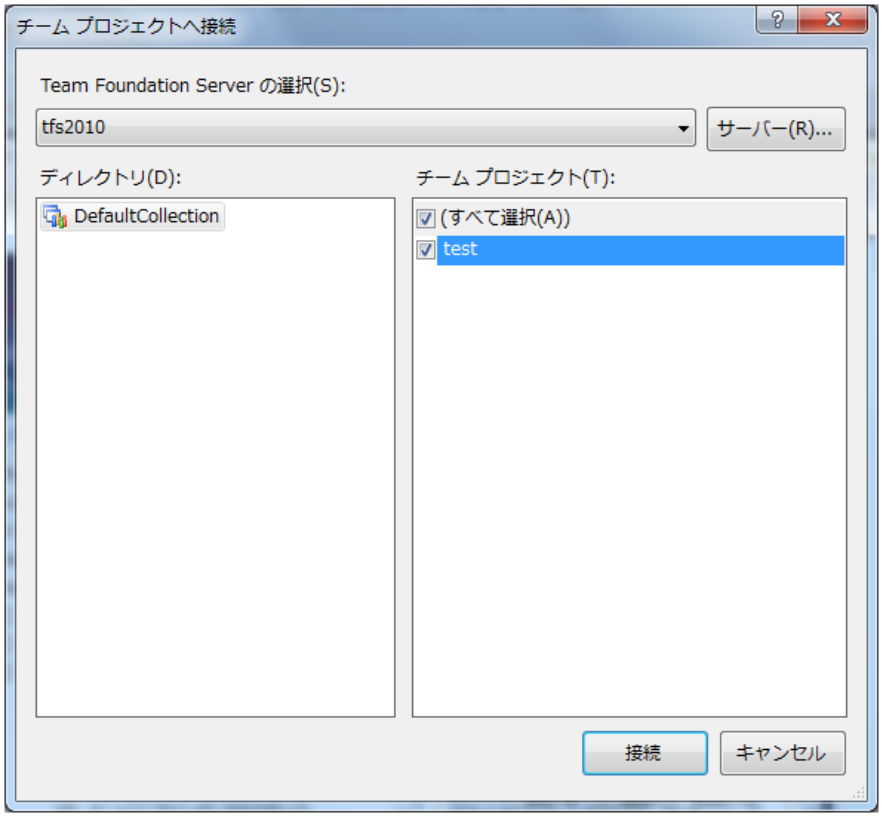


図 6: チーム プロジェクトの選択

下段右側にはチーム プロジェクトの一覧が表示されているので、ここから指定されたチーム プロジェクト (図 6 では test) のチェック ボックスをオンにして、[接続] をクリックしてください。画面は VS に戻り、標準ではソリューション エクスプローラーが表示されている場所と同じ部分に、チーム エクスプローラーが表示されます (図 7)。

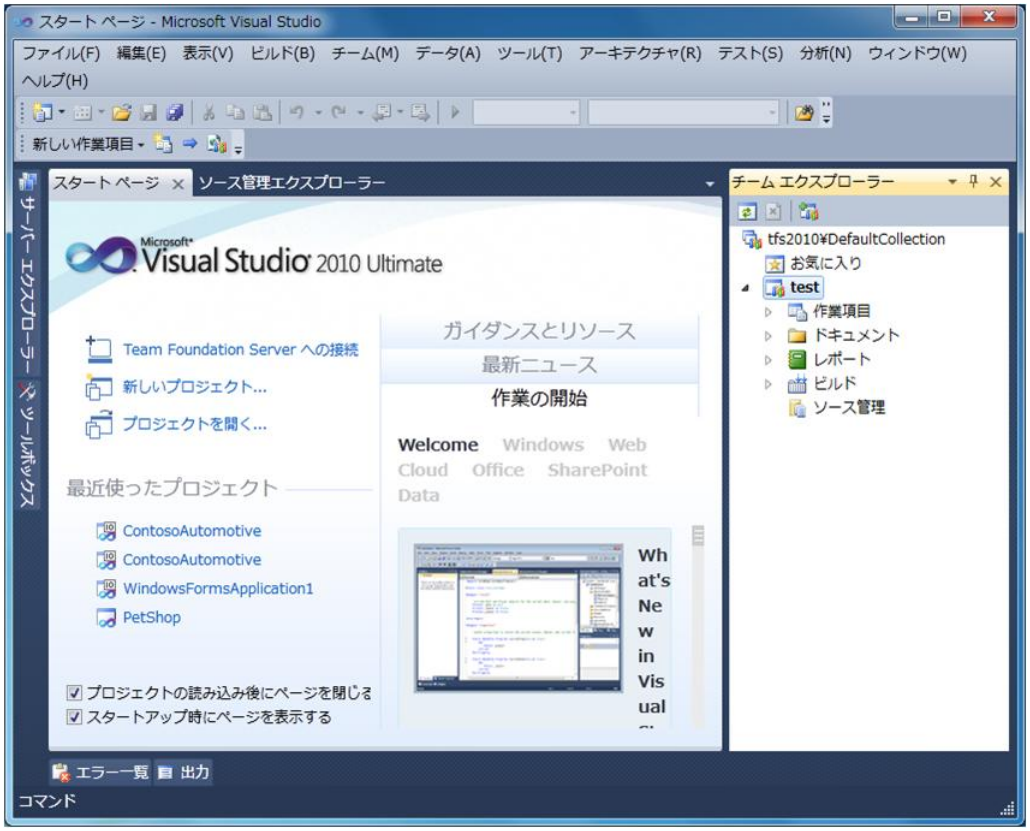


図 7: チーム エクスプローラーの表示

チーム エクスプローラーは VS を使ってチーム プロジェクトに接続する場合のクライアント ソフトで、チーム プロジェクトに対するすべての操作はここから行えるようになっています。今後、ソース管理にとどまらずさまざまな場面で登場するので、よく覚えておいてください。

[コラム] TFS が Active Directory で管理されていない場合

TFS は通常、Active Directory を必要としますが、小規模構成などの場合には、なくても動作させることができます。ただし、その場合には、図 5 で TFS サーバー名を入力した後に認証ダイアログ ボックスが表示されることがあります。その場合には、TFS サーバーに登録されているユーザー名とパスワードを入力して認証する必要があります。

[コラム] 旧バージョンの TFS への接続

VS 2010 からは TFS 2010 だけでなく、TFS 2008 にも接続することができます (残念ながら TFS 2005 には接続できません)。この場合、図 5 の TFS 名を入力するウィンドウで、[接続の詳細] の [パス] ボックスを空白に設定します。

●ワークスペースの設定

チーム プロジェクトに接続できたら、次はソース コード管理のための準備です。TFS では、サーバー上のフォルダーと自端末のフォルダーとをマッピングする (関連付ける) ことを「ワークスペースのマッピングを作成する」と言います。VSS でも「作業フォルダーの設定」という設定項目がありましたが、それとほぼ同様です。ワークスペースをマッピングする場合も、サーバーの特定の場所から初めてファイルを取得する場合に一度設定を行えば、以降は

その設定のもとで作業が行えるようになります。
ここでは、初めてワークスペースを作成する場合と、作成済みのワークスペースを編集する場合の 2 点について確認しておくことにします。

○ **ワークスペースの作成**

では、実際に設定してみましょう。まずはチーム エクスプローラーを表示し、特定のチーム プロジェクトに接続されている状態であることを確認してください。チーム プロジェクトを展開すると図 8 の右側のように [ソース管理] という項目があるので、これをダブルクリックします。

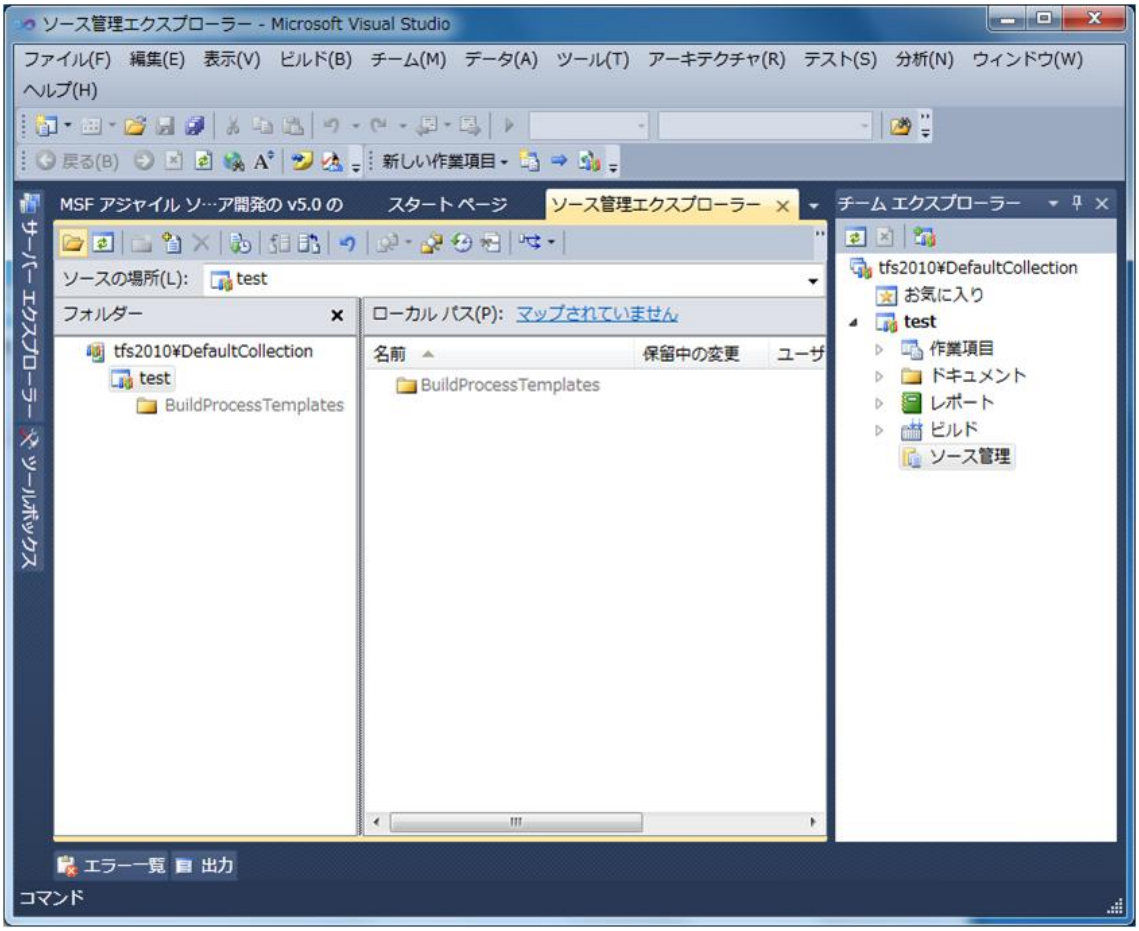


図 8: ソース管理エクスプローラーの表示

すると図 8 の左側のようにソース管理エクスプローラーが表示されます。

ソース管理エクスプローラーは TFS のソース管理機能によって管理されているファイルの一覧を閲覧するためのもので、VSS を起動したときの画面とよく似ています。操作方法もよく似ているため、VSS に慣れている利用者であればここから欲しいファイルを自端末にダウンロードしてくることも簡単だと思いますが、今回は基本のステップということで 1 つずつ順に確認していくことにします。

ソース管理エクスプローラーの画面左側にはフォルダーの一覧、右側にはファイルの一覧が表示されています。ファイル一覧の上部を見ると [ローカル パス] という項目があり、初めて利用する場合、この右側には「マップされていません」と表示されているはずです。マップとは、ワークスペースを使って TFS サーバーのフォルダーと自端末のフォルダーの関連付けが行われていることを示す言葉で、今は何も設定されていないことを示しています。では、「マ

ップされていません」をクリックしてください。[マップ] ウィンドウが表示されました。ここで、サーバー フォルダー (TFS サーバーの該当フォルダー) とローカル フォルダー (自端末の該当フォルダー) をそれぞれ設定します (図 9)。

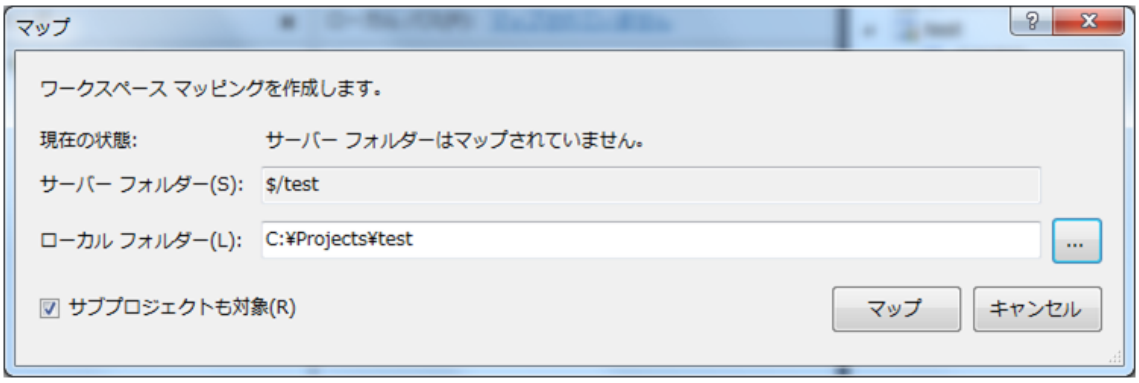


図 9: マッピングの設定

[マップ] をクリックすれば、設定は終了です。新しい設定を行うと、図 10 のような今すぐファイルをダウンロードするかを問うダイアログ ボックスが表示されます。

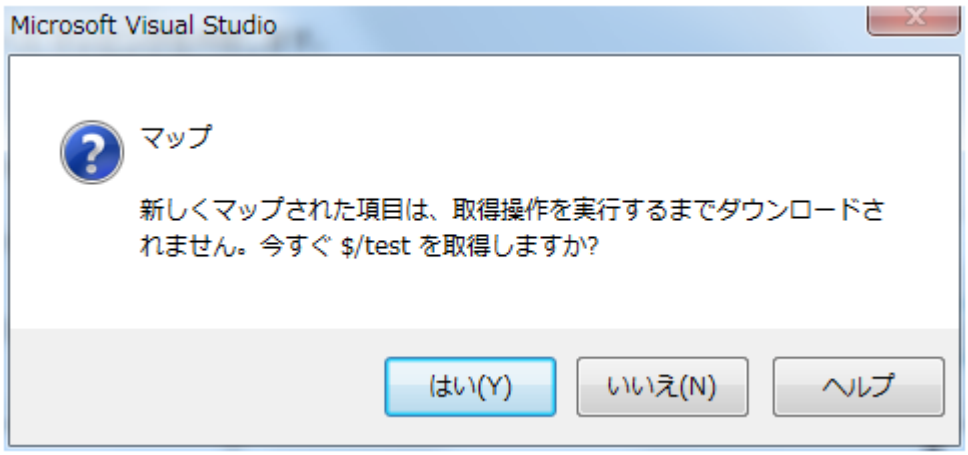


図 10: ファイルの取得確認ダイアログ ボックス

ここは各自の好きな選択肢を選んで問題ありませんが、この後ですぐダウンロードすることになるはずなので [はい] を選んでおくのがよいでしょう。

○ワークスペースの編集

間違ってマップを作成してしまった場合や、マップされている一覧を確認したい場合などには、1 つ 1 つ確認していくよりも一覧で確認できた方が便利です。そこでワークスペースを編集する方法についても覚えておきましょう。

ワークスペースの編集を始めるには、まずソース管理エクスプローラーを表示します。図 11 のように少し広めに表示すると、ソース管理エクスプローラーのツール バー上にワークスペースというドロップダウン リストが見えます。

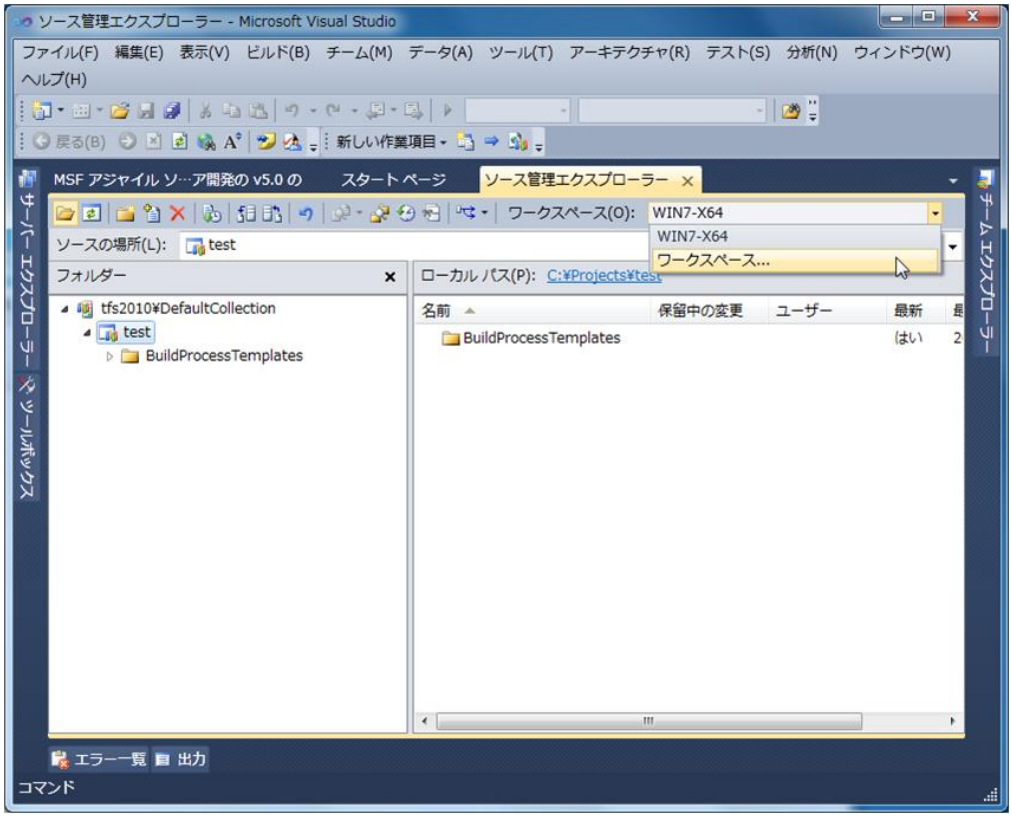


図 11: ワークスペースの編集の開始

このドロップダウン リストには、通常は自端末のコンピューター名が表示されているはずです。ワークスペースの編集を開始するには、このドロップダウン リストからワークスペースを選択してください。すると図 12 のようなワークスペースの管理ウィンドウが表示されます。

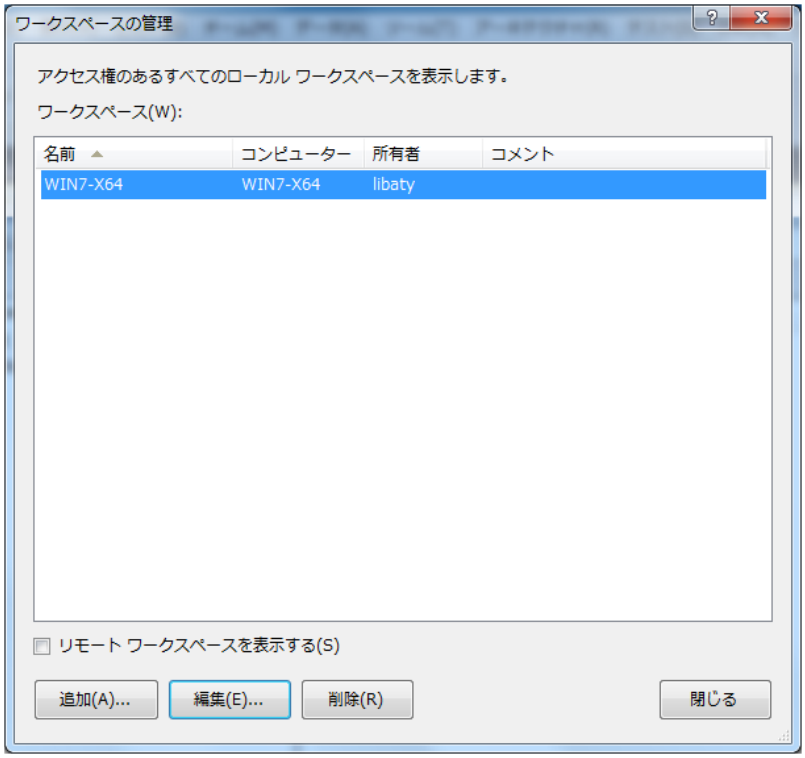


図 12: ワークスペースの管理ウィンドウ

このウィンドウには現在のユーザーが操作可能なワークスペースの一覧、つまり他の端末で同じユーザーで作成したワークスペースも含めた一覧が表示されるので、コンピューター名と所有者の組み合わせから、現在の端末で利用しているワークスペースを特定するようにしてください。編集するワークスペースを特定できたら [編集] をクリックします。今度は、図 13 のような [ワークスペース <ワークスペースの名前> の編集] ウィンドウが表示されます。

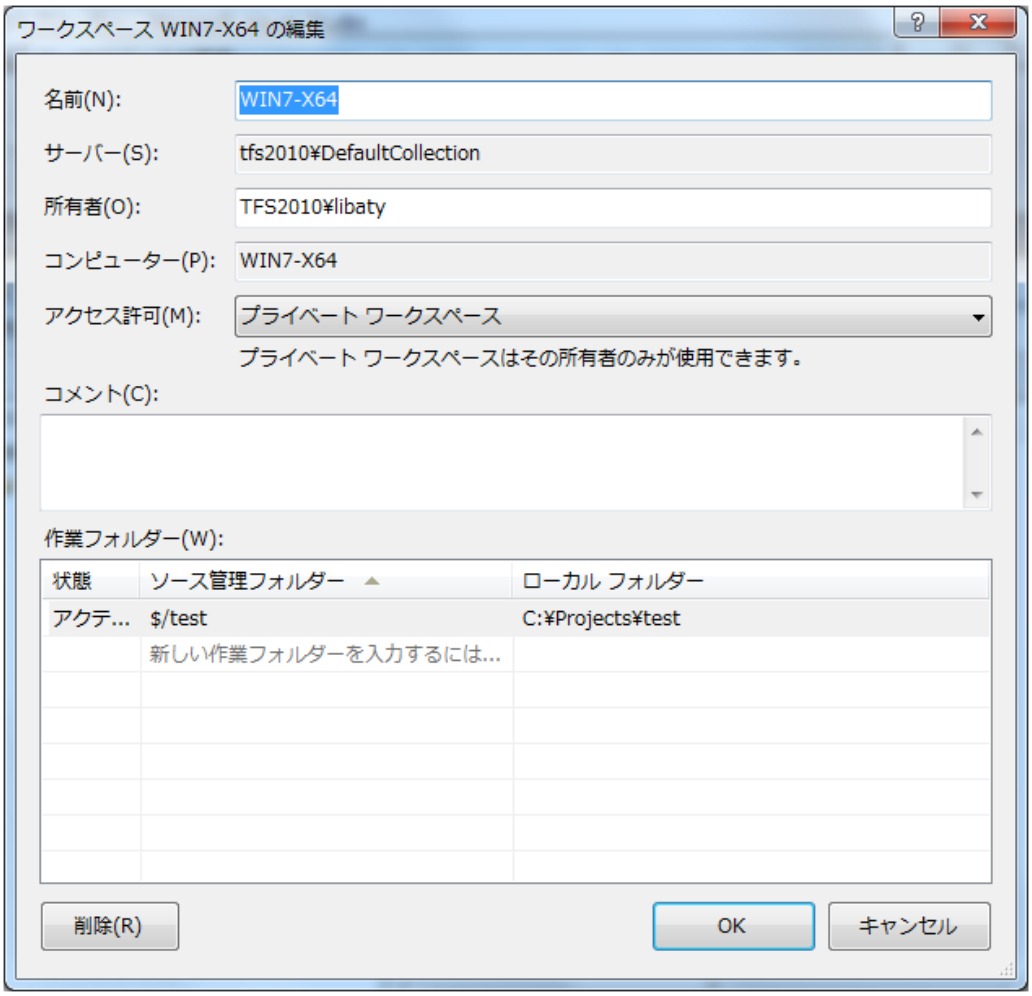


図 13: [ワークスペース <ワークスペースの名前> の編集] ウィンドウ

ここでは、ワークスペースの名前や所有者、アクセス許可などを編集できますが、注目すべきは作業フォルダーとして一覧表示されている部分です。ここで既存の [ソース管理フォルダー] 列や [ローカル フォルダー] 列に設定されている内容を編集することでマッピング情報を修正することができます。また、新規に行を追加すれば新しいマッピングを作成することもできます。さらに、特定の行を選択した状態で左下の [削除] をクリックすることで、既存のマッピングを削除することもできます。

なお、ワークスペースの編集の詳細については ワークスペースの操作 [http://msdn.microsoft.com/ja-jp/library/ms181383(VS.100).aspx] を参照してください。

●ソリューションの取得

ワークスペースの設定ができたら、ソース管理エクスプローラーに登録済みのソリューションを取得しましょう。

実は、ワークスペース作成後の最後の問い合わせダイアログ ボックスで [はい] を選んでいる場合には既に最新のソリューションを取得できてしまっていますが、そうではないことを想定して手順を確認しておきましょう。また、最後にちょっとした Tips を紹介するのでそちらも併せて確認しておいてください。

○ソリューションファイルの取得方法

まずはソース管理エクスプローラーを開いてください。

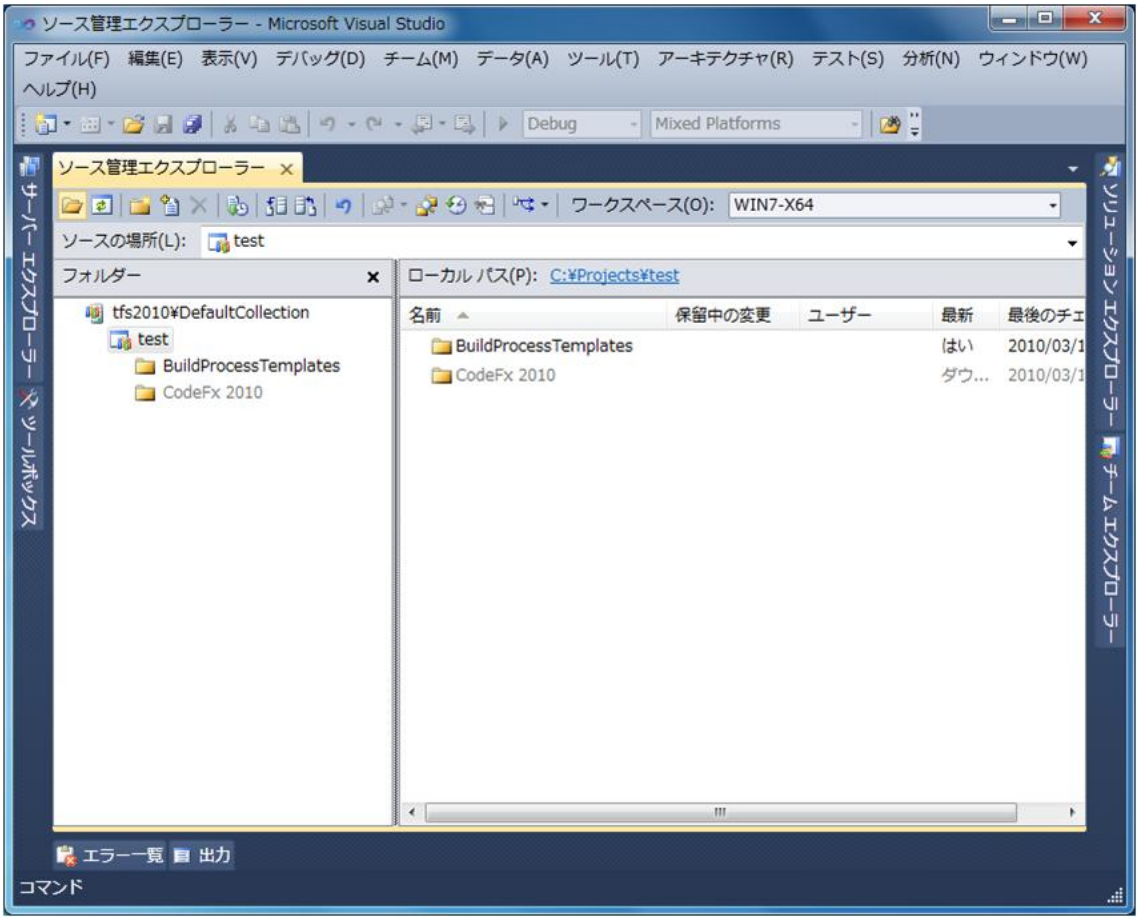


図 14: 未取得ファイルがある場合の表示

図 14 のフォルダー階層をよく見てみると、色の濃い部分（黒色）と薄い部分（グレー色）があるのがわかるでしょうか。色が濃い部分は既に自端末にファイルを取得済みの部分で、薄い部分はまだファイルを取得していないことを示しています。では、図 14 でグレーになっている部分を例に未取得のファイルを取得する方法を確認しましょう。これは非常に簡単です。グレーのフォルダーを選択し、コンテキスト メニューから [最新バージョンの取得] を選択するだけです。既にワークスペースでマッピングが設定されているため、自端末の指定されているフォルダー内にファイルがダウンロードされます。ファイルのダウンロードが終わると、図 15 のようにグレーであった部分も濃い色に変更されます。

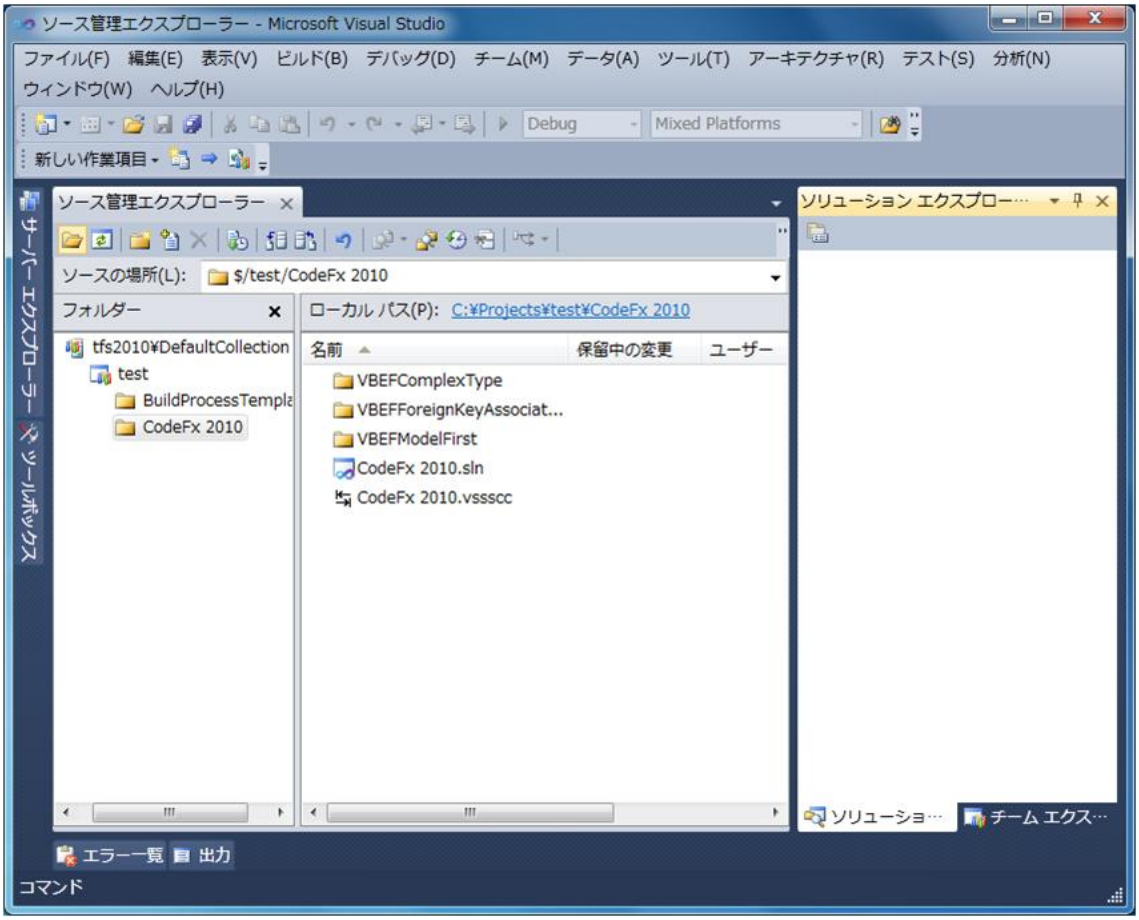


図 15: ファイル取得後の状態

最後に、開きたいソリューション ファイル (この場合は「CodeFx 2010.sln」ファイル) をダブルクリックするなどして開くことで、そこから先はいつもどおりの開発を始めることができます。

○誤ってローカル フォルダーのファイルを削除してしまった場合の対処

ここでは、ソース管理エクスプローラーから最新のファイルを取得した後で間違ってローカル フォルダー内のファイルを削除してしまった場合の対処方法を紹介します。この状況は、図 16 の「ReadMe.txt」のように VS のソリューション エクスプローラーではファイルが見つからない状態として報告されます。

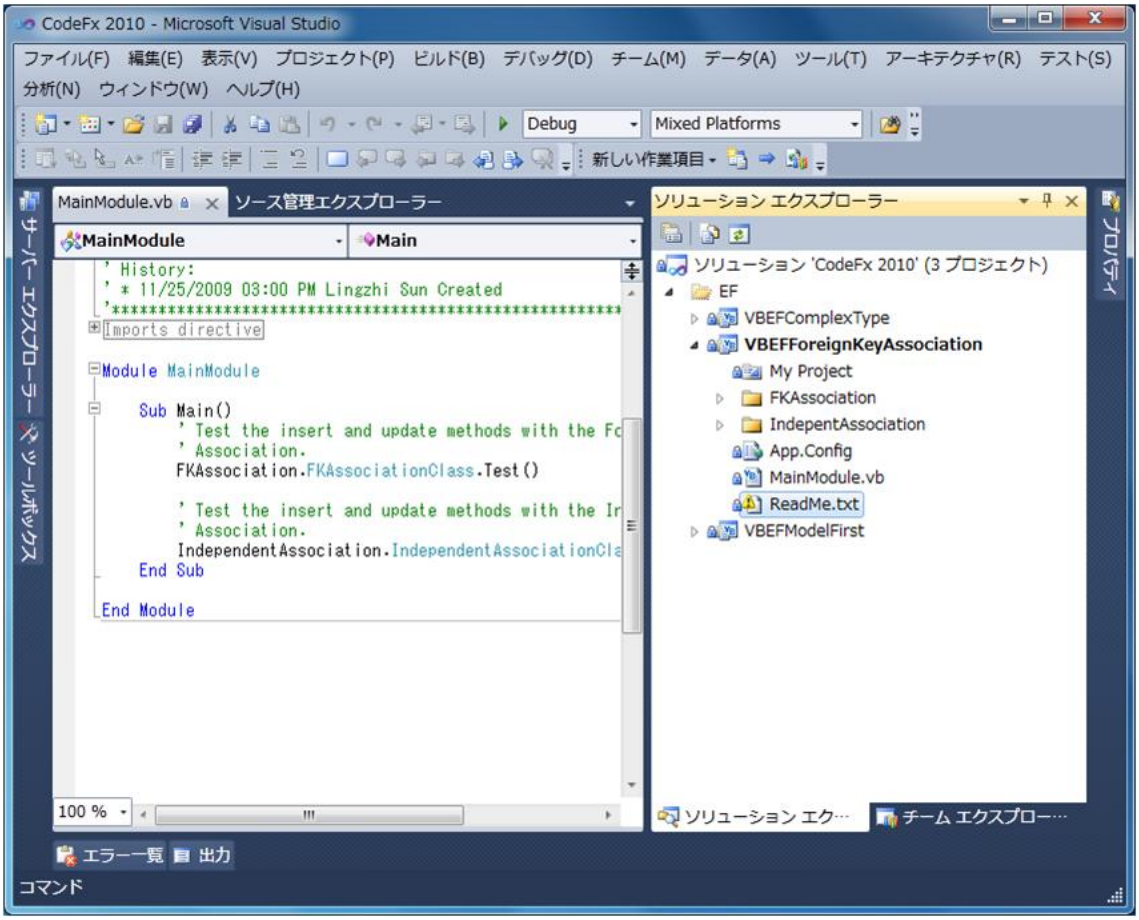


図 16: ファイル パス不明の状態

このとき、「ReadMe.txt」のコンテキスト メニューから [最新バージョンの取得] を選択してみてください。すると、ファイルの取得中ダイアログ ボックスが一瞬表示されてすぐに取得が終わったように見えます。しかし、Windows エクスプローラーで確認してみるとファイルの取得が行われていないことがわかります。このようなことになってしまうのには理由があります。

これは、TFS によってどの端末がどのファイルのどのバージョンをダウンロードしたかが記憶されているためです。さすがにローカル フォルダーからファイルが削除されたことまでは把握できないために、このようなミスマッチが発生してしまいます。このような状態になってしまったときには、[最新バージョンの取得] ではなく [特定バージョンの取得] を選択しましょう。[特定バージョンの取得] は、本来は過去のバージョンのファイルを取得するためのコマンドですが、ここで図 17 のようにバージョンの種類を [最新バージョン] に設定し、[ローカル バージョンが指定のバージョンと一致する場合でもすべてのファイルを上書きする] チェック ボックスをオンにして取得することで、強制的にファイルを取得することができます。

図 17: 特定バージョンのファイルの取得

●チェックアウトとチェックイン

ワークスペースを作成し、ファイルを取得できれば、いよいよ開発のための準備も完了です。後は自分の実力を十二分に発揮してソース コードを記述していく作業です。ここでは、ファイルの編集を行い、結果をソース管理に登録するという点で基本的な操作といくつか用意されている機能を確認しておきましょう。

○ファイルのチェックアウト

なにはともあれ編集を始めるためには、ファイルをチェックアウトする必要があります。ファイルをチェックアウトするには、ソリューション エクスプローラーでチェックアウトしたいファイルのコンテキスト メニューから [編集用にチェックアウト] を選択します。すると、図 18 のようなチェックアウトの確認ダイアログ ボックスが表示されます。

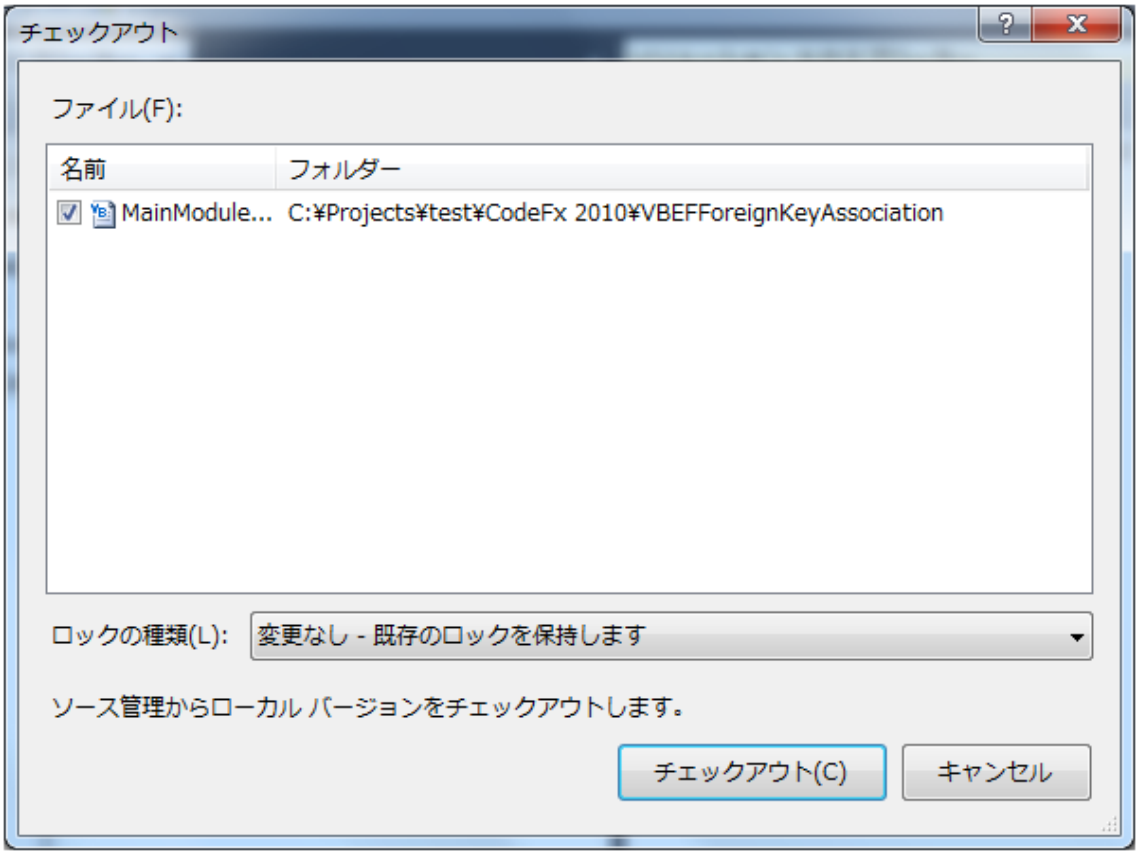


図 18: チェックアウトの確認ダイアログ ボックス

ここでのポイントは、[ロックの種類] というドロップダウン リストです。ロックの種類には▲表 2 に示すとおり 3 種類のモードが用意されていますが、TFS の標準は「変更なし」となっています。

種類	解説
変更なし	ソース管理サーバー上で該当ファイルのロックをしません。他のユーザーがファイルのチェックアウト、チェックインを行うことができます。
チェックアウト	ソース管理サーバーで排他ロックをかけます。他のユーザーがファイルのチェックアウト、チェックインのどちらも行うことができなくなります。
チェックイン	ソース管理サーバー上でチェックイン予約ロックをかけます。このため他のユーザーがファイルのチェックアウトを行うことはできますが、チェックインを行うことはできません。

▲表 2: ロックの種類

VSS の利用経験がある場合には、ロックの種類として [チェックアウト] を利用するのが一番しっくりくるでしょう。VSS はチェックアウト時には必ずロックをかけ、チェックアウトしたユーザーがチェックインして初めてロックが解除されるというモデルが利用されていました。しかし昨今は、チェックアウト時にロックはかけず、チェックインをする際に他のユーザーによる変更がないかを確認し、衝突 (コンフリクト) があった場合に、お互いの差分をマージしてチェックインすることで、開発の速度を向上させようという仕組みが主流になっているため、TFS でも同様のモデルを標準として利用するように設定されています。

長々と仕組みを説明してきましたが、[チェックアウト] ダイアログ ボックスでロックの種類を標準の設定から変更しない場合には、もっと簡単にチェックアウトをすることができます。それは、ソース コードを VS のコード エディター上に開き、そのまま編集を始めてしまう方法です。すると、VS が自動的にロックの種類に「変更なし」を選択したものと見なして、ファイルをチェックアウトしてきます。

今まで VS 上でソース管理ツールを利用したことがない方のために補足しておくと、いずれの方法でチェックアウトを行っても、ソリューション エクスプローラーでは図 19 の「MainModule.vb」ファイルのようにチェックアウト中であることを示す (赤い) チェック マークが表示されます。

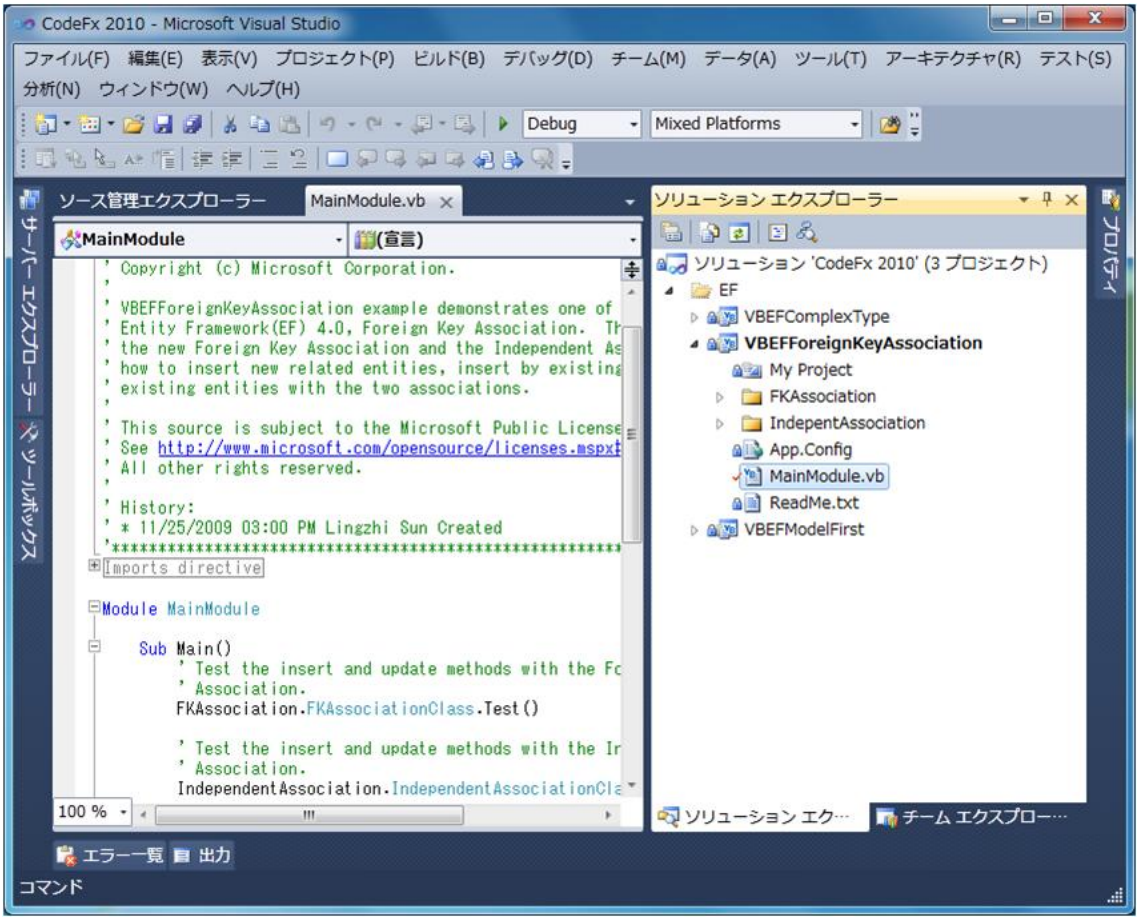


図 19: チェックアウト中であることを示すチェック マークを表示

○ファイルのチェックイン

ファイルの編集が終わったら今度は編集したファイルをチェックインして、変更内容を登録する必要があります。チェックイン操作もチェックアウト同様に簡単で、ソリューション エクスプローラーでチェックインしたいファイルのコンテキスト メニューから [チェックイン] を選択します。すると、図 20 のような [チェックイン] ダイアログ ボックスが表示されます。

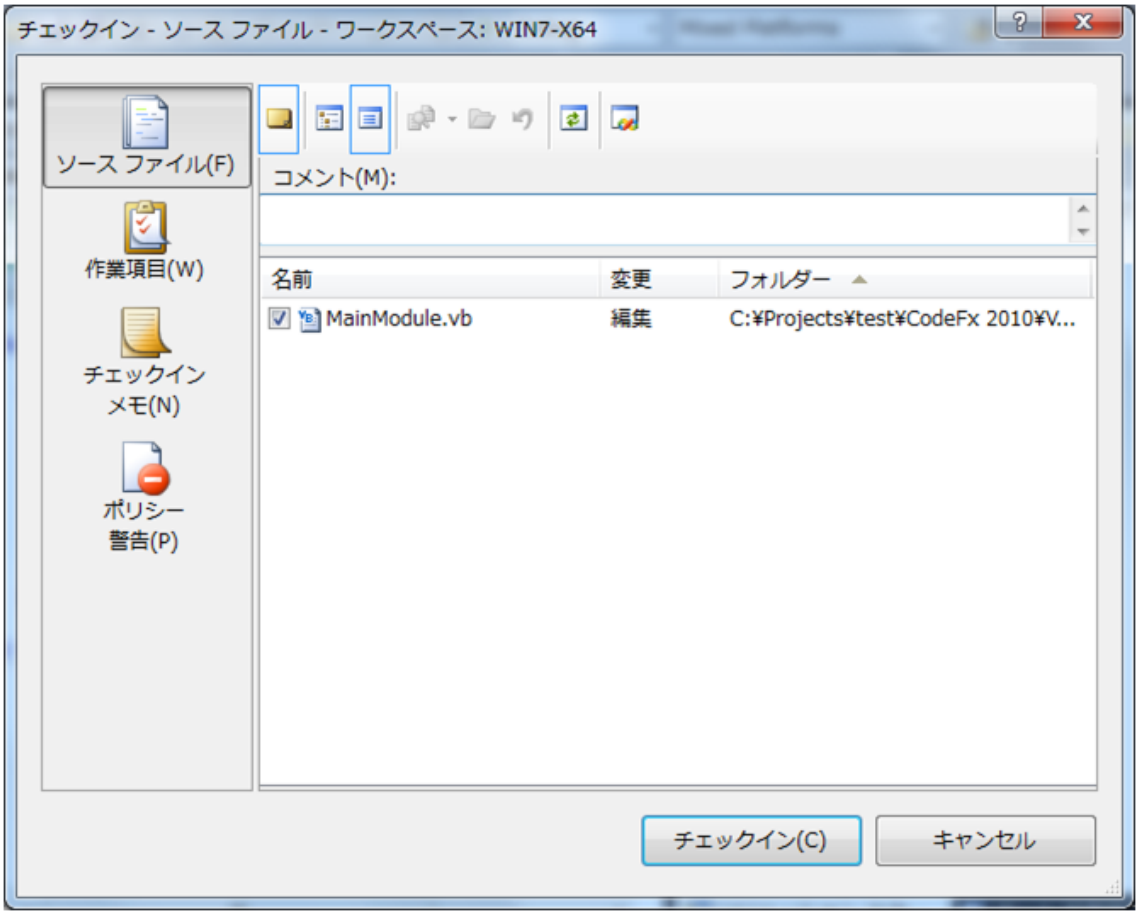


図 20: チェックインの確認ダイアログ ボックス

あまり見慣れない画面かもしれませんが、現時点ではファイルの一覧が表示されている部分と、その上部にあるコメント欄のみ注目してください。ファイルの一覧はまさに今からチェックインしようとしている全ファイルが一覧で表示されます。たとえば、フォルダーやプロジェクト、ソリューションを選択した状態でチェックインを行おうとすると、複数のファイルが表示されることになります。次にコメント欄ですが、たいていのソース管理ツールではチェックイン時にそのチェックインの理由を説明するようなコメントを付けられるようになっています。TFS でも同様で、たとえば変更箇所を解説する文章などを入れておくと後で履歴を検索するときに探し出しやすくなります。ただし、コメントは任意入力項目なので、入力の有無はルール化しておくなどするとよいでしょう。

なお、まったく編集していない状態でチェックイン操作を行うと、図 21 のようなダイアログ ボックスが表示されます。

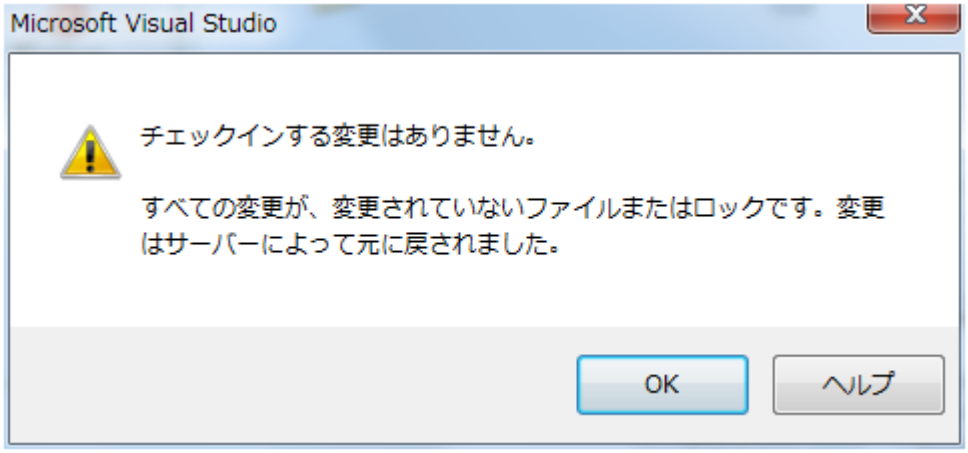


図 21: 未編集でチェックインした場合

これは、変更内容がないため、ソース管理サーバーでファイルにロックがかかっていればロックの解除だけ行うという意味です。特に問題はないので、そのまま [OK] をクリックしてかまいません。しかし、このような場合には変更がないチェックインを行うよりは、チェックアウトの取り消しをした方がよりスマートです。実際に行うには、ソリューション エクスプローラーで対象ファイルのコンテキスト メニューを表示し、[保留中の変更を元に戻す] を選択するだけです。

○保留中の変更の確認

チェックアウトとチェックインは非常に簡単でした。しかし、ファイル数が多くなってくると、どのファイルをチェックアウトしているのか確認するのも一苦労となってきます。そこで、チェックアウト中のファイルの一覧を確認する方法を覚えておきましょう。これは、TFS の機能ではなく、VS にあらかじめ用意されているものなので、既にご存じの方も多いかもしれません。

この機能は [保留中の変更] というウィンドウを表示することで簡単に利用できます。このウィンドウを表示するには、VS の [表示] メニューの [その他のウィンドウ] をポイントし、[保留中の変更] をクリックします。
すると標準では図 22 のように VS の下側に [保留中の変更] ウィンドウが表示されます。ここで、現在チェックアウト中のファイルの一覧を確認することができるため、たとえば何も編集していないはずなのにチェックアウトされているファイルが存在するなどの事態を早期に見つけ出すことができます。

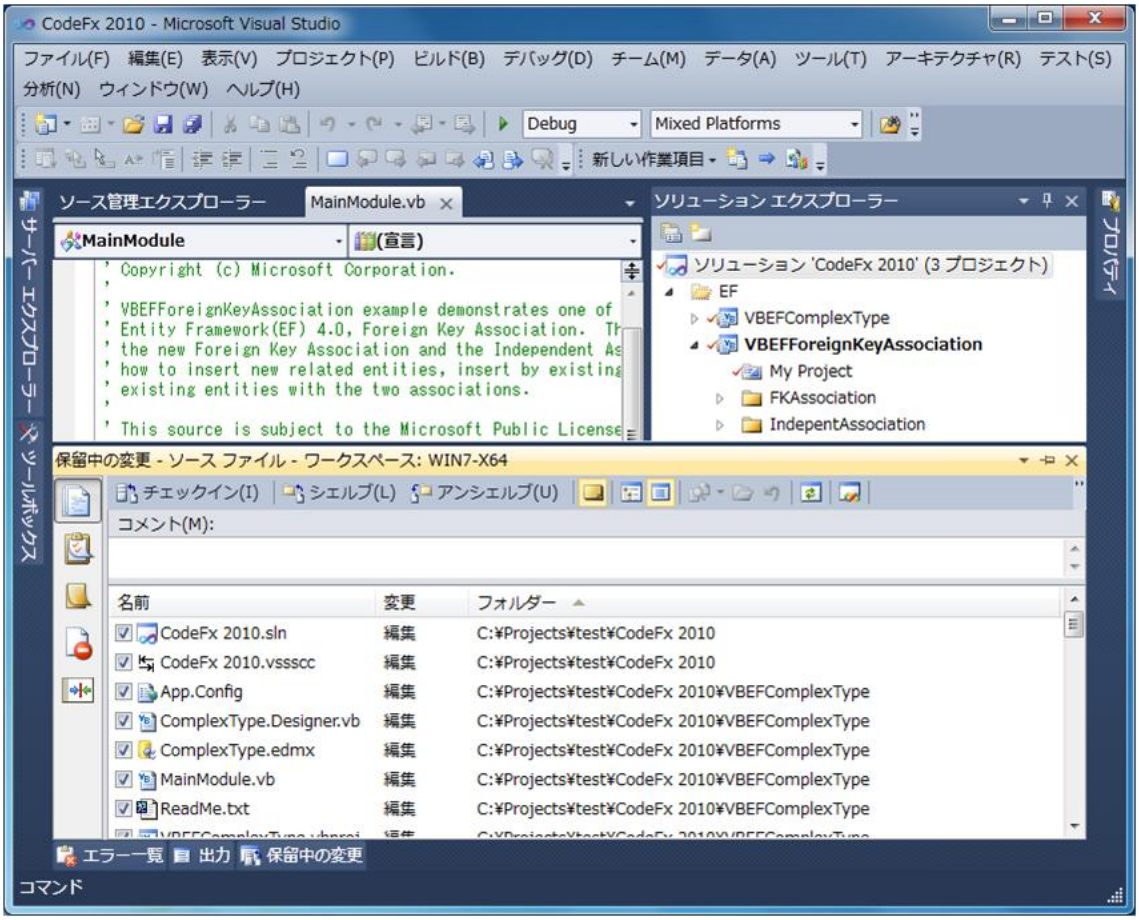


図 22: [保留中の変更] ウィンドウ

また、このウィンドウではもう 1 つのビューを利用することもできます。[保留中の変更] ウィンドウのツール バーで [フォルダー ビューに変更] というボタンをクリックします。すると今度は、図 23 のようにフォルダー構造でチェックアウト中のファイルの一覧を確認することができます。

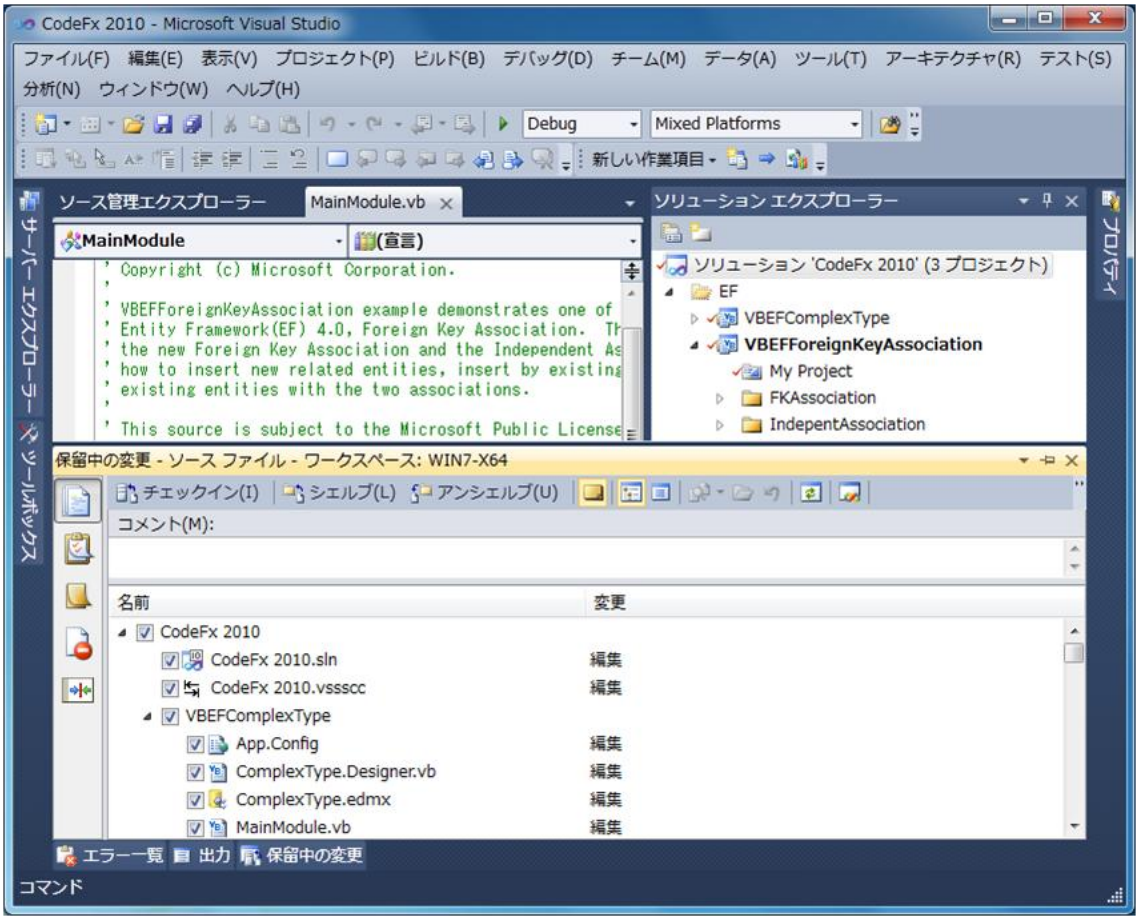


図 23: [保留中の変更] ウィンドウのフォルダー ビュー

[保留中の変更] ウィンドウのツール バーにある [チェックイン] を利用することでもファイルをチェックインできますが、フォルダー ビューを利用していると簡単にフォルダー単位でのチェックインを行うことができます。

○チェックインが衝突した場合の対処

チェックアウトの際にロックの種類を「変更なし」に設定している場合には、チェックインをするときにソース管理サーバーにあるファイル バージョンとローカルのファイル バージョンの変更内容の差分が確認されます。このとき、他のユーザーによってファイルが変更されていると、図 24 のような警告ダイアログ ボックスが表示されます。

図 24: 衝突検知の警告ダイアログ ボックス

この場合には、図 24 のダイアログ ボックスで [OK] をクリックして一度チェックインをとりやめ、衝突を回避する操作を行う必要があります。[OK] をクリックすると 1 つ前で紹介した [保留中の変更] ウィンドウが表示され、衝突を検知したファイルに対して図 25 のような詳細な説明が表示されます。

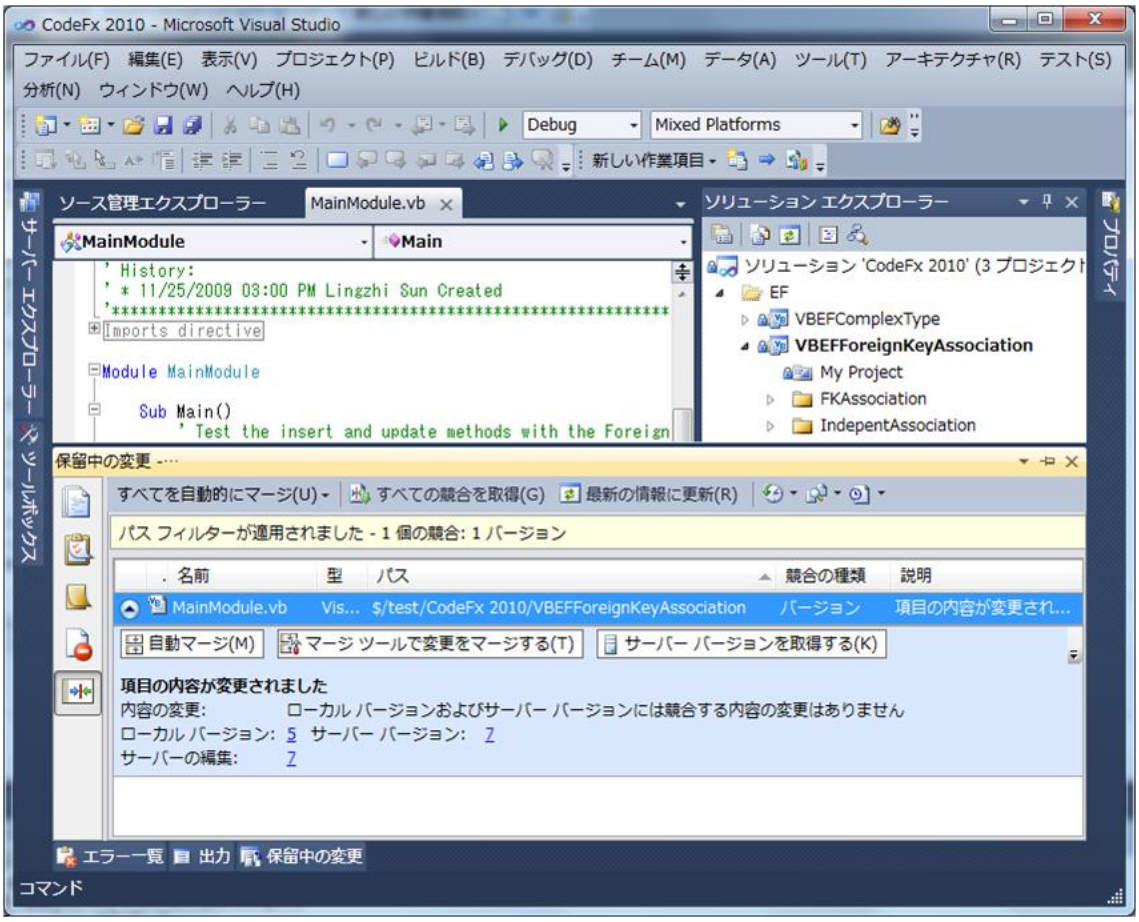


図 25: 衝突検知時の [保留中の変更] ウィンドウ

この内容をよく見てみるとローカル バージョンが 5 なのに対して、サーバー バージョンが 7 とサーバーにあるファイルの方が新しいことがわかります。このとき、利用者がとりえる選択肢は 3 つあります。

1 つ目は [サーバー バージョンを取得する] ことで、これは自分の編集内容を破棄することを意味しています。この選択肢を選ぶことは基本的にはないはずです。

2 つ目は [自動マージ] で、ローカル バージョンとサーバー バージョンの差を自動的に検知して、両方の変更分を 1 つのファイルに反映させます。TFS で衝突が検知された場合の最も基本的な選択肢です。

3 つ目は [マージ ツールで変更をマージする] ことで、手動で変更箇所をマージします。自動マージでも解決できなかった場合に選択します。

たいていの場合には自動マージで解決できてしまいますが、マージ ツールを使わなければいけなくなったときのことを考えて、その操作もこの機会に覚えてしまいましょう。図 25 で [マージ ツールで変更をマージする] をクリックすると、図 26 のようなマージ ツールの画面が表示されます。

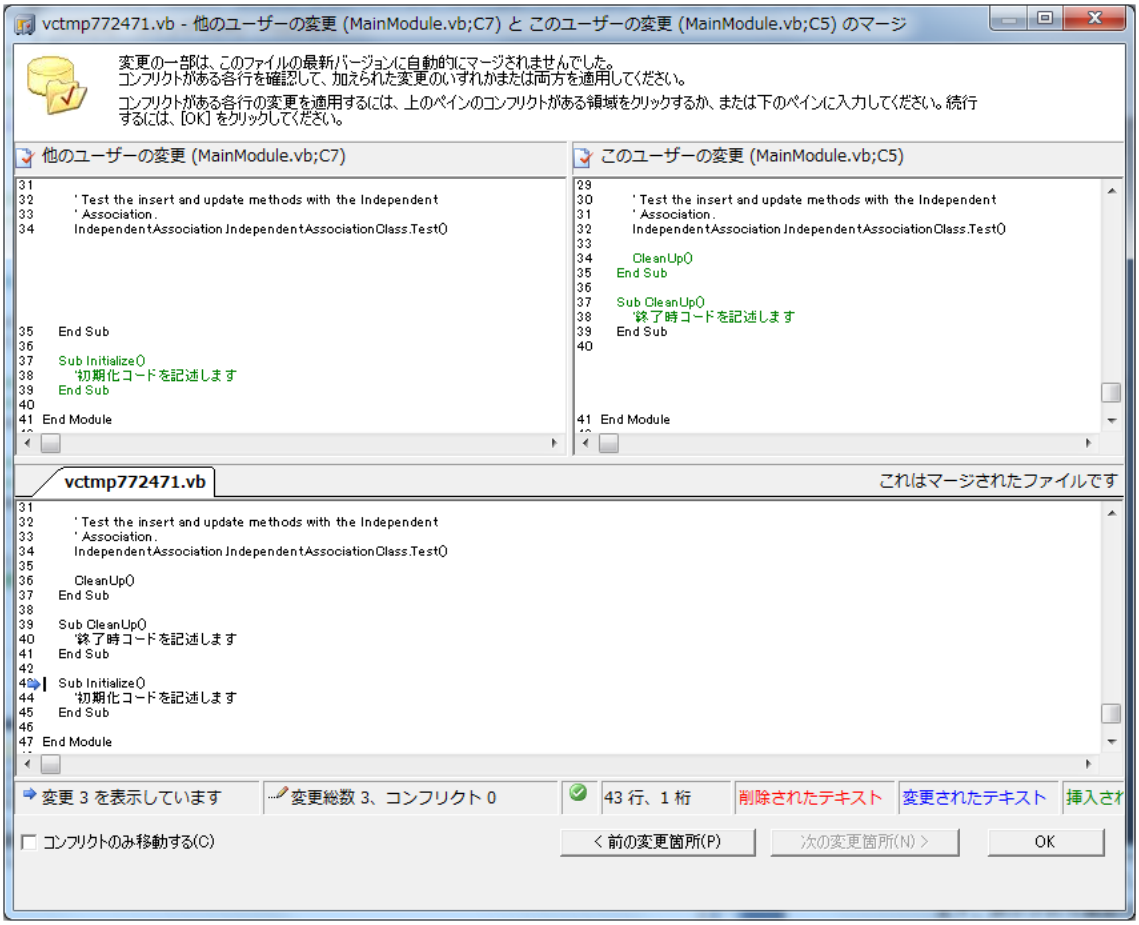


図 26: マージ ツール

図 26 では左上はサーバーにある最新バージョンのファイルの状態、右上は自端末にあるファイルの状態、下半分はマージ結果として生成されるファイルの状態を示しています。ちなみに、画面を開いた直後に下半分に表示されているマージ結果は自動マージによって行われる結果と同様です。自動マージの結果を事前に確認する場合にも、この画面を利用すると、結果の確認が容易になります。

さて、マージ ツールの操作ですが、衝突している場所でどちらのバージョンのファイルを採用するかを決めていく作業になります。そのため、まずは図 26 の左下にある [コンフリクトのみ移動する] チェック ボックスをオンにして、右下にある [< 前の変更箇所] と [次の変更箇所 >] ボタンで衝突している部分を移動しながら、それぞれの箇所でサーバー バージョンの内容かローカル バージョンの内容かを選択していきます。選択は簡単で、衝突箇所のどちらかの領域をクリックするだけです。するとその内容が下側のマージ結果のファイルに反映されていきます。すべての衝突を修正できたら、[OK] をクリックします。保存確認ダイアログ ボックスが表示されるので、[はい] を選択してファイルを保存しておきましょう。

最後に注意ですが、マージをしただけではチェックインは行われていません。あくまでも自分が編集しているファイルにサーバーに反映されていた変更を取り込んだにすぎません。そのため、必ずもう一度チェックイン操作を行うことを忘れないようにしてください。

これで、基本的なチェックアウトとチェックインの手順は終了です。既に VSS などのソース管理ツールを使っている場合にはほとんど変わらなかったはずです。まったくソース管理ツールを使ったことがない場合でもさして難しい内容はなかったのではないでしょうか。この機会に日々の作業結果を安全に管理する術を覚え、不慮の事態に備えられるようにしてみてください。

●特定バージョンの取得

ここで、少し違う話を挟みます。最新バージョンのファイルの取得は、チェックアウト時に自動的に行われるようになっていますが、ある特定時点のファイルを取得する方法についてはまだ述べていませんでした。ここでその方法とそのときに登場する変更セットというものについて説明しておくことにします。

まず、特定バージョンのファイルを取得するには、ソリューション エクスプローラーで対象のファイルを選択し、コンテキスト メニューから [特定バージョンの取得] を選択します。表示される画面は図 17 と同様ですが、今回は [バージョン] の [種類] ボックスでの選択肢に注目してください。ドロップダウン リストを展開すると、図 27 のように 5 つの選択肢が表示されます。

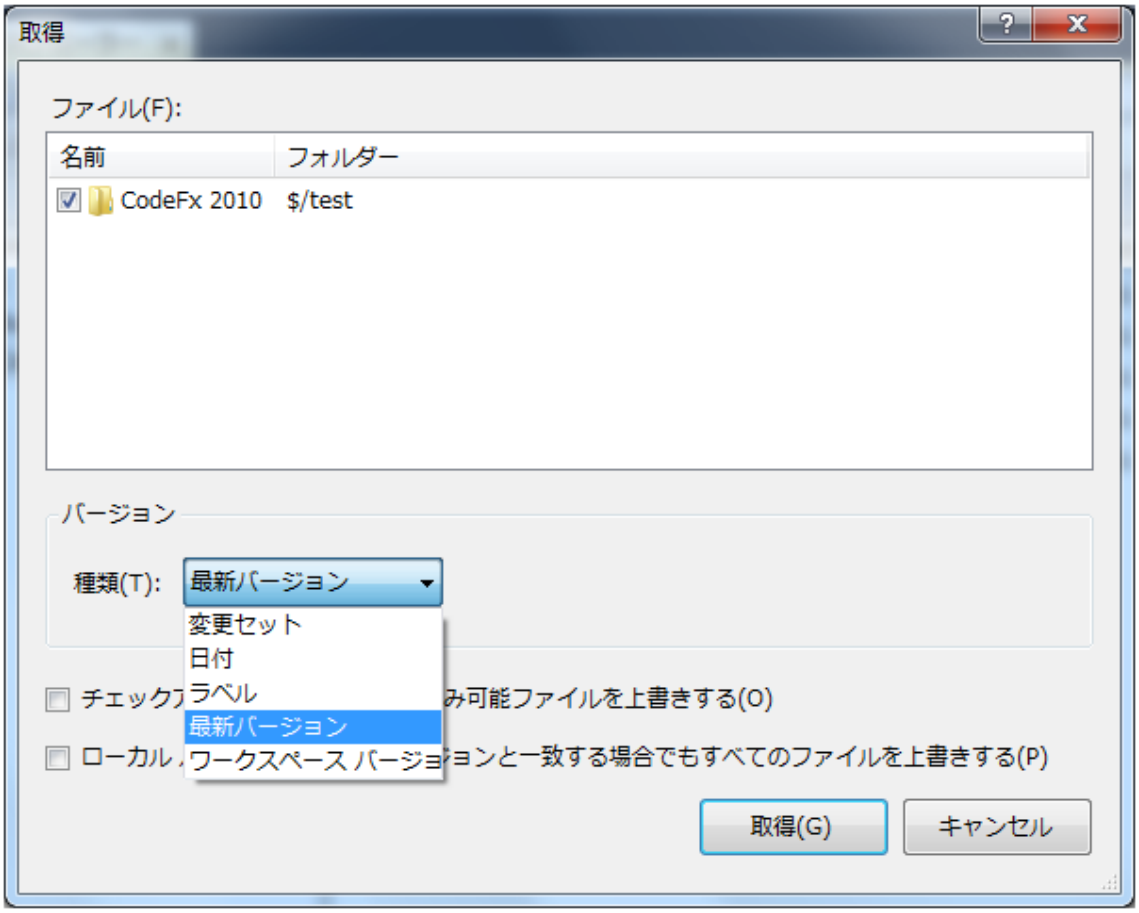


図 27: バージョンの種類

何かしらソース管理ツールを利用したことがあれば、日付、ラベル、最新バージョンについてはなんら違和感がないことと思いますが、これらを含めて、それぞれの意味を▲表 3 にまとめておきます。

種類	説明
変更セット	特定の変更セットに含まれているバージョンを取得する場合に利用します。
日付	ある特定の日付時点でのバージョンを取得する場合に利用します。

ラベル	ある特定バージョン時点で設定されたラベル名のバージョンを取得する場合に利用します。
最新バージョン	ソース管理にある最新バージョンを取得する場合に利用します。
ワークスペース バージョン	指定したワークスペースがダウンロードしている最新バージョンを取得する場合に利用します。

▲表 3: バージョンの種類

[最新バージョン] 以外を選択した場合には、図 27 の右側にそれぞれ対応するドロップダウン リストやテキスト ボックスなどが表示されます。日付やラベルはそのものずばりなのでよいとして、[ワークスペース バージョン] を選択した場合には、右側に表示されるドロップダウン リストで、現在のユーザーがアクセス可能なワークスペースから特定のワークスペースを選択する必要があります。TFS はどの端末がどのファイルをダウンロードしたかを記憶していると説明しましたが、正確にはどのワークスペースがどのファイルをダウンロードしたかを記録していて、その情報に基づいてファイルの取得が行われます。

もう 1 つの選択肢として [変更セット] がありますが、そもそも変更セットとは何かというところから理解しておきましょう。ファイルをチェックインする場合、複数のファイルをまとめてチェックインすることがあります。まとめてチェックインするということは、たいていの場合にはそれらのファイルは同時に編集する必要があった、つまりお互いに関連性がある可能性が高いということになります。しかし、VSS の場合には、ファイルがまとめてチェックインされたとしてもあくまでも 1 つ 1 つ独立したチェックインと解釈されていたため、後から関連性を見つけ出すことは非常に困難でした。一方、TFS ではチェックインはすべて変更セットというもので管理されるようになっていて、変更セットにはまとめてチェックインされたすべてのファイルが含まれます。つまり、ある変更セットを見れば、その中から複数のファイルを見ることができるため、それらが関連性を持って変更された可能性が高いということを簡単に推測できます。

というわけで、バージョンの種類から [変更セット] を選択した場合の操作に話を戻しましょう。この場合、右側には [...] というボタンが表示されます。これをクリックすると、図 28 のような [変更セットの検索] ウィンドウが開きます。

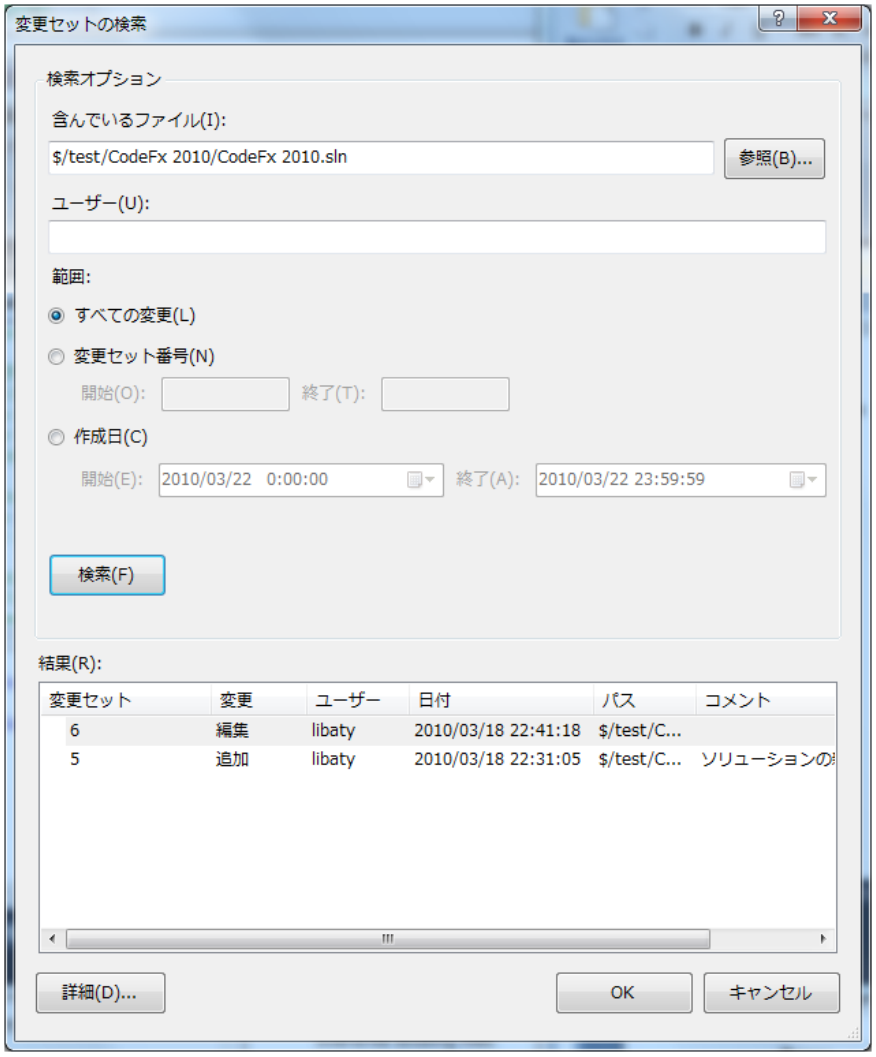


図 28: [変更セットの検索] ウィンドウ

変更セットの検索は、特定のファイルを含んでいるか、だれがチェックインしたものか、いつチェックインされたものかなどを検索条件として実施することができます。図 28 の下側は実際に検索を行った結果を表示しているもので、ここから任意のものを選択することで特定の変更セットを使って特定バージョンのファイルを取得できます。なお、変更セット検索結果のグリッド欄には各チェックイン時のチェックイン コメントも同時に表示されているため、コメントがきちんと入力されていれば、目的の変更セットもより検索しやすくなるでしょう。

■ソース管理を始めるには (TFS 管理者のために)

TFS のソース管理機能を利用した作業の仕方は理解できましたか? 大多数の利用者はここまでのことが理解できていれば日々の作業には困らないはずです。しかし、個々の開発者に対してその環境を用意する立場にある場合は話が別です。たとえば VSS を利用する場合でも共有フォルダーを公開する設定を行い、VSS データベースを用意し、ユーザーの登録とアクセス権の設定という作業が必要でした。

これと同じように TFS のソース管理機能を利用する場合にもいくつか準備しなければいけないことがあります。そこでこの章では、開発者に対してソース管理機能が利用可能な状態で公開するまでの手順を、順を追って説明します。

●TFS ベーシック構成のインストール

まずは、TFS のインストールから始めます。そもそも TFS は開発にかかわるさまざまな情報を管理するためのサーバー製品であるため、従来のバージョンでは Windows Server にインストールすることが前提となっていました。しかし、VSS と同じくらい手軽に使えるようにしようというコンセプトのもと、TFS 2010 は Windows Vista と Windows 7 に限って、クライアント OS にもインストールできるようになっています。クライアント OS を利用するため、同時接続ユーザー数には制限がかかることになりますが、少人数で利用する場合には十分です。今回は、TFS 2010 を Windows 7 にインストールして、セットアップしていく手順を確認していくことにします。

なお TFS 2010 は、IIS (Internet Information Services) を必要とするため、必然的にインストール可能な OS のエディションが限定されることになるので注意してください。具体的には、Windows Vista の場合には、Home Premium、Business、Enterprise、Ultimate Edition のいずれかでかつ Service Pack 2 適用済みのもの、Windows 7 の場合には、Home Premium、Professional、Enterprise、Ultimate Edition のいずれかとなります。

では、さっそくインストールを開始しましょう。インストール用メディアをセットし、Windows エクスプローラーでメディアの内容を表示します。すると、「TFS-x64」または「TFS-x86」というフォルダーがあるので、現在の OS に合わせて適切な方を選択したうえで「setup.exe」を起動します。

[コラム] TFS 2010 は 64 ビット環境にもインストール可能

TFS 2005 または TFS 2008 では、TFS のすべての環境を 64 ビット環境で構築することはできませんでしたが、TFS 2010 ではすべての環境を 64 ビット環境だけで構築することが可能となっています。

後はウィザードに従っていくだけでインストールは完了ですが、図 29 に示すインストールする機能の選択の部分では、特に問題なければすべての機能を選択しておくようにしましょう。ビルド サービスは何かよくわからないから利用しないという意見もありそうですが、後から設定するのでぜひ入れておいてください。

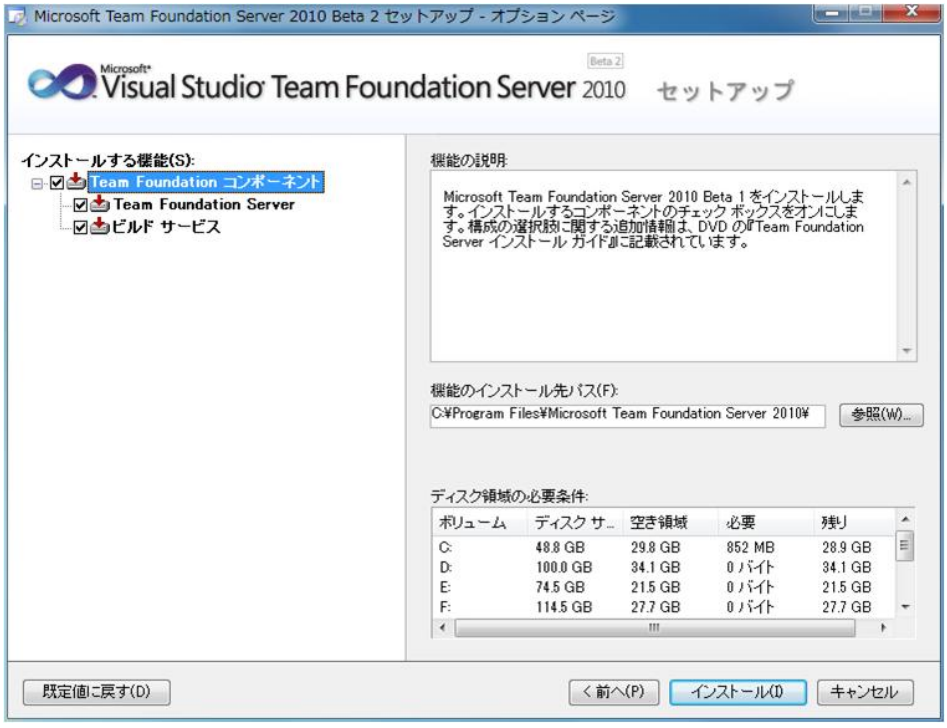


図 29: TFS にインストールする機能の選択

インストールが完了するとセットアップの完了ページが表示されます。ここで、左下の [Team Foundation Server 構成ツールを起動する] チェック ボックスがオンになっていますが、図 30 のようにこれを変更せずにそのまま [完了] をクリックしてください。続けて、TFS のセットアップ作業を行っていきます。

図 30: TFS のインストール終了後の画面

こちらも従来であれば、インストールと同時に TFS のセットアップも実施されていたため、そもそもインストール前の段階でかなり多くの前提条件を満たしておく必要がありました。なおかつ、セットアップ中に実施してもらえない内容もあり、手作業でやっておくことも多かったのが実際です。しかし、TFS 2010 のインストールは、システム要件を満たしてしまえば一切の準備は必要ありません。これは、セットアップに関しても同様です。今回は、細かいことは気にせずにすべて TFS のセットアップ プロセスにお任せして設定していくことにするので、このまま作業を続行します。

さて、本題に戻りましょう。図 31 のように Team Foundation Server の構成画面が表示されます。ここからウィザードを使って、TFS を利用可能にするための各種設定を行っていきます。

図 31: TFS の構成画面

図 31 では、左側のメニューから [基本] を選択して、右下の [ウィザードの開始] をクリックしてください。ウィザードの開始を知らせる画面に切り替わるので、内容を確認してから [次へ] をクリックします。次の画面では、SQL Server の構成をどのようにするかを選択します。ここでは図 32 のように [SQL Server Express] を選択します。

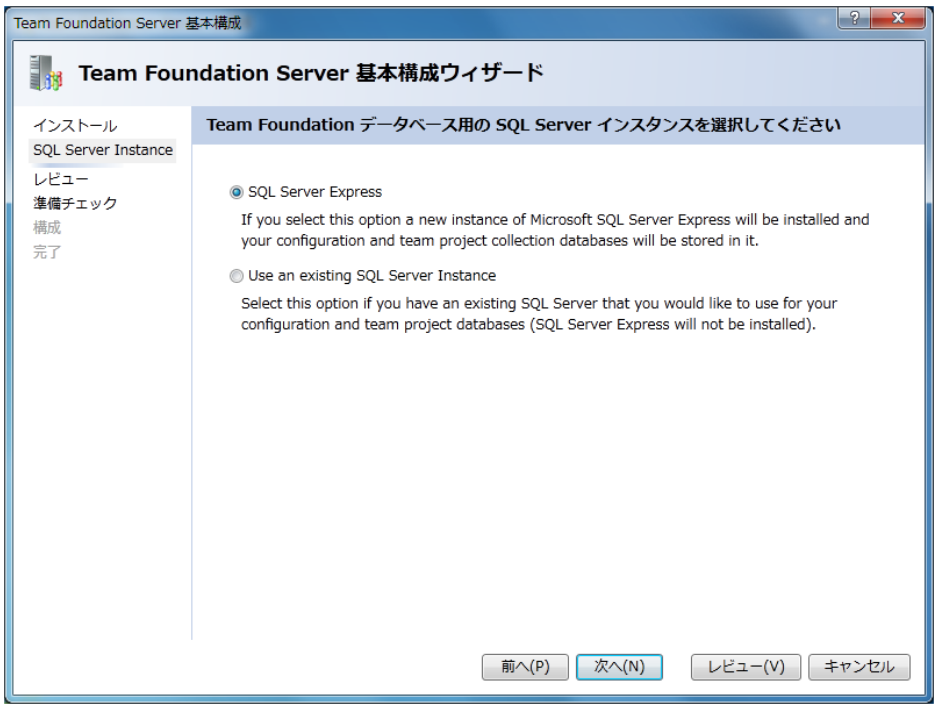


図 32: SQL Server の構成

既に SQL Server がインストールされている環境で図 32 の画面にきた場合は [Use an existing SQL Server Instance] だけが選択可能です。この場合、さらに次の画面では、SQL Server への接続文字列を入力する必要があります。

図 32 で選択が完了したら、[次へ] をクリックします。次の画面では、インストールする内容が表示されます。ここは確認するだけですが、ざっと見ておくとよいでしょう。確認できたら [次へ] をクリックします。すると、セットアップのための検証が行われ、たとえば図 33 のような画面で検証結果が報告されます。

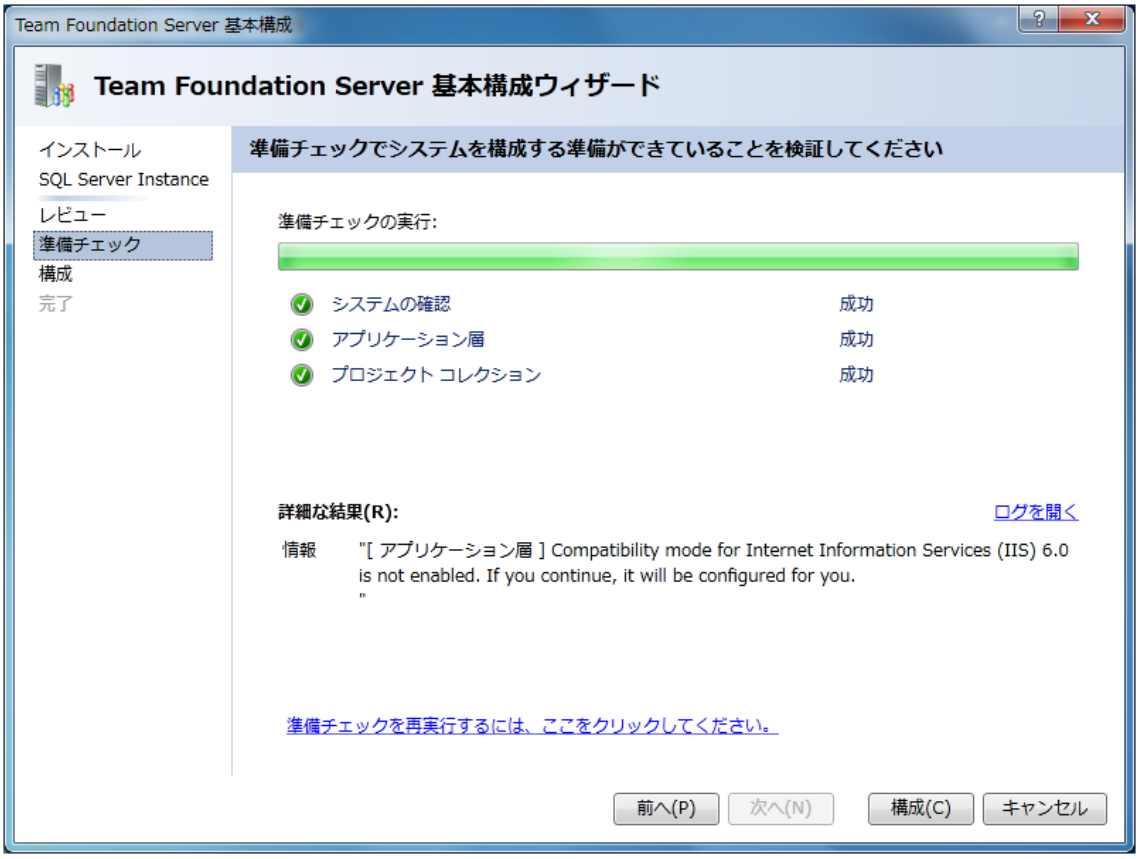


図 33: TFS の構成前チェック結果

ここでは、IIS に IIS 6.0 互換機能が設定されていないという内容が表示されていますが、それもそのはずで、ここでは IIS のインストールを行わずにウィザードを実行しています。しかし、IIS のセットアップもこのウィザード内で行われるため、このまま [構成] をクリックしてしまってかまいません。当然ですが、その他にエラーが表示されている場合には、その内容を修正しておく必要があります。

構成が始まると、IIS の設定や SQL Server のインストールなどが行われるため少し時間がかかりますが、無事に終了すれば、図 34 のように結果の画面が表示されます。このとき、ファイアウォールの例外ポートがいくつか設定されたことも併せて報告されるので、セキュリティ上、押さえておく必要がある場合には注意してください。

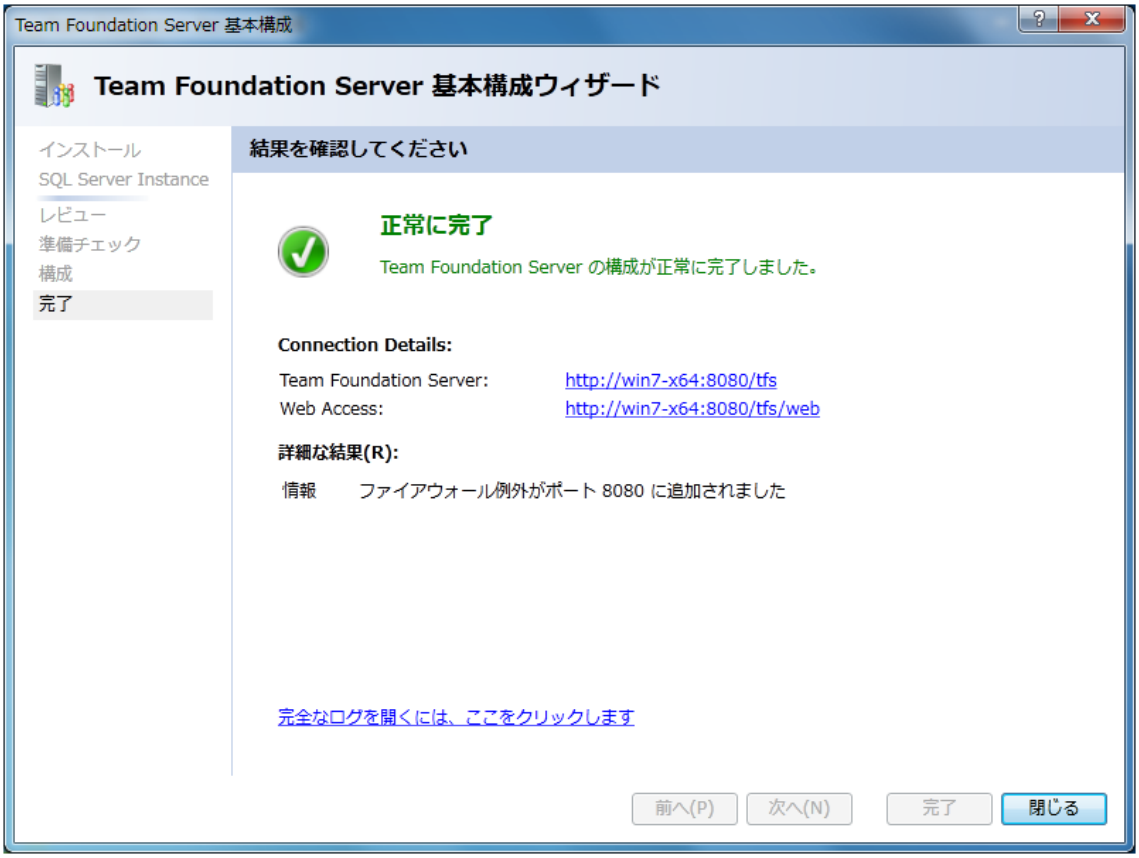


図 34: 構成の結果画面

最後に図 34 で [閉じる] をクリックすると、今度は別の画面が起動します。これは TFS 2010 で新しく導入されたもので、TFS 管理コンソールというものです。管理コンソールを利用することで、TFS の現在の状態を確認したり、各種の設定を変更したりすることができます。本稿では割愛するので、詳細については MSDN ライブラリの Team Foundation の管理コンソール [http://msdn.microsoft.com/ja-jp/library/dd236910(VS.100).aspx] を参照してください。

●チーム プロジェクトの作成

TFS のインストールが完了したら、次はチーム プロジェクトを作成します。ソース管理の日々の使い方の章でも何度か登場していましたが、チーム プロジェクトとは、開発によって生み出される成果物やそこに至る前の作業過程などを TFS で管理する単位のことで、TFS のどんな機能を使う場合でも必ず必要になるものです (図 35)。

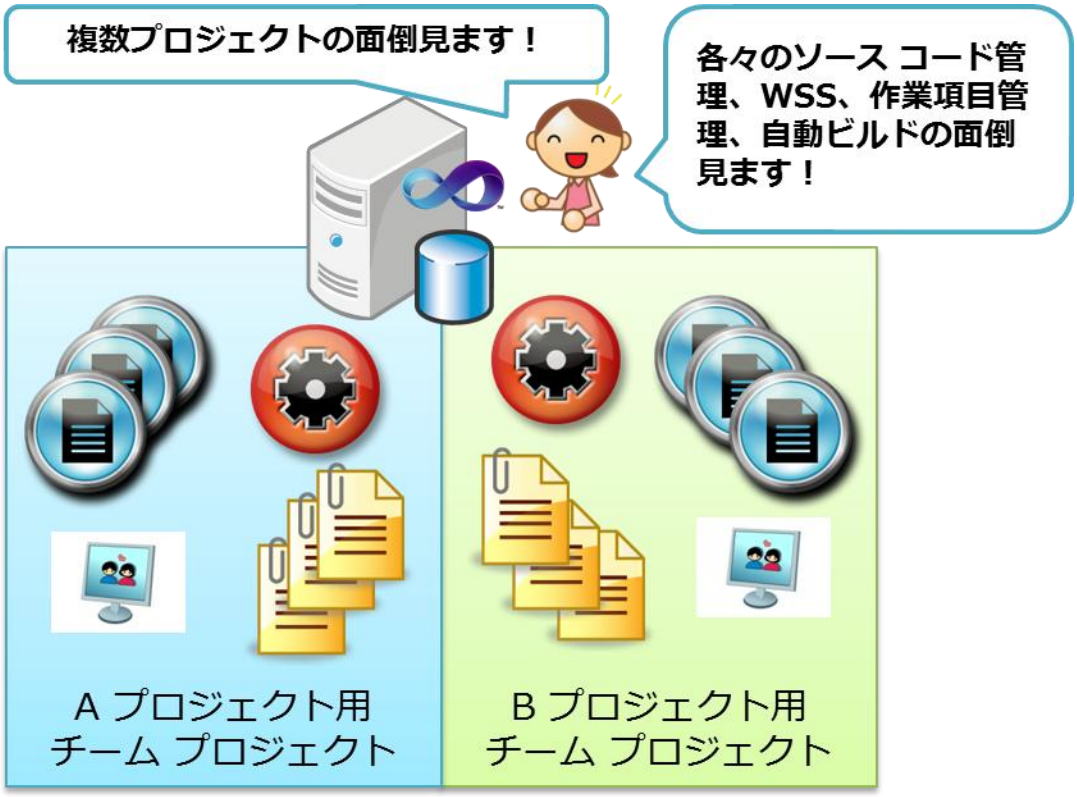


図 35: チーム プロジェクトのイメージ

TFS 2010 では、ここの管理単位であるチーム プロジェクトを複数格納するチーム プロジェクト コレクションという考え方も導入されています。図 35 の場合では、「A プロジェクト用チーム プロジェクト」と「B プロジェクト用チーム プロジェクト」を持っているものと考えておいてください。チーム プロジェクト コレクションは、後述するビルド サービスを設定する単位であったり、TFS が作成する SQL Server データベースの単位であったりします。TFS をインストールした直後では DefaultCollection という名前のチーム プロジェクト コレクションが作成されているので、初めてチーム プロジェクトを作成する場合には、DefaultCollection に格納されるように作成することになります。

[コラム] 従来の TFS と TFS 2010 のデータベースの持ち方の違い

従来の TFS では、すべてのチーム プロジェクトが論理的には 1 つのデータベースに格納されていました。正確には TFS 自体が複数のデータベースに分けてデータを管理するようになっていたので、1 つのデータベースに格納されているわけではありませんでしたが、論理的に見ると TFS とデータベースは 1 対 1 で対応していました。

TFS 2010 では、チーム プロジェクト コレクションの単位で論理的にも物理的にも別々のデータベースが作成されるようになるため、バックアップ リストアの単位を TFS 全体からチーム プロジェクト コレクション単位に変更することができます。これにより、異なるチーム プロジェクト コレクションであれば、バックアップ計画を個別に検討することや、障害発生時のリカバリ ポイントを少なくすることができるというメリットが生まれています。

チーム プロジェクトの作成を開始する前にもう 1 つ確認しておきたいことがあります。それはチーム プロジェクトの作成単位です。チーム プロジェクトを作成する単位についてよく質問を受けるのであえて取り上げることにしますが、答えは簡単です。これからチーム プロジェクトを作成しようとしているあなたはどんな単位で管理を行おうと

考えているか、が答えです。いくつか例を挙げますが、たとえば、少人数で複数の開発を同時並行的に行っている場合、それぞれの開発ごとにチーム プロジェクトを分けるよりはすべてひとまとまりにした方が、管理するには都合が良いかもしれません。逆に大規模な開発では、工程などの単位で分けておいた方が、都合が良いかもしれません。また、プロジェクトの管理として考えると難しい場合には、ソース管理したい単位で検討を行うとわかりやすいかもしれません。このような方針を参考に、実際に作成する単位を決定してください。それでもなお悩まれる場合には、1 つのアプリケーションの単位で作成するのが最もシンプルな選択肢なので、まずはそこから始めてみてください。

さて、実際に作成していきましょう。まず、VS を起動してください。次に前述のチーム プロジェクトへの接続の部分を参考に、TFS への接続を行います。ただし、ここではまだチーム プロジェクトがないので、図 6 に相当する部分は図 36 のように表示されます。

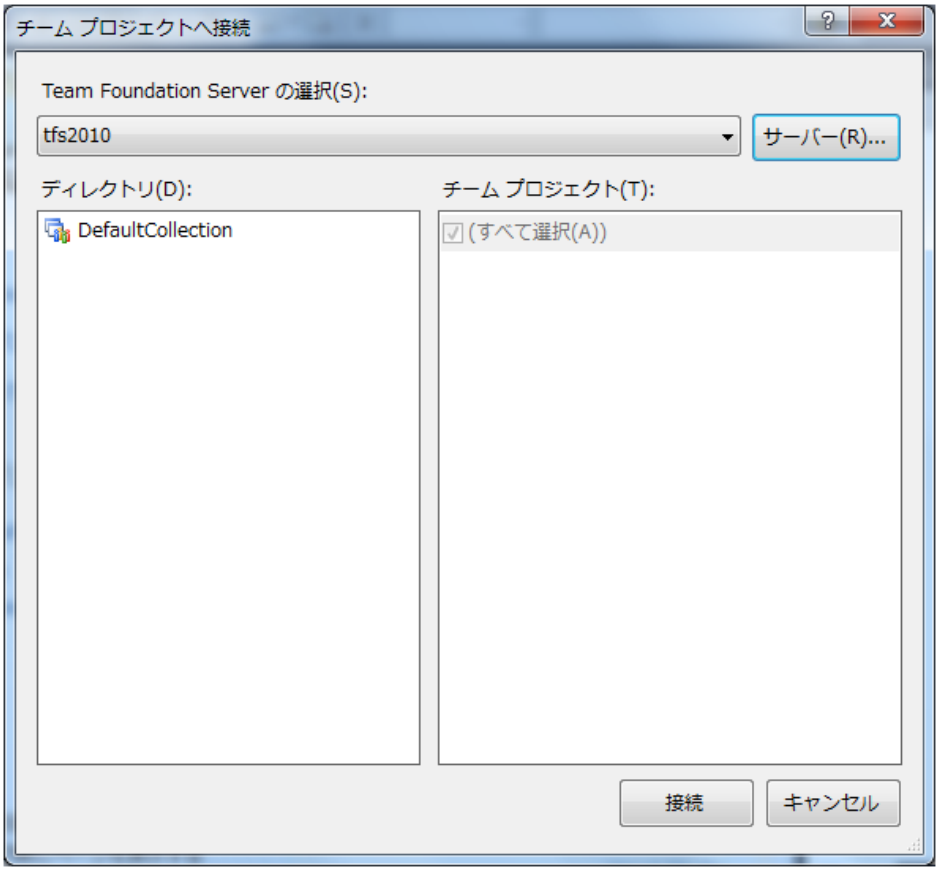


図 36: TFS への接続 (チーム プロジェクトがない場合)

接続が完了し、チーム エクスプローラーが表示されたら、TFS 名¥DefaultCollection となっている部分を選択して、コンテキスト メニューから [新しいチーム プロジェクト] を選択します。すると新しいチーム プロジェクトを作成するウィザード画面が表示されます。最初の画面では、[チーム プロジェクトの名前を指定してください] ボックスにチーム プロジェクトの名前を入力して、[次へ] をクリックします。図 37 では例として「test」と入力してみました。

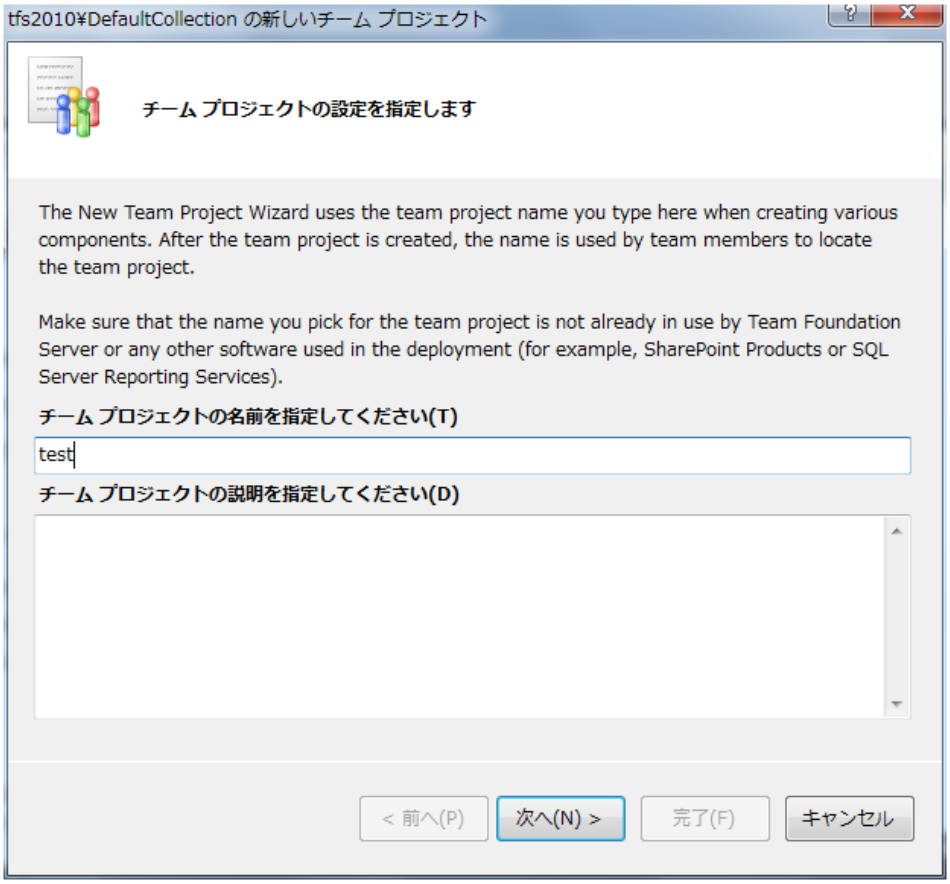


図 37: チーム プロジェクトの名前の入力

次の画面では、プロセス テンプレートを選択します。プロセス テンプレートとは、TFS にあらかじめ用意されている機能の使い方のひな型となっているもので、標準ではアジャイル開発向きの MSF Agile とウォーターフォール開発向きの MSF CMMI というものがあります。しかし、ソース管理 + α 程度の機能を使う場合にはどちらを選んだとしても大差はありません。ここでは、標準のままで [次へ] をクリックします。

次の画面ではチーム サイトの設定を行います。チーム サイトとは、チーム プロジェクトのためのポータル サイトのことで、WSS (Windows SharePoint Services) を使って、ファイル共有や連絡事項の共有、Wiki の利用などを行うことができるサイトです。これは作っておくと後で便利なので、ここも標準のままで [次へ] をクリックしましょう。

次の画面ではソース管理に関する設定を行います。チーム プロジェクトを作成すると、そのチーム プロジェクトのためのソース管理のエリアが必ず 1 つは作成されることになります。

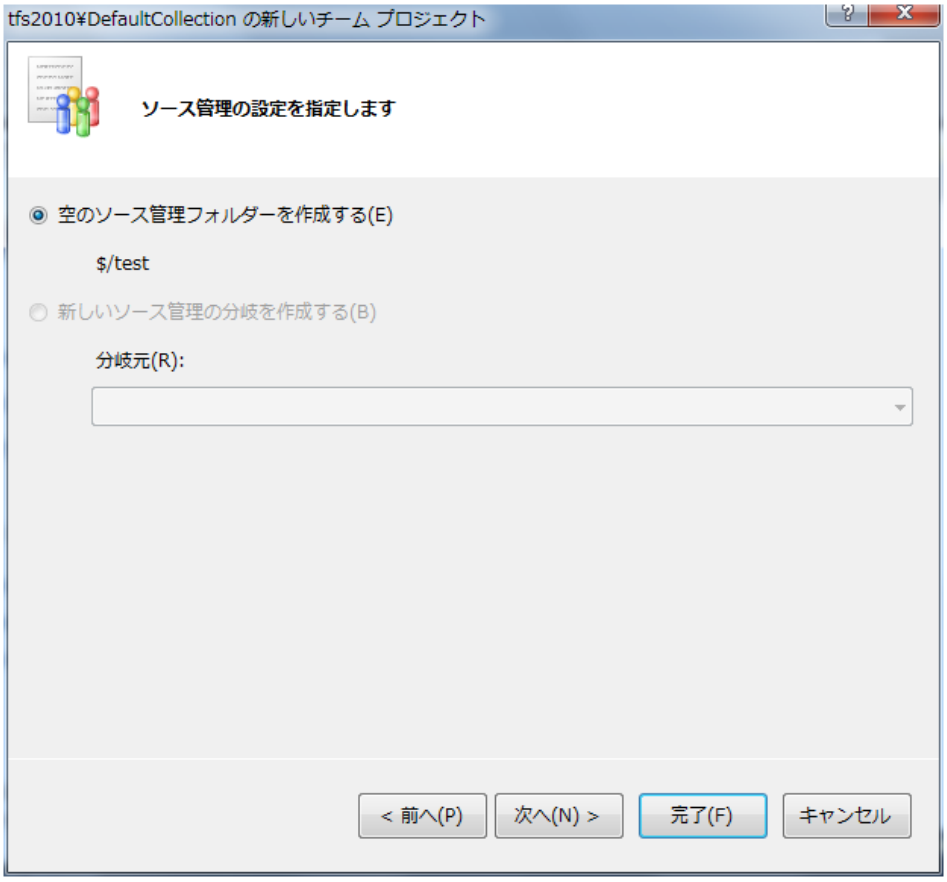


図 38: ソース管理の設定

図 38 では [空のソース管理フォルダーを作成する] が選択されています。標準ではこれでかまいませんが、たとえば、既に別のチーム プロジェクトが作成してあって、そこからソース管理を分岐させて新しいソース管理エリアを作成したい場合には、図 38 の下側の選択肢を選んで分岐元となるソース管理エリアを設定しましょう。
[次へ] をクリックして先に進み、最後の画面で設定してきた内容の確認を行い、[完了] をクリックします。作成には少し時間がかかりますが、正常に終了すればその結果を示す画面が表示されるので、その画面はそのまま [閉じる] をクリックして終了します。最終的に図 39 のようにチーム エクスプローラーには作成したチーム プロジェクト (ここでは test) が表示され、VS の中央部分には選択したプロセス テンプレートに応じたガイダンス ページが表示されます。

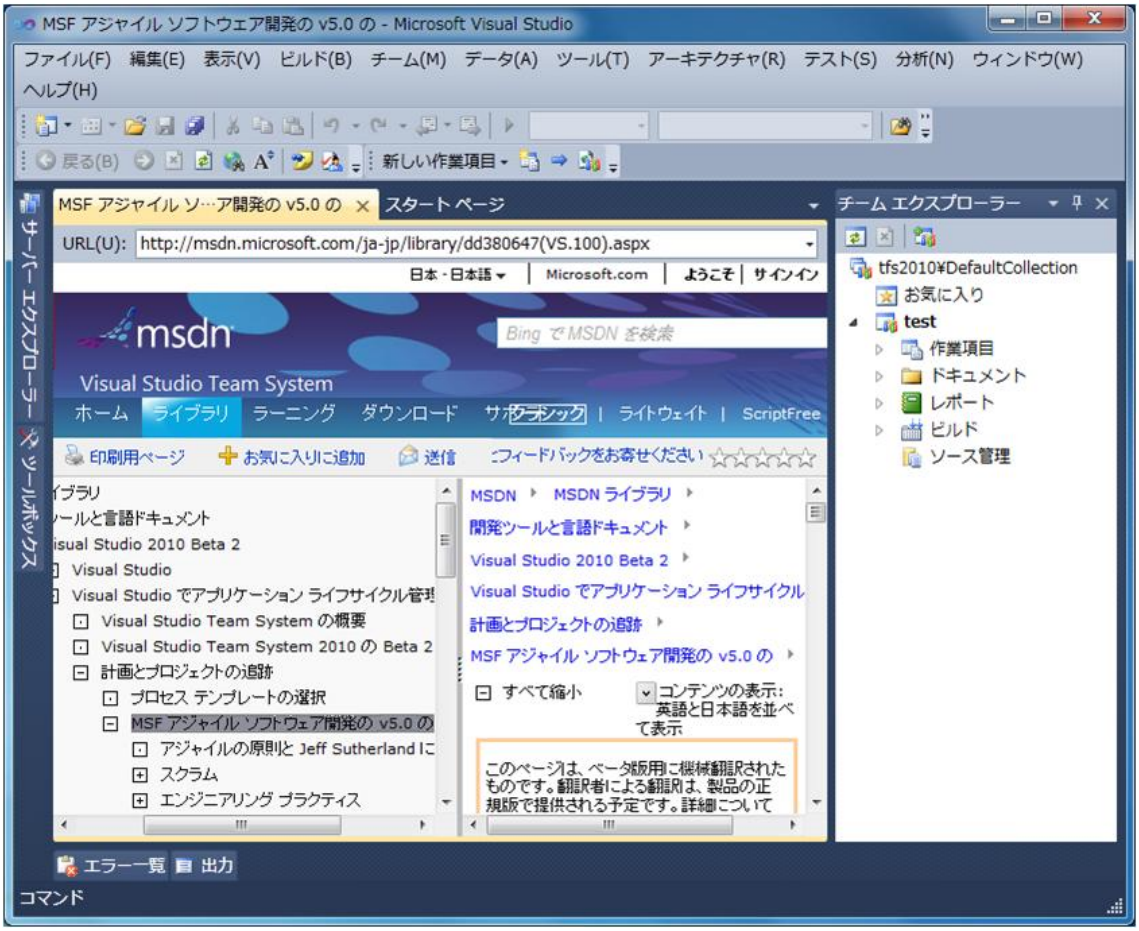


図 39: チーム プロジェクトの作成の終了

ガイダンス ページはプロセス テンプレートに従って TFS を使いこなそうという場合以外にはすぐに必要なものではないので、時間のあるときに見る程度でかまいません。

●ソース管理の設定の確認

チーム プロジェクトが作成できたら、次はソース管理のための設定を行っておきましょう。大した手間ではありませんが、重要なポイントも含まれているため、特に VSS を利用していた場合には、違いを認識して設定していく必要があります。

まず、チーム エクスプローラーでチーム プロジェクトを選択し、コンテキスト メニューから [チーム プロジェクトの設定] – [ソース管理] を選択します。ソース管理の設定画面が開くので、ここで [チェックアウトの設定] タブの中身を設定しておきましょう。

1 つ目は [複数のチェックアウトを有効にする] ですが、初期値では有効に設定されています (図 40)。

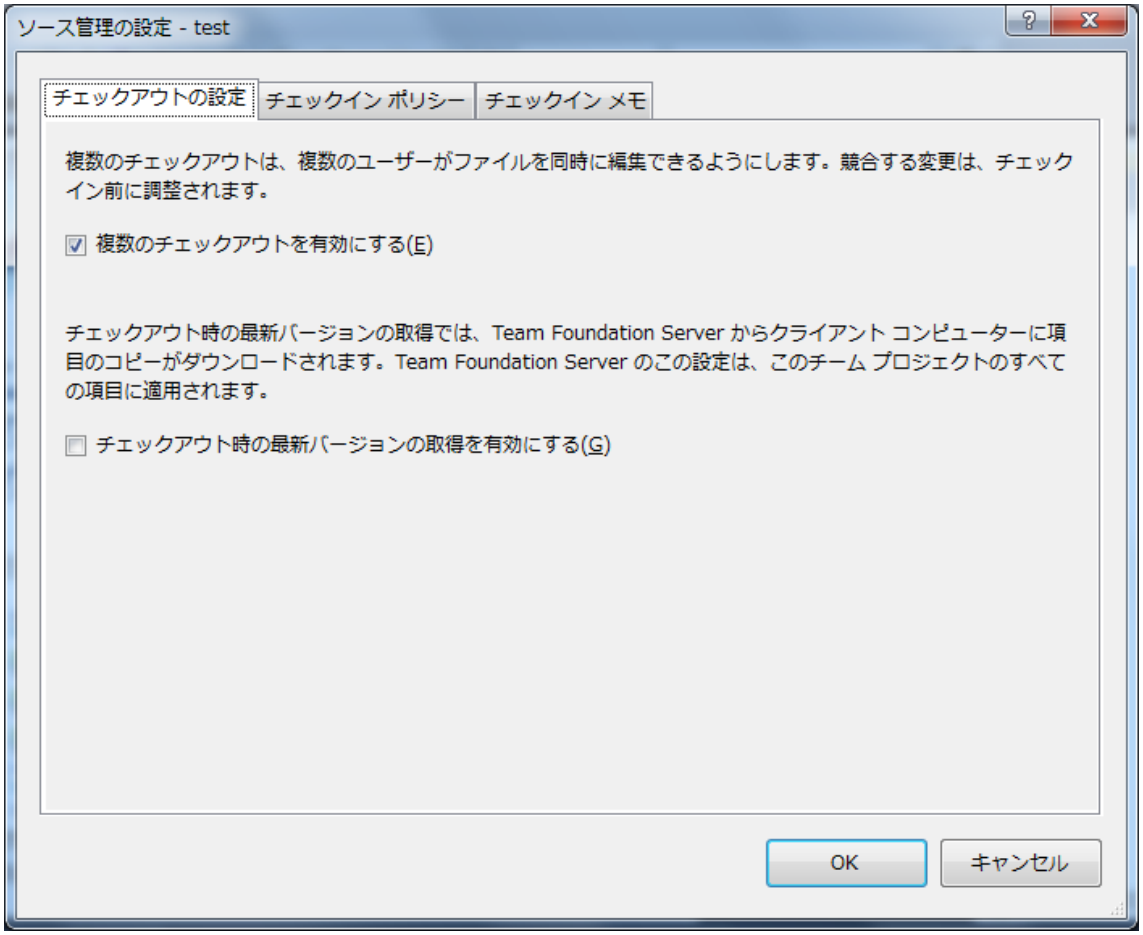


図 40: ソース管理のチェックアウトの設定

VSS を利用していた場合、ファイルをチェックアウトして編集する際にサーバー上のファイルはロックがかけられ、他の人は触ることができませんでした (ここではこのようなやり方を排他ロック モデルと呼ぶことにします)。一方、最近のソース管理ツールでは、チェックアウト時にはロックをかけず、チェックインする際に他のユーザーに編集されていないかを確認し、編集されていなければそのままチェックインし、編集されていた場合には変更点をマージしてチェックインするという方法が主流になりつつあります (ここではこのようなやり方をマージ モデルと呼ぶことにします)。TFS では標準はマージ モデルですが、[複数のチェックアウトを有効にする] チェック ボックスをオフにすることで、VSS と同じ排他ロック モデルにすることができます。どちらにもメリット、デメリットがあるので、これからの作業に都合の良い方を選択してください。

2 つ目は [チェックアウト時の最新バージョンの取得を有効にする] の設定です。TFS では、チェックアウト時にわざわざ最新バージョンを取得して編集するよりもチェックアウトのマークだけ設定して編集を開始してしまい、後でマージすればよいという考え方が前提にあります。最新バージョンを取得しようとして失敗してしまうとそこで作業が止まってしまうため、できるだけ作業を継続できるようにしようというわけです。しかし、これは VSS 利用者にはかなり違和感のある設定です。1 つ目の設定と併せて有効にすれば、完全に VSS と同じ動きをするようになるので、こちらも作業に都合の良い方を選択してください。

なお、他のタブの内容については現時点ではあまり気にしなくて問題ありません。設定が完了したら、[OK] をクリックして変更を保存して終了します。

●セキュリティの設定

セキュリティというとずいぶんと幅広い世界になってしまって、身構えてしまいがちですが、ここではアクセス権の設定を行います。アクセス権の設定であれば普段 Windows エクスプローラーなどで行っているのではないでしょうか。その経験があれば難しいことはほとんどありません。2 か所に設定する必要があるので少々面倒ではありますが、覚えてしまうだけなのであまり身構えずにトライしてみましょう。

○チーム プロジェクトのアクセス権設定

1 つ目はチーム プロジェクトのアクセス権です。チーム プロジェクトとはさまざまなものを管理する単位であると同時に、それらのものを入れる器でもあります。ファイルを入れるフォルダーのようなものと考えると、そのフォルダーにアクセス権の設定が必要なようにチーム プロジェクトにもアクセス権の設定が必要となることがわかるのではないでしょうか。

アクセス権を設定するには、チーム エクスプローラーでチーム プロジェクトを選択し、コンテキスト メニューから [チーム プロジェクトの設定] – [セキュリティ] を選択します。すると図 41 のような画面が表示され、ここから権限の設定をしていくことができます。

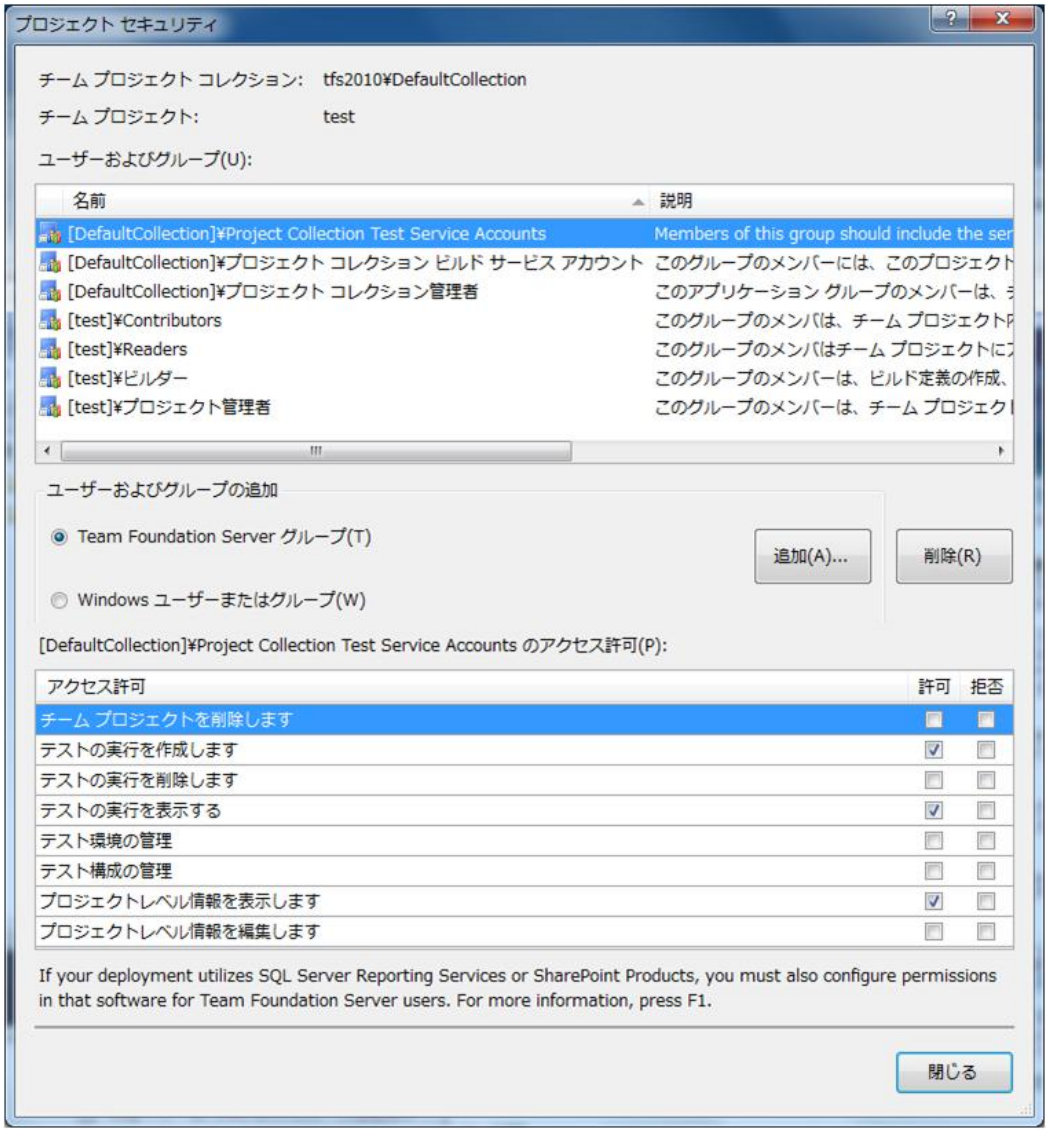


図 41: チーム プロジェクトのセキュリティ設定

いろいろと出てきているので少し確認しておきます。まずは [ユーザーおよびグループ] ボックスです。図 41 はチーム プロジェクト作成後に標準で設定されている項目ですが、それぞれ▲表 4 のような意味を持っています。

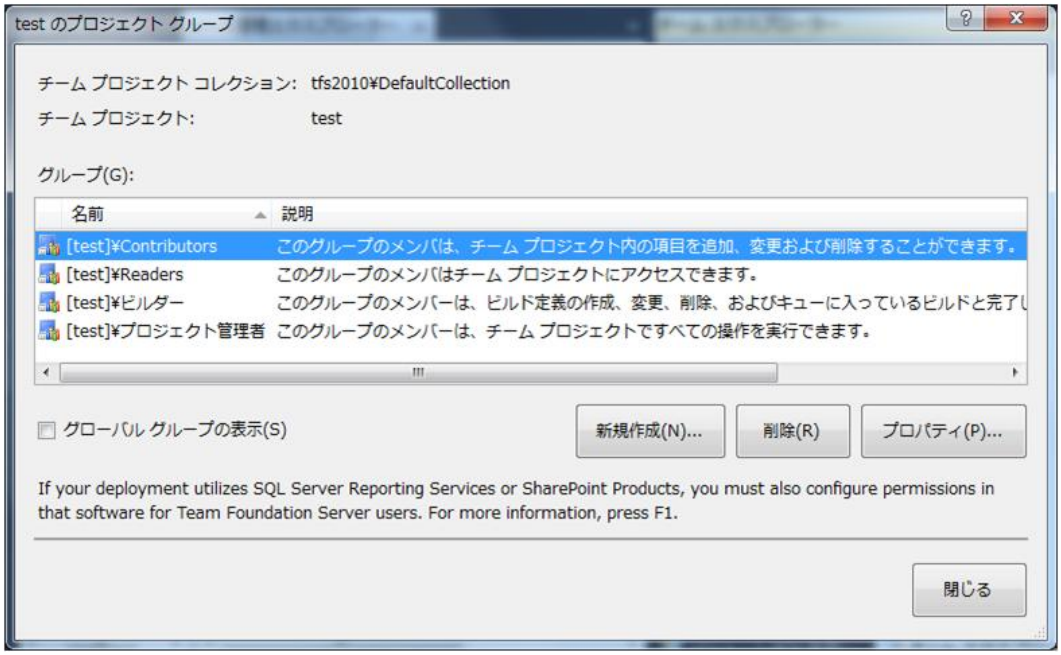
グループ名	解説
[DefaultCollection]¥Project Collection Test Service Accounts	VS 2010 で導入されたテスト管理機能を利用する場合にそのサービス アカウントが登録されているグループです。
[DefaultCollection]¥プロジェクト コレクション ビルド サービス アカウント	自動ビルドを実行する際に利用されるサービス アカウントが登録されているグループです。
[DefaultCollection]¥プロジェクト コレクション管理者	複数のチーム プロジェクトにまたがって特権操作を許可されるグループです。
[test]¥Contributors	チーム プロジェクト内の項目の追加、変更、削除を許可されるグループです。
[test]¥Readers	チーム プロジェクトにある情報の参照を許可されるグループです。
[test]¥ビルダー	自動ビルドのビルド定義の作成、変更、削除、および実行を許可されるグループです。
[test]¥プロジェクト管理者	チーム プロジェクトですべての操作を許可されるグループです。

▲表 4: チーム プロジェクトに登録されているグループ

接頭辞に [DefaultCollection] が付いているものは複数のチーム プロジェクトにまたがって利用されているグループで、このグループに対しての権限の設定を行う必要はありません。接頭辞に [test] が付いているものはこのチーム プロジェクトのために用意されているグループで、このグループは各チーム プロジェクトで任意に権限を設定していくものです。

それぞれのグループに標準で割り振られている権限は、図 41 の下側に表示されます。ここから、既存のグループに対して権限を変更することも、既に作成済みの他のグループを追加して権限を設定することも可能です。しかし、これらの設定を厳密にやっていこうとするとなかなかたいへんな作業になるので、ここでは内容を確認するだけにして、設定画面を閉じます。

続いて、もう一度チーム エクスプローラーでチーム プロジェクトを選択し、コンテキスト メニューから [チーム プロジェクトの設定] – [グループ メンバーシップ] を選択します。今度は、▲図 42 のような画面が表示されます。



▲図 42: グループ メンバーシップの設定

先ほど図 41 で確認した [test] から始まるグループが表示されています。この画面ではこのチーム プロジェクト専用の新しいグループを作成したり、グループに対してユーザーを追加したりといったことを行います。たとえば、チーム プロジェクトの管理者に設定したいユーザーはプロジェクト管理者グループに登録し、チーム プロジェクトを使って作業を行うユーザーは Contributors グループに登録するといった設定になります。大多数のユーザーは Contributors グループの権限があれば問題ないので、そこにユーザーを追加する例を挙げます。Contributors グループを選択している状態で [プロパティ] をクリックしてください。すると図 43 のようなプロパティ画面が表示されます。

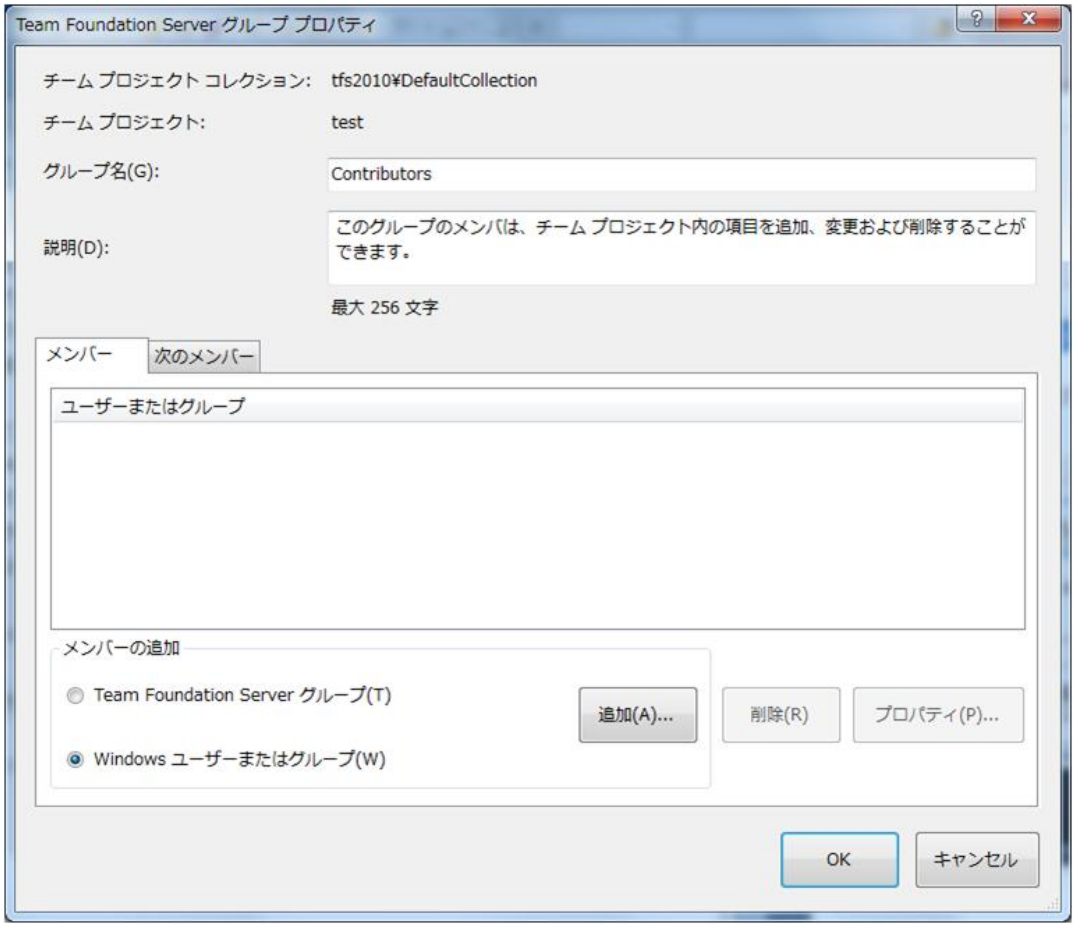


図 43: グループのプロパティ画面

ここで [メンバー] タブにユーザーまたはグループを追加することで、Contributors グループにユーザーを追加します。メンバーを追加する場合、[メンバーの追加] のところで [Team Foundation Server グループ] か、[Windows ユーザーまたはグループ] のいずれかを選択することができます。前者は、表 4 に示したような TFS に既に作成済みのグループを登録する場合に利用し、後者は Windows で管理されているユーザーまたはグループを登録する場合に利用します。[Windows ユーザーまたはグループ] を選択した状態で [追加] をクリックすると、Windows エクスプローラーのアクセス権設定でおなじみのウィンドウが表示されるので、ここでこれからチーム プロジェクトを利用するユーザーのアカウントやそのユーザーが所属するグループを追加するようにしてください。

[コラム] Active Directory がない環境での設定方法

TFS と上記設定を行う端末が Active Directory で管理されておらず、同一ワークグループにいない場合に、TFS がインストールされている端末とは別の端末を利用して上記の手順を実施しようとすると [Windows ユーザーまたはグループ] の名前を入力する部分で必要なユーザー名やグループ名を見つけ出すことができません。このような環境の場合には、TFS がインストールされている端末で上記手順を実施するようにしてください。

○ソース管理のアクセス権設定

2 つ目はソース管理のアクセス権の設定です。チーム プロジェクトのアクセス権を設定するとチーム プロジェクトは利用できるようになりますが、ソース管理部分は権限設定の部分がチーム プロジェクトとは独立しています。基本的には、チーム プロジェクトのセキュリティ設定で Contributors グループに所属するように設定したユーザーは

ソース管理も問題なく使えるようになっていますが、念のため確認しておくことにします。

ソース管理のアクセス権を設定するには、チーム エクスプローラーでチーム プロジェクトの [ソース管理] をダブルクリックし、ソース管理エクスプローラーを表示します。ソース管理エクスプローラーでは、フォルダー ビューからたとえば、チーム プロジェクト名と同名のフォルダーを選択し、コンテキスト メニューから [プロパティ] を選択します。プロパティ画面が開くので、[セキュリティ] タブを選択すると図 44 のような内容が表示されます。

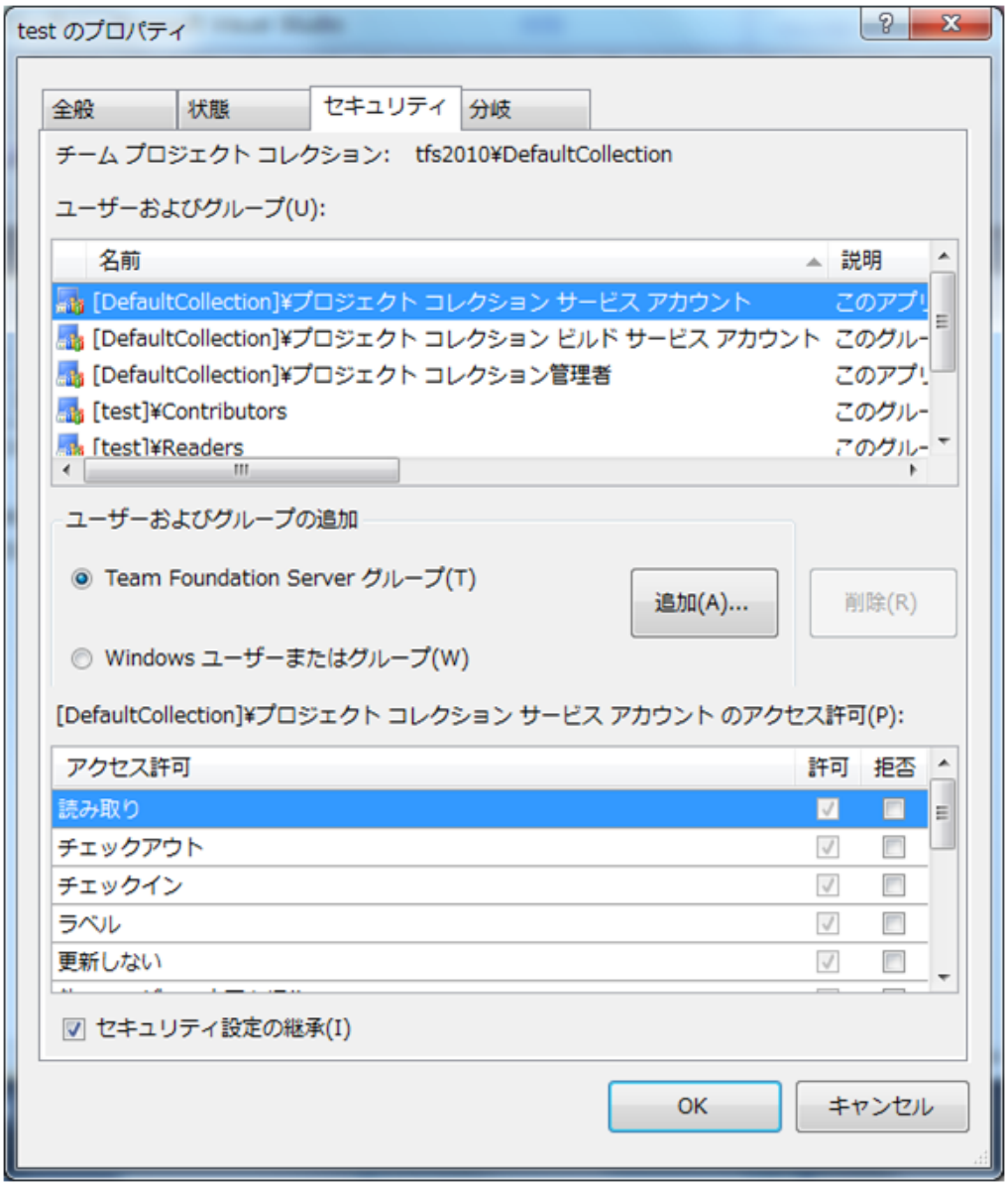


図 44: ソース管理のセキュリティ設定

[ユーザーおよびグループ] ボックスには、チーム プロジェクトのときと同様のグループが表示されています。それぞれのグループにはその下部にあるアクセス許可欄で権限設定が行われています。たとえば、[test]¥Contributors グループのアクセス許可を確認するとほとんどの項目が許可されていることが確認できます。

今回は特に変更する必要はありませんが、チーム プロジェクトのグループ メンバーシップで新しいグループを作成している場合には、そのグループがソース管理に対しても適切なアクセス許可を持てるように、こちらの設定も忘れないようにしてください。

●ソリューションの登録

ここまで来るのにずいぶん時間がかかりましたが、いよいよ開発対象となるソリューションの登録です。ソリューションの登録自体はチェックイン、チェックアウトの作業とさほど変わらないため非常に簡単です。

まず、VS を起動します。今までの設定が順調に進んできていればこの時点でチーム プロジェクトにも接続されているはずです。続けて、既存のソース管理されていないソリューションを開くか新しいソリューションを作成してください。ソリューションが完成したところで、コンテキスト メニューを開き、[ソリューションをソース管理に追加] を選択します。すると図 45 のような [ソリューション <ソリューション名> をソース管理に追加] ウィンドウが表示されます。

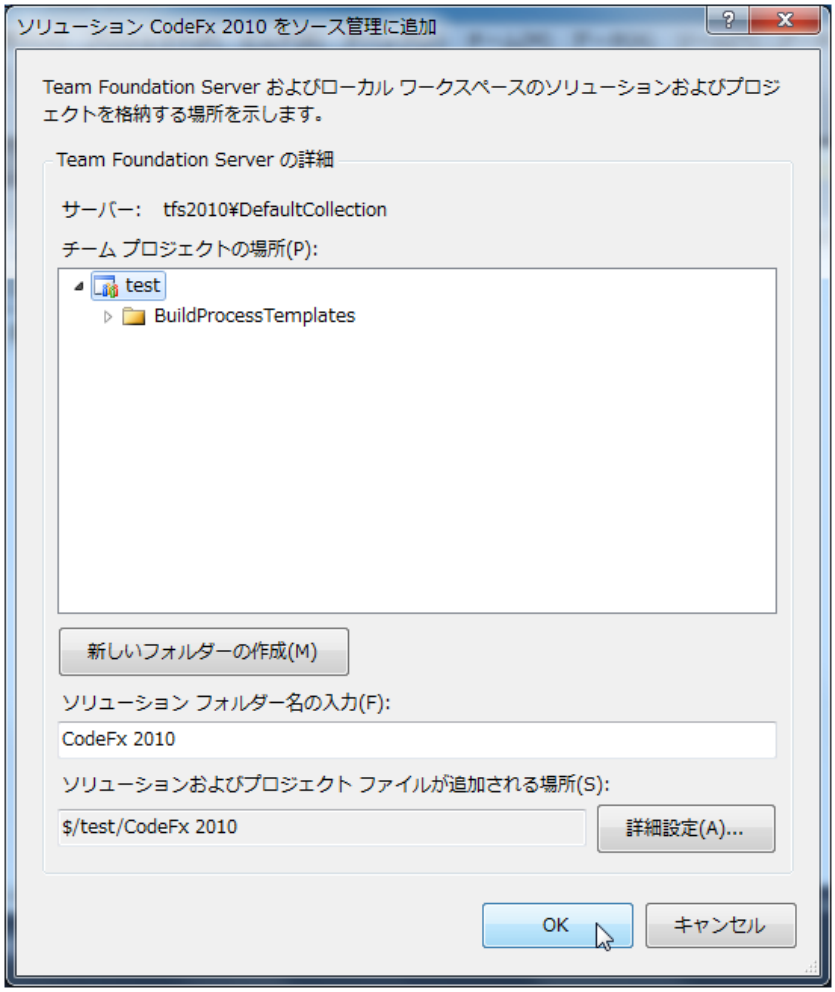


図 45: [ソリューション <ソリューション名> をソース管理に追加] ウィンドウ

チーム プロジェクトの場所を選択して、[OK] をクリックしてください。図 45 では本稿の中で作成してきた test というチーム プロジェクトを選択します。登録が終了すると、ソース管理エクスプローラーおよびソリューション エクスプローラーでは図 46 のような状態になります。

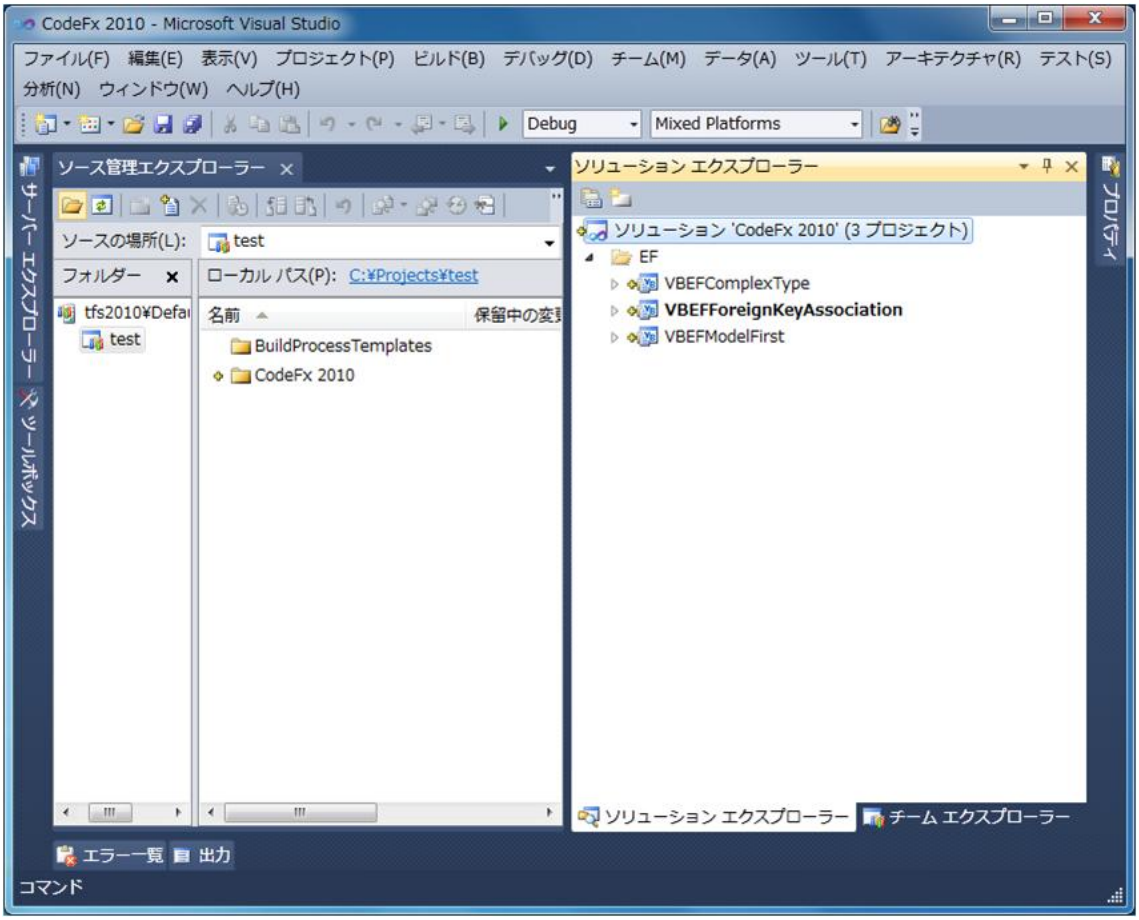


図 46: ソリューションの登録後の状態

この段階では、ソリューションがソース管理に登録されただけでまだチェックインはされていません。このため、たとえばソリューション エクスプローラー上では各プロジェクトなどに新規追加を示す＋マークが表示されている状態となっています。登録を確定するには再度ソリューションを選択し、コンテキスト メニューから [チェックイン] を選択して、前述のチェックインの操作を参考に変更内容を登録してください。

以上でソリューションの登録は終了です。これで開発者はソリューションを取得して自分の担当範囲の開発を始めることができるようになりました。

[コラム] ソリューションとプロジェクトの設計

通常、開発を始める場合にはある程度ソリューションとプロジェクトの構造を設計してから作業に臨んでいることと思います。ソース管理ツールを利用していない場合には、多少設計の詰めが甘い状態で開始した後に、比較的簡単に構造を変えてしまいがちですが、本質的にはこれはかなり危険な作業です。

ソース管理ツールを利用していると、構造の変更にはきちんとした手順が必要になるため、自ずと事前にきちんと設計しておいた方がよいということになります。特に今まできちんと計画されない構造変更でデグレーションを起こすなど問題を発生させていたような場合には、この機会に事前にできるだけきちんと考慮されたソリューションとプロジェクトの構造を作ったうえでソース管理に登録するように心がけるようにしましょう。

●自動ビルドの設定

さて、これで開発者が開発を始められる環境が整いました。しかし、せっかく TFS を利用するので、ソース管理

を使っていればすぐに利用できる自動ビルドについても設定してしまいましょう。

○自動ビルドとは

自動ビルドを初めて聞く方もいると思うので、簡単に解説しておきます。

普通に VS を使って開発をしている場合、ビルドは開発をしている本人が VS 上で実行するものです。自分の担当範囲が問題なくビルドできることを確認するうえでは、これでなんら問題はありません。しかし、ソース管理ツールを必要とするような開発では少なからず複数の開発者が参加していることでしょう。この場合、全員の分をまとめてビルドしても問題ないことを確認するのが後になればなるほど、問題があった場合の修正には膨大な時間を要することとなってしまいます。

このような状態を防ぐ目的で早期に全体をビルドした際の問題を把握し、早い段階で問題を是正していけると便利です。また、ビルド結果の確認と修正が短いサイクルで繰り返されれば、影響箇所を特定するのも容易で、その修正も比較的軽微なもので済ませられる可能性が高くなってきます。この短いサイクルの単位としてよく使用されるのが、日ごとにすべての変更を取り込んだビルドを実施することであったり、もっと細かい単位ではチェックインごとのビルドであったりします。しかし、このように短いサイクルのビルド作業をすべて手作業でやっていたのでは時間だけがかかってしまうので、ここで自動ビルドを利用します。利用イメージは図 47 のようになります。

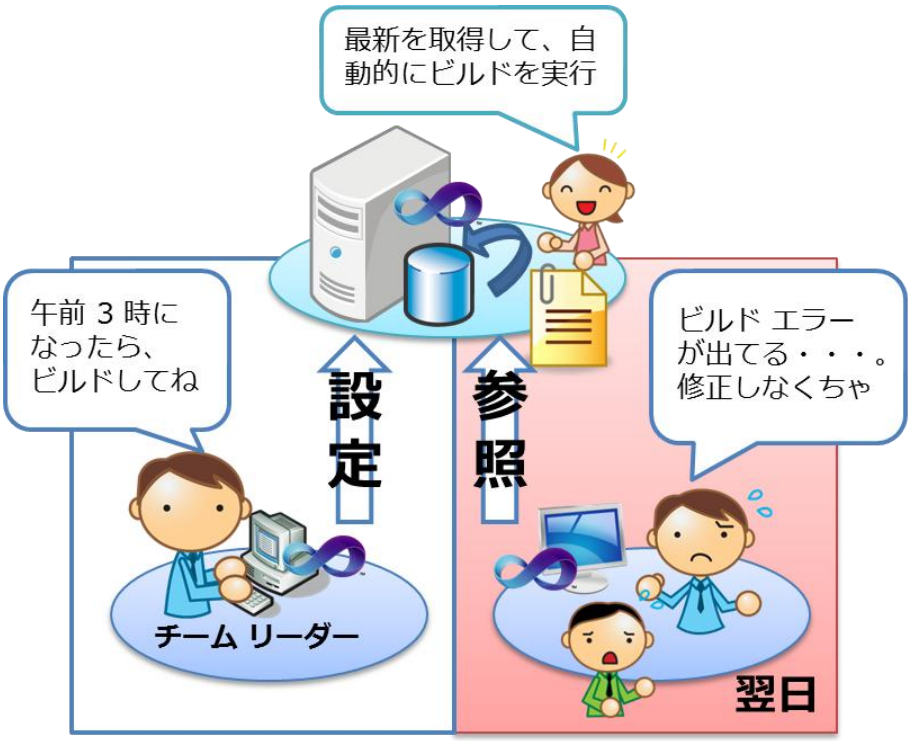


図 47: 自動ビルドとは

たとえば日次でチェックを行っていきたい場合に、深夜 3 時にビルドを行うように設定し、翌日にそのビルド結果を参照して、エラーがあればその日の作業の始めの作業としてその修正を行っていくという流れになります。このようにすることで、サーバーに登録されているファイルは常にビルド可能であることが保証され、最低限の品質であるビルド可能な状態を常に維持できるようになります。

○自動ビルド サービスの設定

自動ビルドを利用するには、TFS 側で追加の設定が必要です。

まず、Windows の [スタート] メニューの [すべてのプログラム] をポイントし、[Microsoft Team Foundation Server 2010] をポイントして [Team Foundation 管理コンソール] をクリックします。管理コンソールが開いたら左側のツリー メニューから [Team Foundation ビルド構成] を選択し、右側の領域で [インストール済み機能の構成] をクリックします (図 48)。

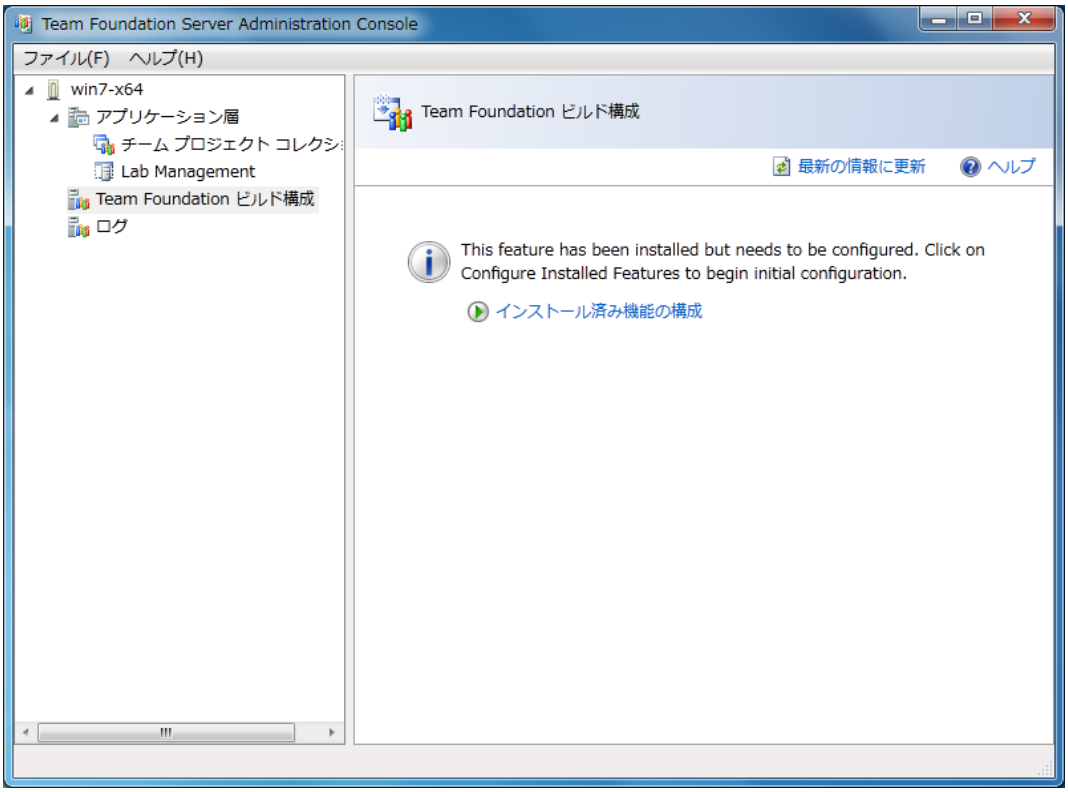


図 48: ビルド サービスの構成の開始

すると TFS の初期構成を行ったときと同じようなウィザード画面が表示されるので、[ウィザードの開始] をクリックしてください。最初の画面はウィザードの案内ページなので、そのまま [次へ] をクリックしましょう。

次の画面ではチーム プロジェクト コレクションを指定するテキスト ボックスがあります。特定のチーム プロジェクト用にビルド サービスを設定したい場合には、テキスト ボックス右側の [参照] ボタンから TFS に接続して、DefaultCollection というチーム プロジェクト コレクションを選択してください。この状態の例は図 49 のようになります。

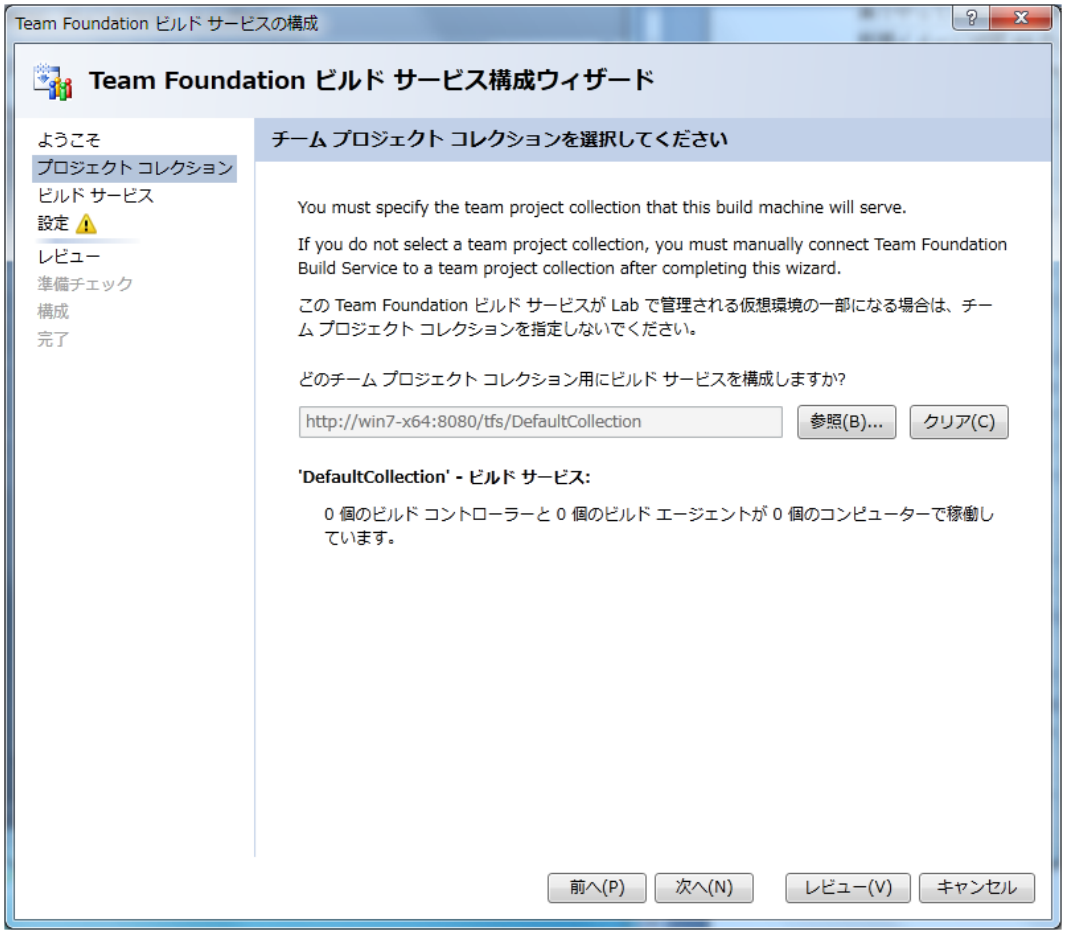


図 49: チーム プロジェクト コレクションの指定

次の [ビルド サービス] 画面では、ビルド サービスの数を設定できますが、ここは推奨設定のまま進めてしまってかまいません。[次へ] をクリックして先に進みましょう。[設定] 画面ではビルド サービスに利用するサービス アカウントと通信に利用するポート番号を図 50 のように設定します。

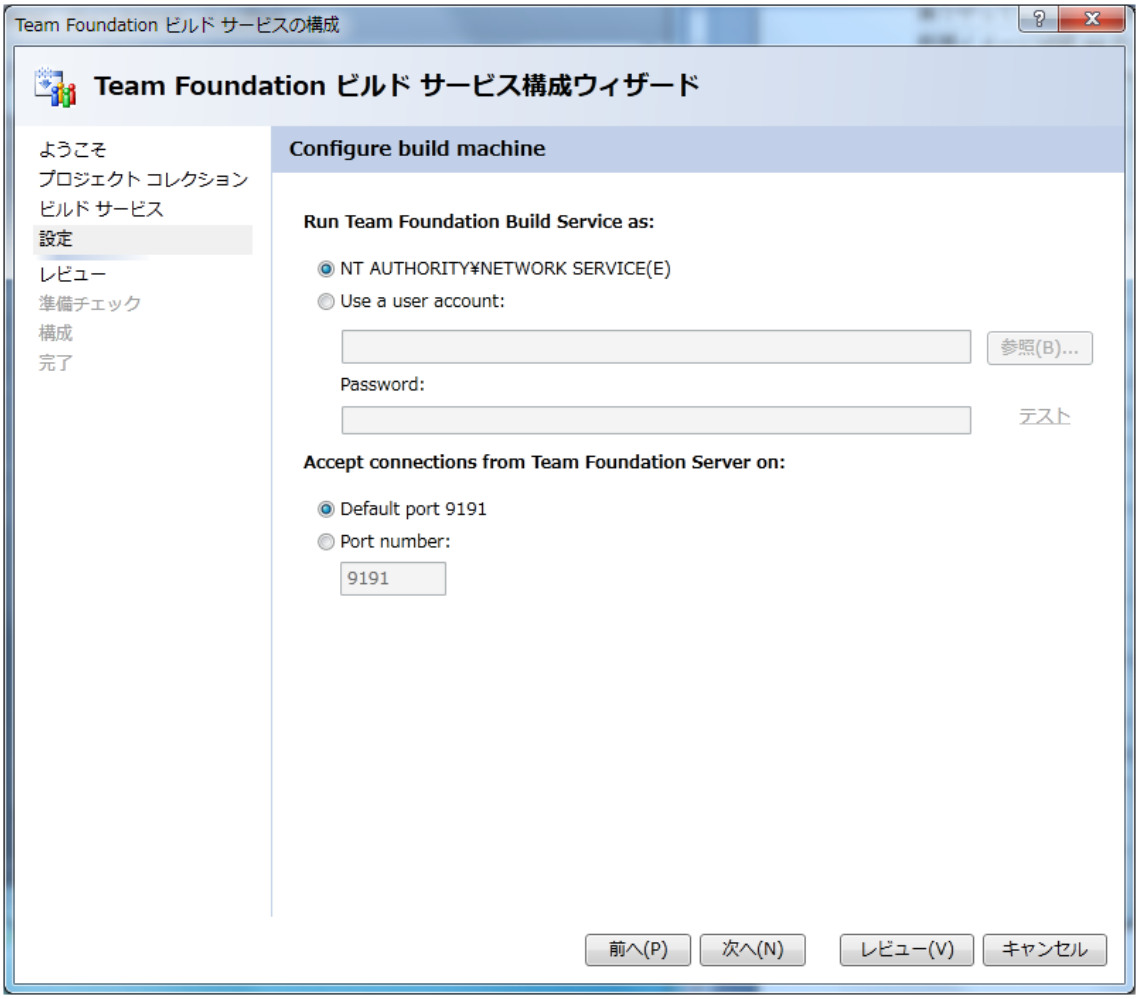


図 50: ビルド サービスの設定

ビルド サービスに利用するアカウントはできれば、専用に作成したユーザー アカウントが望ましいですが、Network Service でも問題ありません。ただし、Active Directory を利用している環境下では、Active Directory に登録されているユーザー アカウントを指定しておいた方が安全です。ポート番号は特に問題なければ既定のままで変更の必要はありません。

設定が完了したら [次へ] をクリックし、事前チェックでエラーが表示されなければそのまま [構成] をクリックしてサービスのセットアップを実行しましょう。セットアップが完了したら [次へ] をクリックし、最後の完了画面で [閉じる] をクリックすれば終了です。

○自動ビルドの設定

前述の例に基づいて、自動ビルドの設定をしてみましょう。チーム エクスプローラーでチーム プロジェクトの [ビルド] を選択し、コンテキスト メニューから [ビルド定義の新規作成] を選択します。すると、コード エディター上に新しいビルド定義の作成エディターが表示されます。左側に 6 つのメニューがあるので上から順番に設定していきます。1 つ目は [全般] の設定で、ここではビルド定義名という名前を設定します。どんなビルド定義なのかを表す名前を入力しておくとよいでしょう。2 つ目は [トリガー] の設定です。ここは自動ビルドがどのようなタイミングで実行されるかを指定する部分で、自動ビルドの肝となる設定です。前述の例に基づくと毎日 3 時に実行してほしいので、この場合には図 51 のようになります。

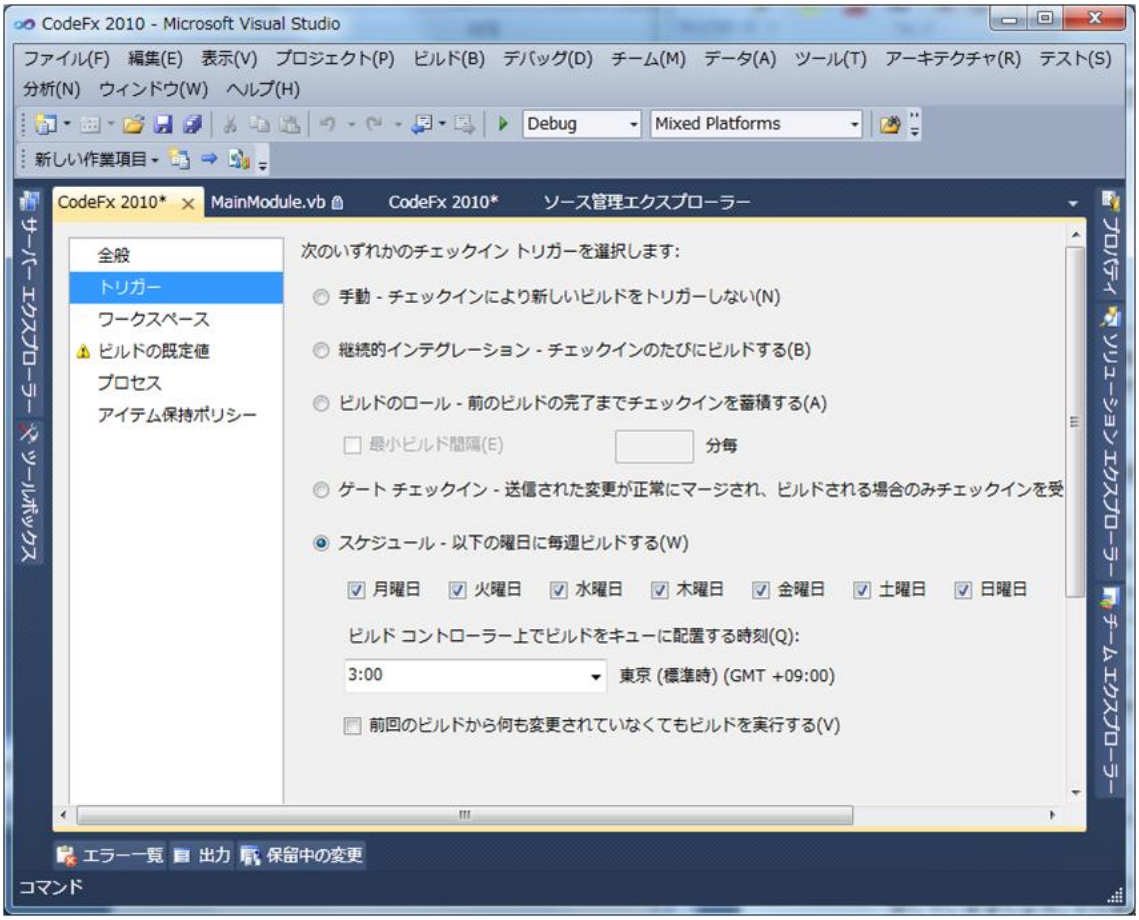


図 51: ビルド定義のトリガー設定

図 51 で指定しているスケジュールを含め、それぞれの項目の意味については▲表 5 を参照してください。

トリガー名	説明
手動	利用者によって明示的に実行されるまでビルドが行われません。
継続的インテグレーション	ユーザーによってチェックインが行われるたびにビルドが実行されます。
ビルドのロール	ビルドが指定された時間間隔で定期的に実行されるようになります。
ゲート チェックイン	ユーザーによってチェックインが行われた際、実際にはチェックインを保留した状態でビルドが実行され、ビルドに成功した後でチェックインが行われるようになります。
スケジュール	指定した曜日の指定した時間にビルドが実行されます。

▲表 5: 自動ビルドのトリガーの種類

自動ビルドでよく利用されるトリガーは [スケジュール] ですが、アジャイル開発などでは [継続的インテグレーション] も利用されることが多いかもしれません。ゲート チェックインは TFS 2010 の新機能で、継続的インテグレーションがチェックイン後のビルドなのに対して、ゲート チェックインはチェックイン前にビルドを行い、成功後にチェックインを行うところが決定的に異なります。

3 つ目は [ワークスペース] の設定です。ここでは、ソース管理フォルダー上のどのフォルダーの内容をビルドするかを設定します。ソース管理エクスプローラーでチーム プロジェクトのフォルダー配下にあるファイルをビルドする設定が既定で入っているので、とりあえずこのままでかまいません。

4 つ目は [ビルドの既定値] の設定です。ここではビルド コントローラーとビルドの出力パスを設定します。ビルド コントローラーは先ほどビルド サービスの設定で設定したものが既定値で選ばれているので、このままでかまいません。ビルドの出力パスはビルドのサービス アカウントが書き込み可能な共有フォルダーを指定してください。単純には TFS をインストールしている端末に共有フォルダーを作成して、そのフォルダーにフル コントロール権限を設定した状態でそのフォルダー パスを入力すれば完了です。例として、図 52 では「¥¥tfs2010¥result」フォルダーを同様の条件で作成して設定してあります。

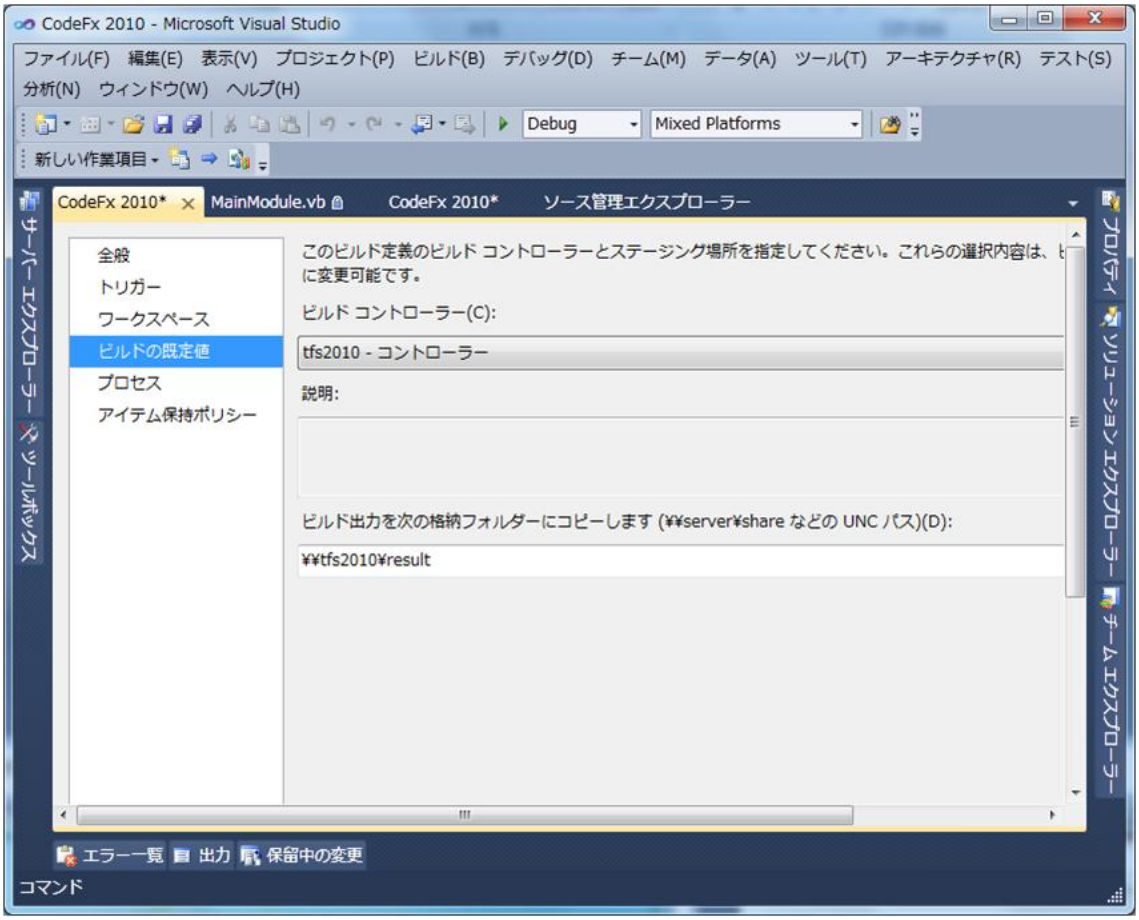


図 52: ビルドの既定値の設定

5 つ目の [プロセス] では、自動ビルドの手順をどのように構成するかを設定します。いろいろと設定することができますが、手始めにコンパイルだけができればよいので、このままの設定にしておいてかまいません。いずれもう少し複雑な処理を行いたくなった場合に見直しを行ってください。

最後は [アイテム保持ポリシー] の設定です。これはビルド結果をどのように保存しておくかという設定です。標準では大体がそれぞれごとに 10 個の履歴を保持するようになっていますが、これについては好きなように設定してかまいません。すべての設定が完了したら、Ctrl + S キーなどで保存してください。すると、チーム エクスプローラーではチーム プロジェクトの [ビルド] の配下に作成したビルド定義が表示されるようになります。

今回の設定では、毎日 3 時になると自動的にビルドを実行することができますが、すぐに手動で実行したい場合に

は、利用したいビルド定義 (図 51、図 52 の例の場合には CodeFx 2010 という名前) を選択し、コンテキスト メニューから [新しいビルドをキューに配置] を選択してください。ビルドを実行するための設定の画面が表示されますが、そのまま [キューに登録] を選択して実行します。すると、ビルド エクスプローラーが表示され、ビルドが実行されます。ビルド エクスプローラーで実行中のビルド定義名をダブルクリックすると、進行状況の詳細を確認することができ、終了すると図 53 のような画面で確認することができます。

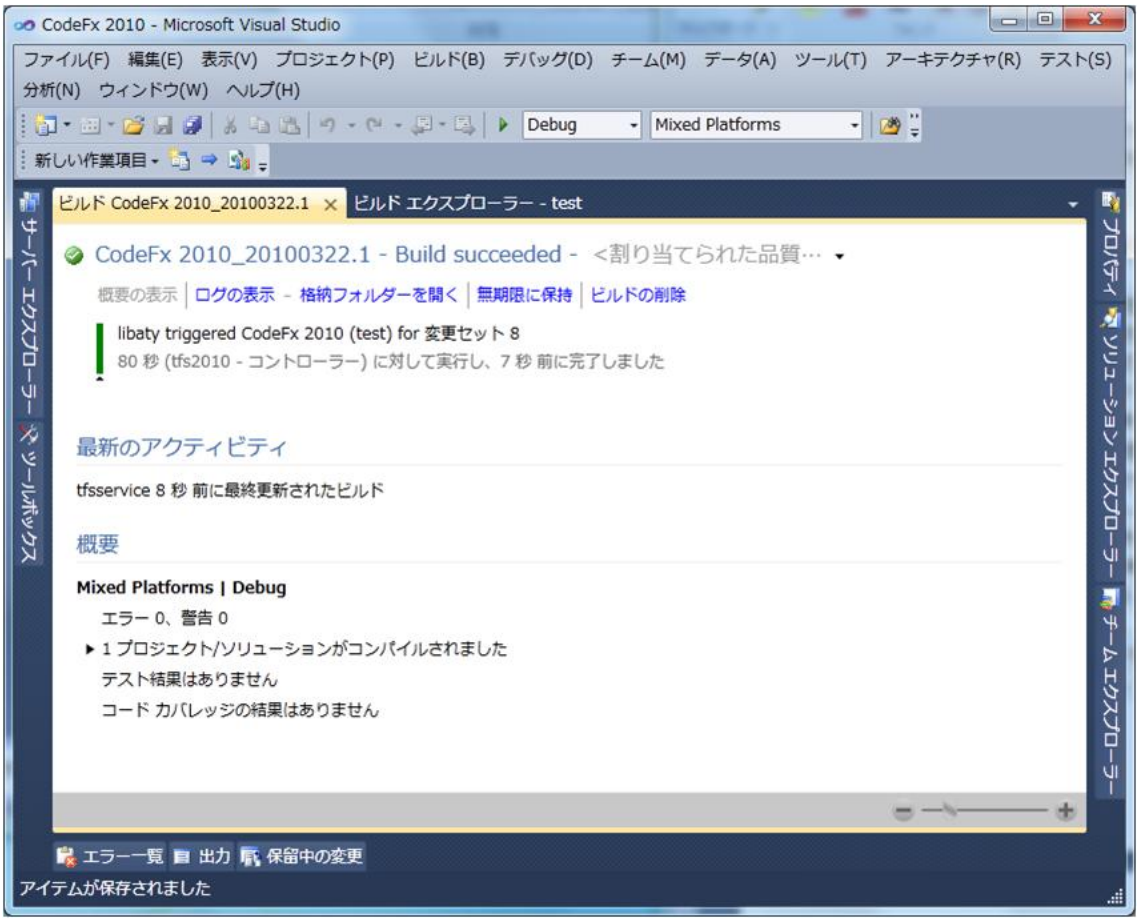


図 53: ビルドの結果

図 53 は正常にビルドが終了したことを示していますが、ここでビルド エラーがある場合には、概要欄にその内容が表示されるので、確認して、エラーを修正できます。

これで、後は勝手に毎日ビルドが実行されるので、次の日に作業を開始する際にはまずは自動ビルドの結果を確認し、エラーがあれば修正する作業から始めるようにします。そうすることで、最低でも全体のコンパイルができる品質を常に確保しておくことができます。

■ソース管理の高度なタスク

ここまででソース管理の使い方を覚え、ソース管理を使い始めるための準備についても覚えることができました。このまま利用するだけでもソース コードを安全に管理し、かつ一定の品質を保ちながら開発を継続していくことが可能です。しかし、TFS には品質向上のためにもう少しいろいろと機能が用意されています。そこで、最後にそれらの機能をざっと俯瞰してみることにします。もしすぐに使ってみたいという機能があった場合には、MSDN ライブラリへのリンクを参考により詳細な内容を把握してください。

●チェックイン ポリシーの利用

チェックイン ポリシーはその名のとおり、チェックインに制約を設ける機能のことです。まずは図 54 を見てください。

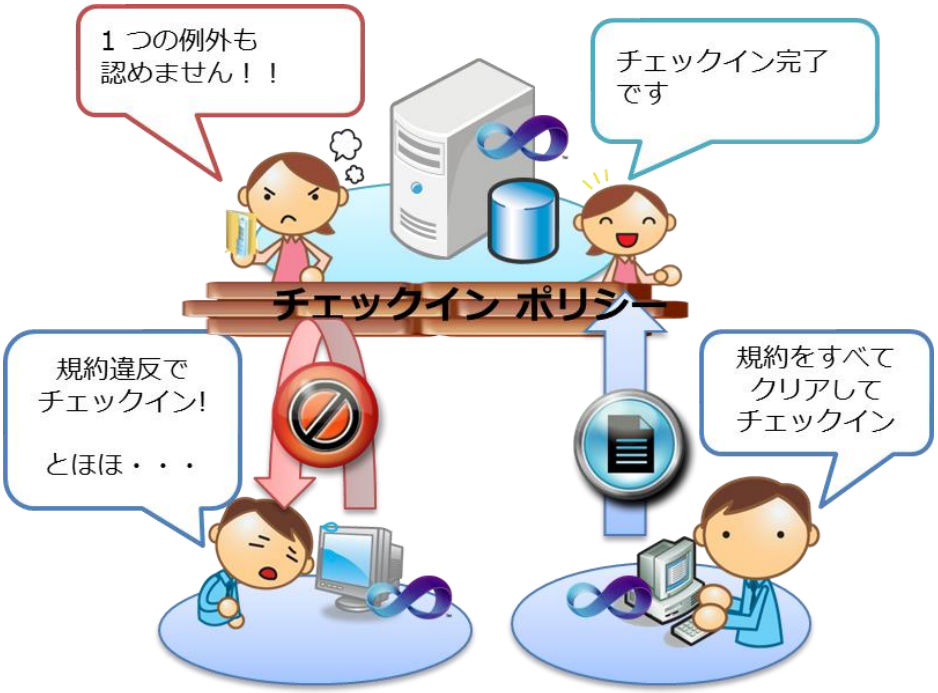


図 54: チェックイン ポリシーのイメージ

チェックイン ポリシーが設定されていると、チェックインを行おうとした場合、ポリシー違反がないかどうかの確認が行われます。この際、ポリシー違反があると図 54 の左のようにチェックインができません。この場合には、ポリシー違反として報告された内容を修正したうえで再度チェックインを行うと、図 54 の右のようにチェックインできるようになります。

このようにすることで、ソース管理に登録されているファイルは常にポリシーを満たしていることが保証されるため、より高品質なコードだけが存在していることになります。自動ビルドを設定していれば、ビルドの結果によって全体の状況を確認できますが、チェックイン ポリシーを利用することでそれよりも早い段階で、または別のアプローチによる手段として低品質なコードがないかどうかを確認できるというわけです。

実際にチェックイン ポリシーを設定する場合には、チーム エクスプローラーでチーム プロジェクトを選択し、コンテキスト メニューから [チーム プロジェクトの設定] – [ソース管理] を選択します。表示された画面で [チェックイン ポリシー] タブを選択し、[追加] をクリックすると図 55 のように 4 種類のチェックイン ポリシーが表示されます。

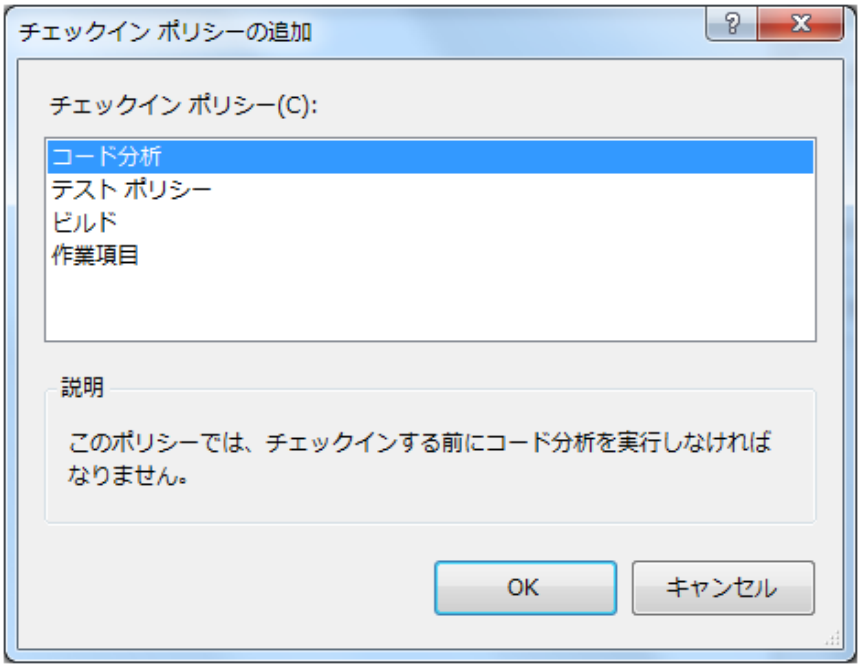


図 55: チェックイン ポリシーの種類

ここでは詳細は割愛しますが、仮にコード分析を有効にした状態で、ソース コードを編集してチェックインを行おうとしてみます。いつもどおりチェックインの確認ダイアログ ボックスが表示されますが、ここで左側から [ポリシー警告] を選択すると、図 56 にあるようにポリシーが満たされていないという警告が表示されました。

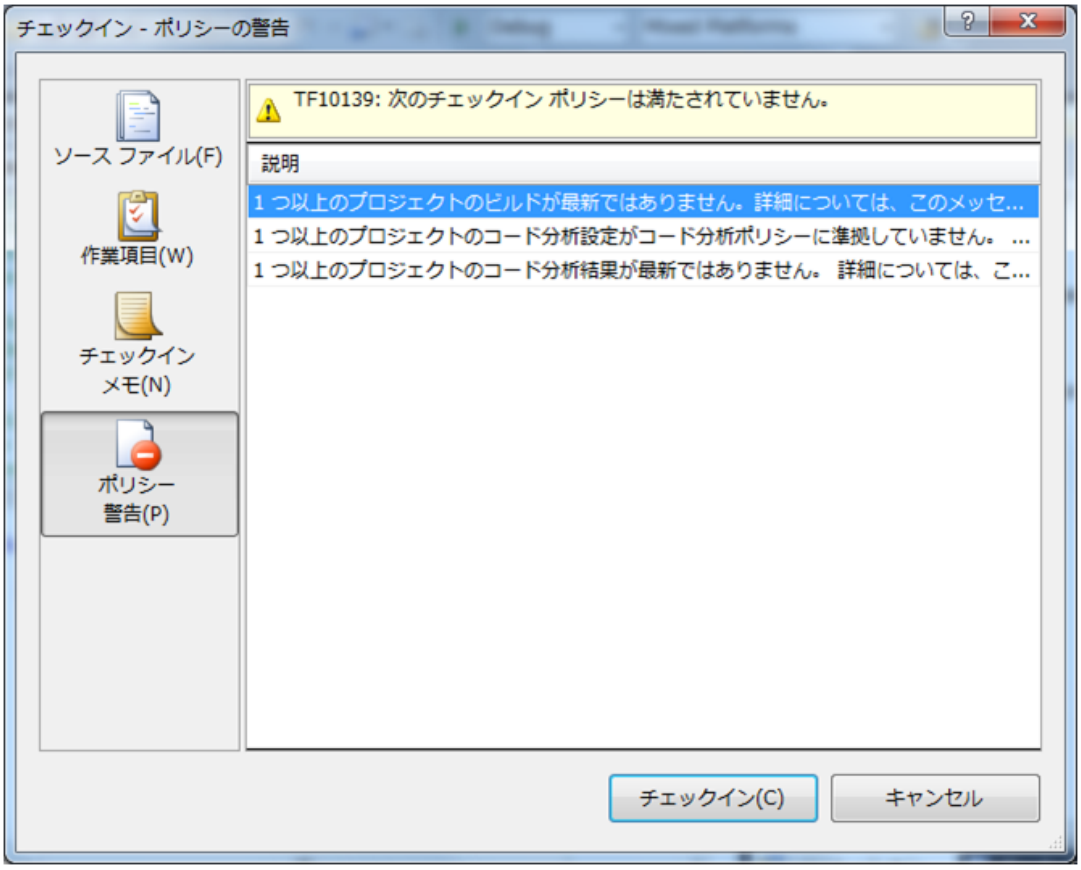


図 56: チェックイン ポリシー違反の警告

チェックインを行いたいユーザーはこの警告に従って、内容を修正したうえで再度チェックインを行うという流れになります。ここでは詳細は割愛するので、チェックイン ポリシーの詳細に関しては MSDN ライブラリの品質ゲートの設定と実行 [http://msdn.microsoft.com/ja-jp/library/ms181458(VS.100).aspx] をご覧ください。

●シェルブの利用

シェルブとは聞き慣れない言葉ではないでしょうか。少なくとも VSS には同様の機能は存在していませんでした。シェルブを和訳すると「棚上げ」となりますが、これはチェックインを棚上げする機能であることを表しています。なぜそのような機能が必要かというと、チェックイン ポリシーを満たしていないソース コードをとりあえず TFS 上に登録しておきたいというニーズを満たすためです。ソース管理ツールを導入する目的の 1 つにソース コードを安全にバックアップしておくという目的がありますが、チェックイン ポリシーを満たせずにソース コードがチェックインできないとファイルが利用者の端末内にしか存在しないことになり、この目的を満たすことができなくなってしまいます。だからといって、チェックイン ポリシーを満たしていないソースをチェックインさせるわけにもいかないので、その中間の機能としてシェルブが用意されています。

この機能があることでチェックイン ポリシーを満たしていない場合でもシェルブを実行して、ソースを TFS に登録することができます。これによって、チェックイン ポリシーを使いたい、けれど安全にソース コードのバックアップを保持しておきたいという一見相反する要求を満たすことができます。

シェルブはユーザーから見た場合にはチェックインとほとんど変わりません。シェルブを実行したいファイル (またはプロジェクトなど) でコンテキスト メニューから [保留中の変更をシェルブ] を選択します。すると図 57 のような画面が表示されるので、シェルブに名前を付けて [シェルブ] を実行するだけです。

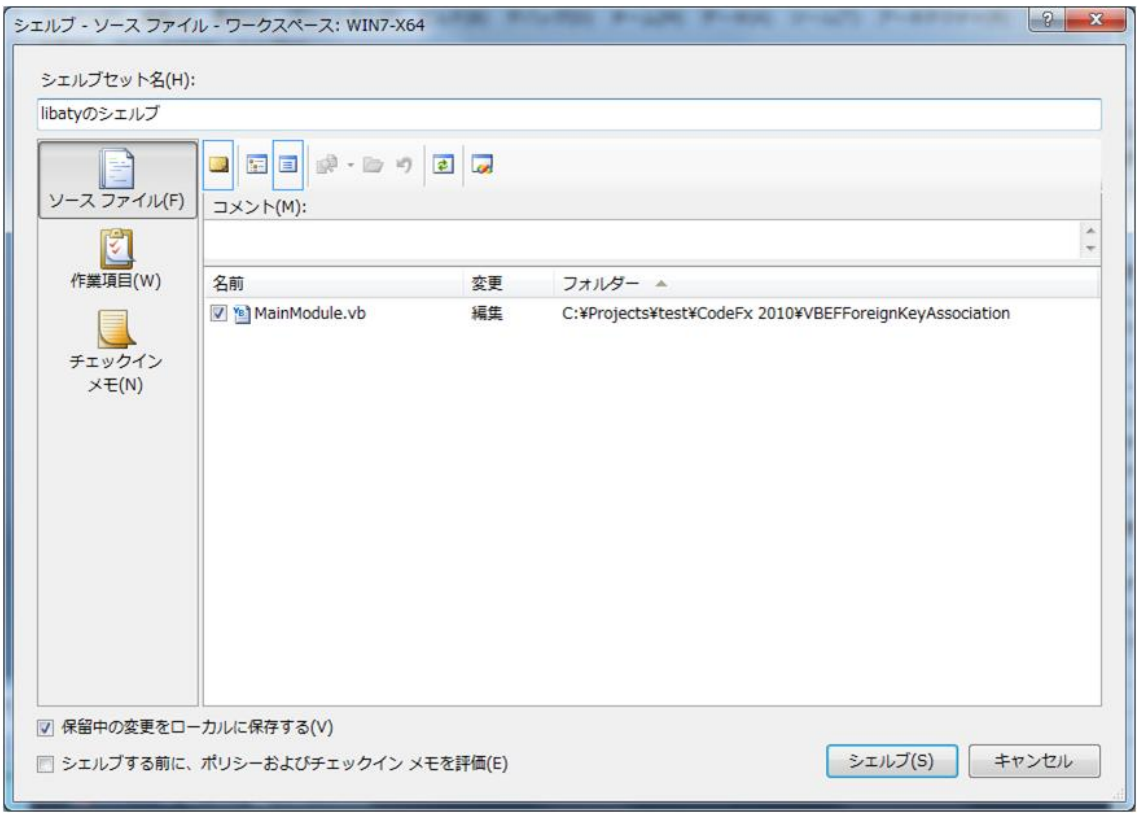


図 57: シェルブの実行

ただし、シェルブを実行した場合はチェックインが行われているわけではないので、ソリューション エクスプローラーで見た場合の対象ファイルに付いているアイコンは変更されません。よりスマートにシェルブを利用するには再度コンテキスト メニューから [保留中の変更を元に戻す] を選択しておくとよいでしょう。

チェックアウトに相当する操作はアンシェルブと言います。アンシェルブを実行したいファイルのコンテキスト メニューから [保留中の変更をアンシェルブ] を選択します。今度は図 58 のようなアンシェルブの画面が表示されるので、シェルブの一覧から取得したいものを選択して、[アンシェルブ] を実行します。

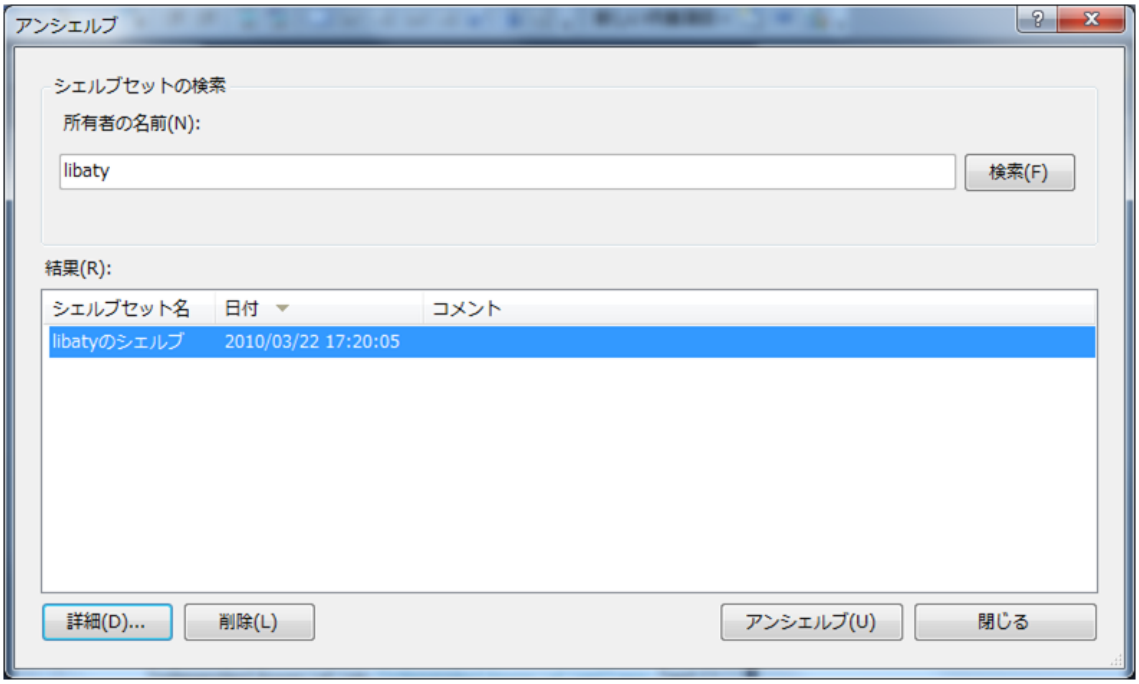


図 58: アンシェルブの実行

すると、自端末のファイルはシェルブしたときのファイルに置き換わるので、ここから作業を継続できます。

シェルブに登録したファイルは、他のユーザーがアンシェルブを実行することで共有することもできます。たとえば、コード レビューを実施する場合のフローとして、開発者がシェルブに登録し、レビューアーが対象ソース コードをアンシェルブしてレビューを実施し、開発者はレビュー結果を修正したうえでチェックインするという使い方もできます。

シェルブの場合も変更セット同様にシェルブセットという考え方が導入されていて、どのファイルがまとめてシェルブされたのかがわかるので、レビュー対象コードがどれなのかを見つけ出すことも容易で、品質向上に効果が期待できます。

シェルブについての詳細は MSDN ライブラリのシェルブセットの操作 [http://msdn.microsoft.com/ja-jp/library/ms181403(VS.100).aspx] を参照してください。

●BVT の利用

BVT (Build Verification Test) も、アジャイル開発のプラクティスの 1 つである継続的インテグレーションというものを実施していないとなかなか耳慣れない言葉かもしれません。

これは、自動ビルドのプロセスの一部として一緒にビルド結果を確認するテストを組み込んでしまおうというもので

す。単純には単体テストを実行することが多いですが、TFS では他にも静的コード分析を組み込めるようになっています。チェックイン ポリシーにも (静的) コード分析、単体テストがありましたが、自動ビルドにもこれらを組み込めるというわけです。チェックイン ポリシーでは、「チェックイン対象のソース コードがどうか」という視点で確認するのに対して、自動ビルドでは「ビルド対象全体でどうか」という視点で確認するため、両方設定することでより厳重にチェック プロセスを敷くことができます。このような考え方は多層の品質ゲートと呼ばれるもので、品質をクリアするためのゲートを多重に設定することで、1 つクリアするたびに目的の品質を満たしていくことになります。

BVT を設定するには、チーム エクスプローラーでチーム プロジェクトのビルド内の編集したいビルド定義を選択し、コンテキスト メニューから [ビルド定義の編集] を選択します。すると、自動ビルドの設定の部分で行ったのとまったく同じ画面が表示されます。今回は、[プロセス] 部分にだけ注目しましょう。プロセスの部分を見ると図 59 のようにビルド プロセス パラメーターがリスト表示されている部分があります。

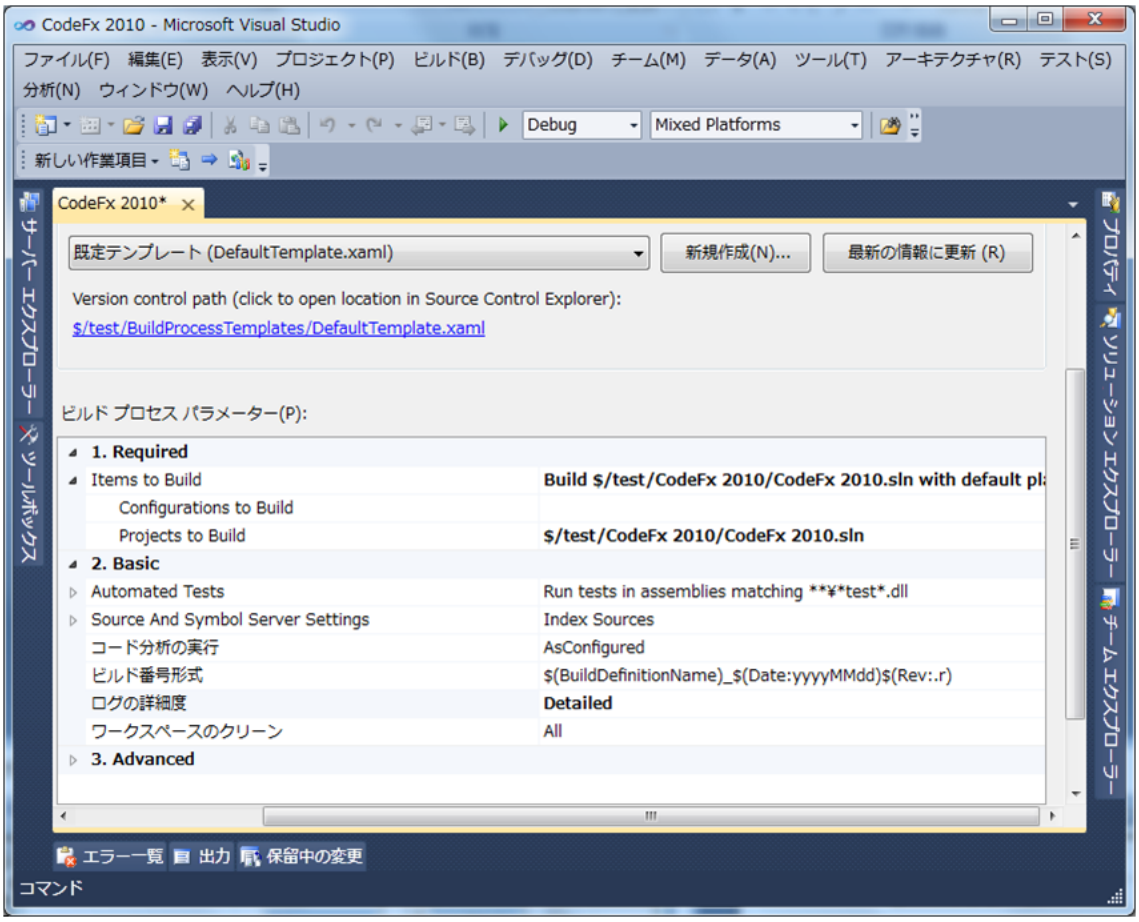


図 59: ビルド プロセス パラメーター

BVT という観点でビルド プロセス パラメーターを見る場合には、2.Basic のカテゴリ内にある、[Automated Tests] と [コード分析の実行] に注目します。設定内容を見ると、[Automated Tests] は "Run tests in assemblies matching **¥*test*.dll" となっています。実は普通にビルド定義を作っただけでも dll 名に test という名前が含まれているものがあればそれが単体テストを含んだ dll であると解釈してテストを実行します。[コード分析の実行] については "AsConfigured" となっています。こちらも対象のプロジェクト ファイルで静的コード分析が有効になっていれば、自動ビルドの中で実行するようにあらかじめ設定されています。これらは要件に合わせて変更できますが、既に設定が存在しているのでこれに合わせて BVT が実行されるようにソリューションやプロジェクトを構成し

てみてはどうでしょうか。

BVT の詳細については MSDN ライブラリの既定のテンプレートを使用して、ビルドを定義します [http://msdn.microsoft.com/ja-jp/library/dd647547(VS.100).aspx] を参考にしてください。

[コラム] 多層品質ゲートは必要か?

品質ゲートを設けることで高品質なコードを維持し続けられることは間違いありません。品質を維持するという視点だけで見れば、品質ゲートはいくつあっても無駄になることはないでしょう。

しかし、高品質なコードを維持するためには、いかにツールの支援があったとしてもそれなりのコストを払う必要があります。コストとスケジュール、求められる要求などの複数の条件を総合的に見たときには、多層の品質ゲートを設けることが過剰品質につながる可能性もあります。品質という観点だけで見ると品質が良いに越したことはありませんが、実際に運用していく場合にはバランスを考えて設定するようにしてください。

ただし、単に余計に時間がかかるなら設定しないと安易に考えているのであれば、まずは試しに設定してみることをお勧めします。実際にやってみて、コストに見合う結果が得られないようであれば、やり方を見直す、それでも無理ならば最終的にはゲートを取り除くというアプローチをしてみてください。

●作業項目との連携

開発作業を行っている際、最終的な成果物は動くアプリケーションですが、そこに至る前には要件を決定し、見た目 (ユーザー インターフェイス) を決定し、アプリケーションの構造設計をし、と他の作業も合わせてさまざまな作業が必要となります。TFS では実際にできあがる成果物だけではなく、この作業そのものをも管理することができます。それを可能にするのが TFS に用意されている作業項目管理機能です。ソース管理とは直接は関係がありませんが、TFS ではソース管理、自動ビルドに次いで非常に大切な位置付けになっている機能なので、作業項目管理機能について確認した後に、ソース管理と作業項目管理の連携について解説することにします。

○作業項目管理とは

この機能で行われるのは、作業そのものの管理なので、単純には 1 つ 1 つの作業を TFS に登録し、その 1 つ 1 つで作業の内容、担当者、開始日、終了日などを管理していくことで作業の進捗を把握していくことになります。要求内容そのものを登録しておいたり、テスト内容を登録したり、テストによって発見されたバグを登録したりといったことが行え、最終的な成果物であるアプリケーションを作る過程で発生するさまざまなものが作業項目管理機能によって管理されていくことになります。イメージをつかみやすくするために図 60 をご覧ください。

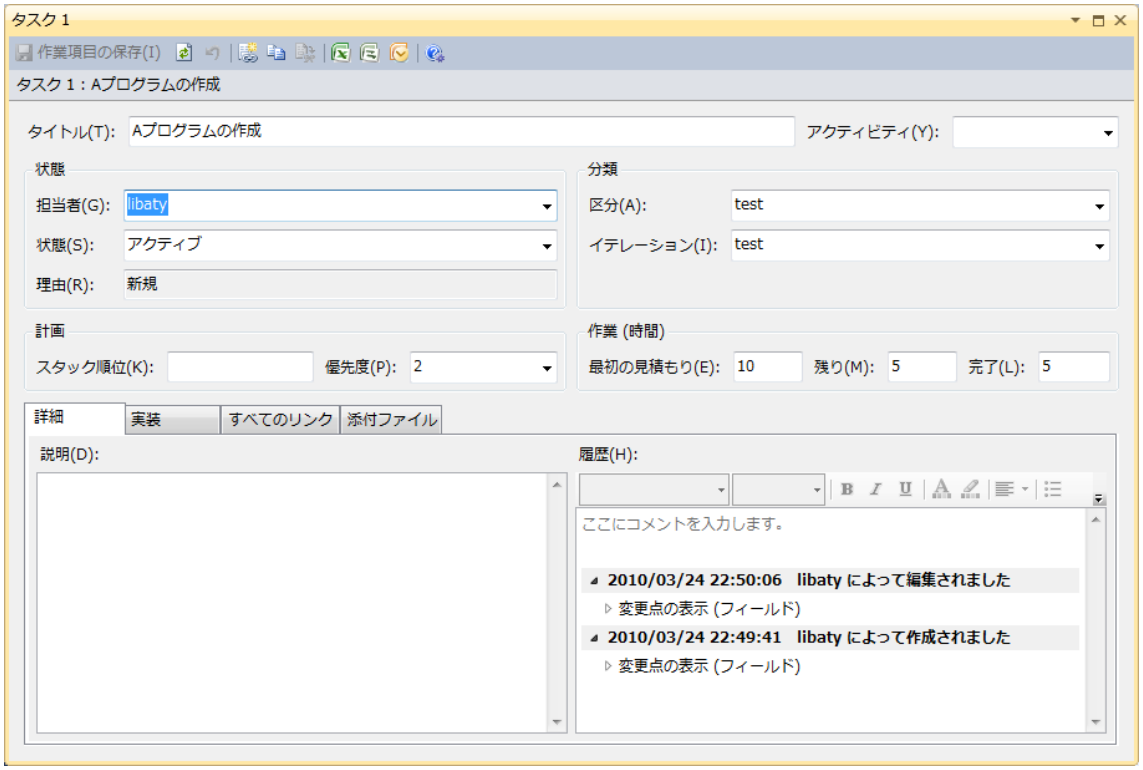


図 60: 作業項目の画面

この画面は作業項目のうち、タスクというものを編集している画面です。タイトルや担当者、状態、予定作業時間（最初の見積もり）や残存作業時間（残り）などの項目があり、作業の進捗が管理されていることがわかります。また、右下には履歴欄があり、各行を展開すると作業項目が保存されるたびにその変更履歴がすべて記録されています。このように 1 つ 1 つの作業を TFS にタスクとして登録しておくことで、日々の推移をすべてデータとして蓄積し、後から簡単に参照できるようにすることが最大のメリットとなっています。なお、どのような作業項目が利用できるかはチーム プロジェクト作成時に軽く触れたプロセス テンプレートによって決められており、MSF Agile と MSF CMMI ではそれぞれ

作業項目の種類	Agile	CMMI	説明
ユーザー ストーリー	○		ユーザーの利用シナリオの管理に利用するもの
タスク	○	○	設計や実装など各種作業の内容の管理に利用するもの
テストケース	○	○	実施する予定のテスト内容の管理に利用するもの
共有ステップ	○	○	テストケースのうち、共有可能な手順の管理に利用するもの
バグ	○	○	テストで発見されたバグ内容の管理に利用するもの
懸案事項	○	○	問題や障害、懸案事項の管理に利用するもの
変更要求		○	変更要求のトラッキングや基準計画への変更の管理に利用するもの
必要条件		○	開発に求められる各種の要件の管理に利用するもの
レビュー		○	設計書やソースのレビュー結果の管理に利用するもの
リスク		○	リスクの内容と対処方法の管理に利用するもの

表 6 のとおりです。

作業項目の種類	Agile	CMMI	説明
ユーザー ストーリー	○		ユーザーの利用シナリオの管理に利用するもの
タスク	○	○	設計や実装など各種作業の内容の管理に利用するもの
テストケース	○	○	実施する予定のテスト内容の管理に利用するもの
共有ステップ	○	○	テストケースのうち、共有可能な手順の管理に利用するもの
バグ	○	○	テストで発見されたバグ内容の管理に利用するもの
懸案事項	○	○	問題や障害、懸案事項の管理に利用するもの
変更要求		○	変更要求のトラッキングや基準計画への変更の管理に利用するもの
必要条件		○	開発に求められる各種の要件の管理に利用するもの
レビュー		○	設計書やソースのレビュー結果の管理に利用するもの
リスク		○	リスクの内容と対処方法の管理に利用するもの

表 6: MSF Agile と MSF CMMI の作業項目の種類

実際に作業項目を利用する場合にはこれらの種類を使い分けて管理に必要なデータの蓄積を行っていくことになります。

このようにしてデータをためておき、後で作業の結果を確認したい場合には、作業項目の履歴をたどることで、作業の流れを知ることができますが、その際にその作業によって作成されたソースがどんなものかを確認することができればさらに便利です。そのためにチェックインの際に、そのチェックイン内容を作業項目に関連付けるという機能が用意されています。作業項目とチェックインが関連付けられることにより、特定のソース コードがどの作業の成果物なのかを簡単に把握できるようになります。

○作業項目への関連付け

では、チェックインを行う際に作業項目への関連付けを行う手順を確認します。これは非常に簡単な手順で行うことができます。チェックインを行う際に表示されるチェックインの確認画面で左側から [作業項目] を選択してください。すると図 61 のように右側には標準設定で自分の担当作業項目の一覧が表示されます。

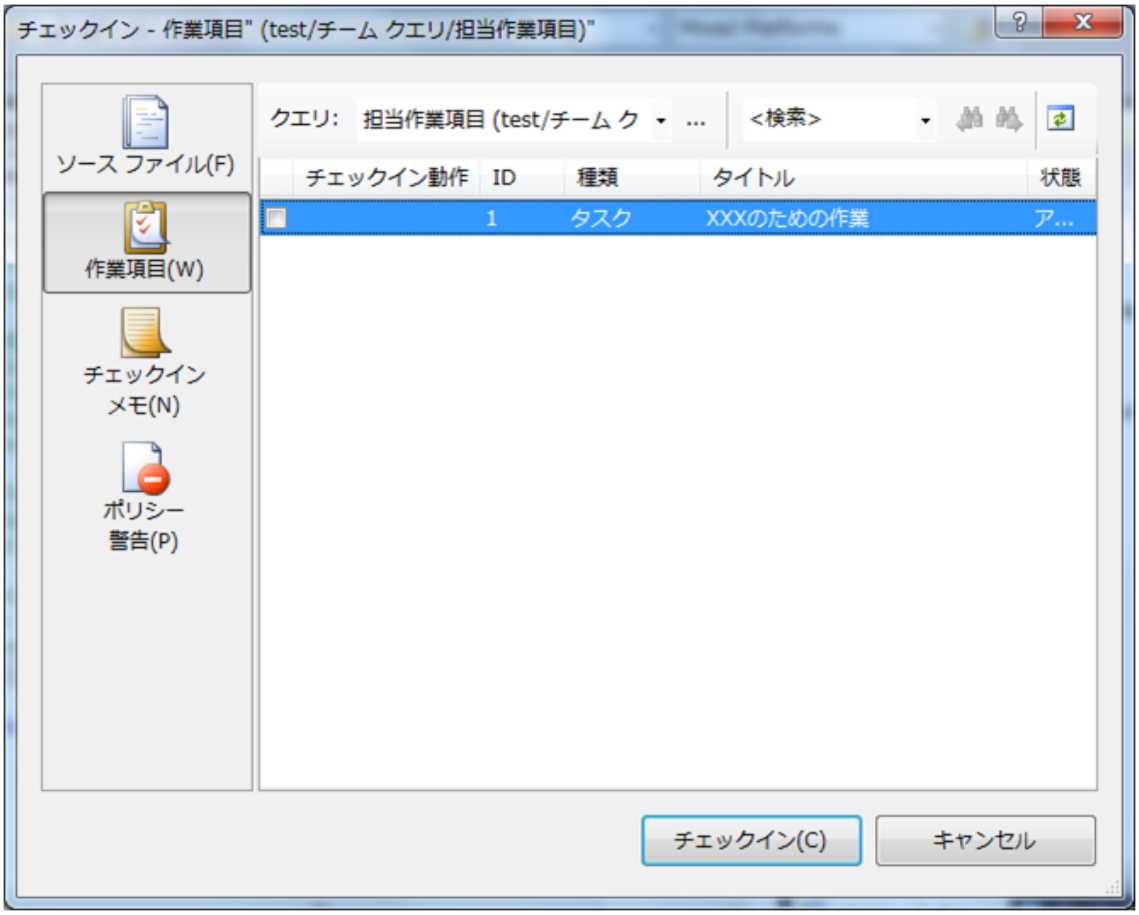


図 61: 作業項目への関連付け

ここで、該当作業項目の左列にあるチェック ボックスをオンにしたうえでチェックインを行うと作業項目に関連付けがされます。なお、該当する作業項目が見つけられない場合には、上部にあるクエリのドロップダウン リストから検索する作業項目の種類を変更することもできます。

関連付けを行った作業項目の内容を見てみると、図 62 の下段に表示されている "変更セット 9" のように先ほどのチェックインがこの作業項目に関連付けられていることがわかります。

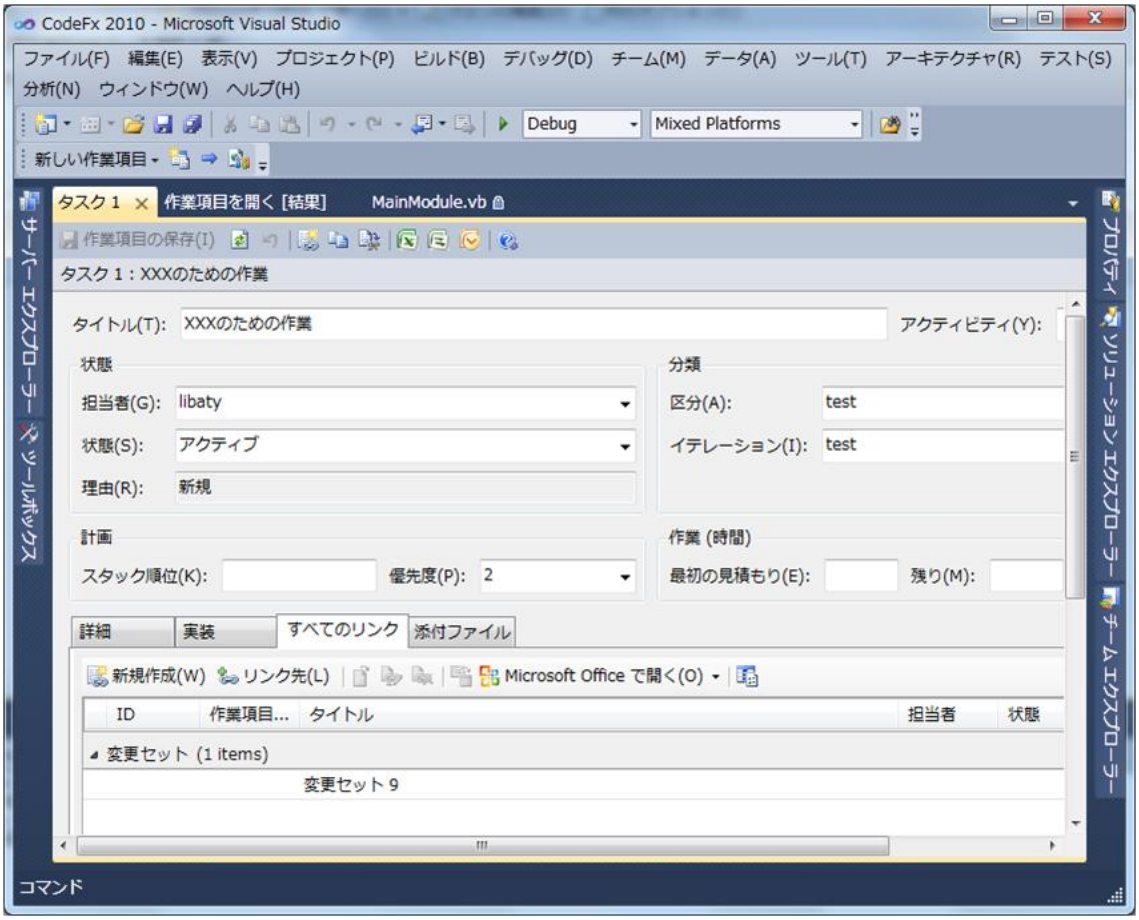


図 62: 作業項目で変更セットの確認

作業項目の使い方について詳細には説明していないため、少しイメージしにくいかもしれませんが、作業とその結果を結び付けて管理する術があるということを把握していただければ問題ありません。作業項目の詳細については MSDN ライブラリの バグ、タスク、およびその他の作業項目追跡 [http://msdn.microsoft.com/ja-jp/library/dd286718(VS.100).aspx] に説明があるのでそちらをご覧ください。

●VSS からの移行

TFS には当初から VSS などのソース管理ツールから TFS へリポジトリ データを移行するためのツールが付属しています。もちろん TFS を利用してもらいたいがために用意されているものですが、既に VSS を利用している場合でもより便利な TFS へ乗り換える手段が用意されているのは嬉しいことです。

実際に移行するには移行のための準備、移行するためのデータの用意、実際の移行とやらなければいけないステップが多いため、少したいへんな作業にはなりますが、既に VSS を利用しているから利用できないと決めつけてしまう前にぜひ一度検討してみてください。

本稿執筆時点ではまだ、TFS 2010 用の VSS からの移行方法についてのドキュメントが MSDN ライブラリには公開されていないようですが、TFS 2008 のときのものがソース管理および障害/変更追跡の Team Foundation への移行 [http://msdn.microsoft.com/ja-jp/library/ms253186.aspx] という名前で公開されているので、参考にしてください。

たとえば VSS で蓄えてきた履歴情報を含むすべてのデータを移行することができ、アプリケーションのライフサイクルとして欠かせない情報をそのまま引き継げるというのは非常に重要なことです。ただし、仮に最新バージョン

のファイル一式があればよいというのであれば、VSS から最新バージョンのすべてのファイルを取得し、VSS による管理から外したうえで、改めて TFS に新規に登録するだけでも TFS を使い始めることはできます。過去の履歴データをすべて破棄することになってしまうので危険ではありますが、場合によっては選択肢の 1 つとして覚えておくと便利かもしれません。

■まとめ

TFS によるソース管理の実践方法、その環境を作る方法をステップ バイ ステップで確認してきましたが、内容を理解することはできたでしょうか。ソース管理にしてみてもさまざまな機能が用意されているため、いきなりすべての機能を使っていこうというのはなかなか骨の折れる作業です。まずは基本的なチェックイン、チェックアウトがしっかり行えるように実践していくことが大切です。ソース管理の実践ができるようになった後で、それ以外の機能についてはその後に少しずつ徐々に導入していけるように計画していけば十分でしょう。

今回、TFS の導入では、最も基本的な機能だけを利用するように設定しましたが、Windows Server に導入することで、TFS が持つすべての機能を利用できるようにもなります。これは、標準構成またはアドバンスド構成と呼ばれるもので、TFS が蓄積したデータのレポートなどさらにいろいろなことができるようになります。Windows Server に入れるため、そもそも別途 OS が必要であったり、SQL Server も Standard Edition 以上が必要になったりと越えなければいけないハードルも相応に高くはなりますが、ソース管理を実践できるようになった後には検討するに値するだけの価値を提供してくれます。

まだ、ソース管理ツールを導入していないあなた、この機会にぜひソース管理ツール、できれば TFS の導入を検討してください。既に VSS を利用されているあなた、さらにさまざまなメリットを提供してくれる TFS の導入をぜひ検討してください。導入にあたり、本稿がその一助になれば幸いです。

■著者紹介

WINGSプロジェクト りばてぃ

普段は SIer として日々.NET 系の開発プロジェクトに励んでいますが、VSTS や TFS をこよなく愛するがゆえに、たまーに目的と手段が入れ替わっていることも。マイクロソフト系のセミナーやまれにコミュニティの活動にも出没することもありますので、一緒に VSTS の世界にディープダイブしましょう。

■監修者紹介

山田祥寛

千葉県鎌ヶ谷市在住のフリーライター。Microsoft MVP for ASP/ASP.NET。執筆コミュニティ「WINGS プロジェクト」代表。書籍執筆を中心に、雑誌／サイト記事、取材、講演までを手がける毎日。http://www.wings.msn.to/


Microsoft®