NPO 法人　ソフトウェアテスト技術振興協会

テストツールまるわかりガイド(入門編)

Version 1.0.0

ASTER テストツール WG　著

2012/7/2

目次

はじめに

NPO 法人 ASTER（Association for Software Test EngineeRing：ソフトウェアテスト技術振興協会）では、日本のソフトウェアテスト技術向上のためにテスト技術者の教育、新しいテスト技術の研究などに取り組んでいます。その取り組みの一環として、テストツールを開発現場へ普及推進するため、2010 年 11 月に ASTER テストツール WG を設立し、活動してきました。

ASTER テストツール WG には、テストツールを開発、販売しているベンダー企業や、テストツールを活用しているソフトウェア開発企業の方々に参加いただいています。テストツール WG では、ベンダー間の垣根を取り払い、日本の開発現場で効果的にテストツールを活用できるようにするにはどうすべきか議論を重ねてきました。議論をとおして、テストツールを使いこなせる開発現場こそが品質のよいソフトウェアを世の中に送り出すことができると確信し、その役に立つことこそがテストツールの価値であることを共有するという結論に至りました。

議論の結果、テストツール WG では以下の 3 つの方向性で WG の活動を進めていくことにしました。

1. 日本の開発現場へのテストツール普及推進（なぜツールが必要なのかを理解していただく）
2. マネジメント層へテストツールの価値をアピール
3. 日本の開発現場へテストツール活用方法の最適解を提案

そして、上記方向性の 1 番目に関わる活動として、開発現場でテストツールを使いこなすために必要なソフトウェアテストとテストツールの基礎知識、また、テストツールを使う時のポイントをまとめ、公開することとしました。

本書の狙いと対象読者

本書の狙いはソフトウェアテスト技術の向上と、それによる日本のソフトウェア品質の向上です。それを実現するために、テストツールと言う「道具」がソフトウェアテストの現場でどのような作業を支援するか解説しています。そのため、本書ではテストツールの詳細な機能紹介はしていません。

対象読者は、ソフトウェア開発の現場で実際にテストツールを使うことになる技術者の方々を想定しています。現場経験が 1 年～2 年目の技術者の方でも読み進められるようにしていますので、自らの作業の効率化をどう進めればよいか考えるきっかけにしていただけます。また、熟練の技術者の方やプロジェクト管理に関わる方は、現場の改善や若い技術者への指導の一助として活用していただくこともできるでしょう。

テストツールの開発、販売に関わる方には、各社ツールの現場での活用方法/運用方法の提案や導入支援の一助として本書を活用していただくことを想定しています。

本書の構成

本書は全部で六章から構成されています。

第一章ではテストツールがなぜ開発現場で必要なのかを解説しています。

第二章では、開発現場のテスト作業の状況をチェックする「開発プロジェクト健康診断チェックリスト」を用意しました。このチェックリストを活用して、どのような状況なのか？またどのようなツールが効果的かセルフチェックが行えます。

第三章では、ソフトウェア開発プロジェクトの中でソフトウェアテストを行う際に必要となる基本的な作業内容（つまり、テストプロセス）を紹介しています。テストツールはテスト作業を効率化するツールであるため、テストにはどのような作業があるのかを再確認していただくことが狙いです。

第四章では、テストツール体系として、テストツールの種類を一通り解説しています。第三章のソフトウェアテストの作業毎にツールの種類を整理しているので、どのようなツールがテスト作業を支援するのか理解が容易になるでしょう。また、第四章にはツールの種類毎に「使用上の注意」も併記しています。使い方を間違えると発生しうる問題が分かるので一読をお勧めします。

第五章、第六章は、現在入手可能なテストツールの紹介です。第三章で解説したテストの作業を支援するツールはどのようなものがあるのかを確認していただけます。本書では１ツールに対して１ページの簡単な解説になっていますが、各ツールについて紹介されている URL を併記していますので、そちらを参照していただければツールの具体的な機能について理解ができるようになっています。

なお、第五章、第六章のツール紹介では、現在入手可能なすべてのツールが網羅されているわけではなく、ASTER テストツール WG に参加しているベンダー企業にて開発、販売しているツール、もしくはソフトウェア開発企業のメンバーが実際に利用したことのあるオープンソースのツールを掲載しています。ツールの紹介は今後本書を改版していく際に見直し、拡充を進めていく予定です。（年１回～2 回の頻度で見直しをする予定にしています）

上記で概説した各章の関係を図に表すと下記のようになります。

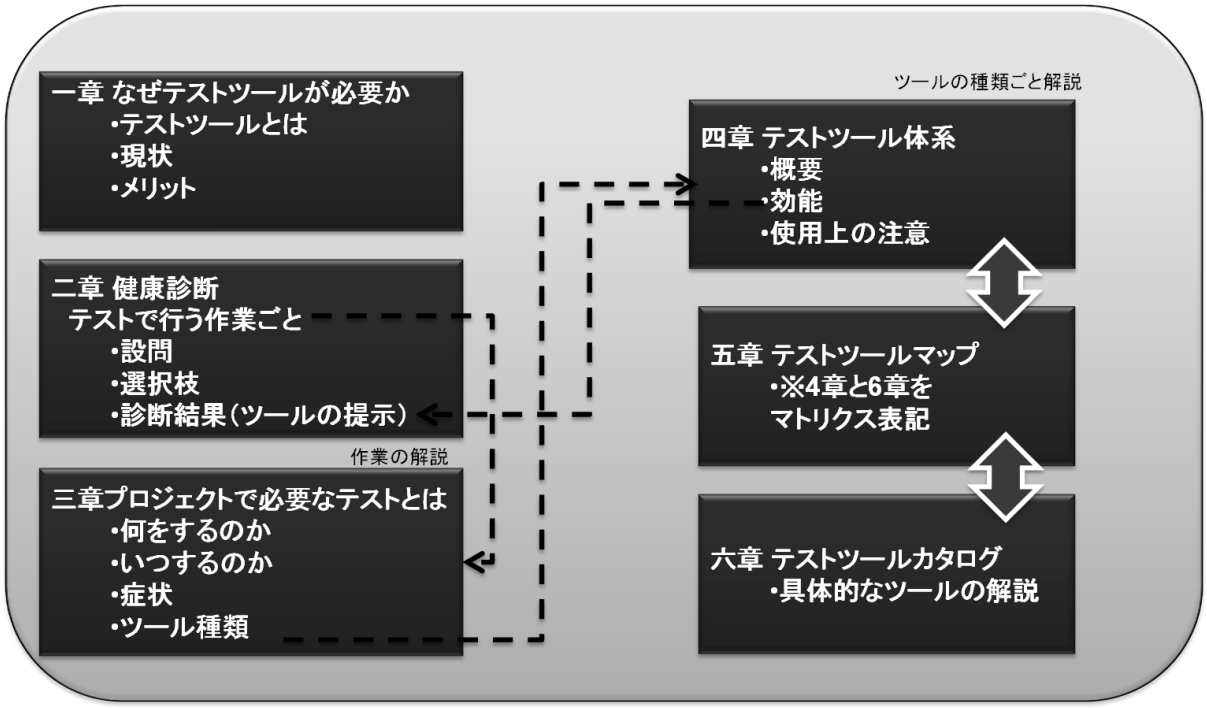


図 0-1 本書の構成

第二章の診断結果で出てきたテストツールの解説を第三章と第四章で行い、第四章で紹介したテストツールの種類に該当するプロダクトとして入手できるものを第五章、第六章で確認できるようになっています。

破線の矢印は、第二章の「テストで行う作業」を第三章で解説、第三章の「ツールの種類」を第四章で解説、そして第四章の「ツールの効能」が第二章の診断結果につながるようになっていることを意味しています。また、第五章に向けたふたつの太い矢印は、第四章と第六章との関係を示しています。

本書で採用したテストツールの分類

本書では、テストツールをテスト作業（つまり、テストプロセス）毎に分類しています。テストツールを現場に適用し有効活用してもらうためには、ツールがどの作業を支援するものなのかより明確にすべきであると考えたからです。そのため、ソフトウェアテストの標準規格や他の市販のテストツールに関する文献とは分類方法が異なることもあります。

本書のテストツールの分類を図にすると下記のようになります。

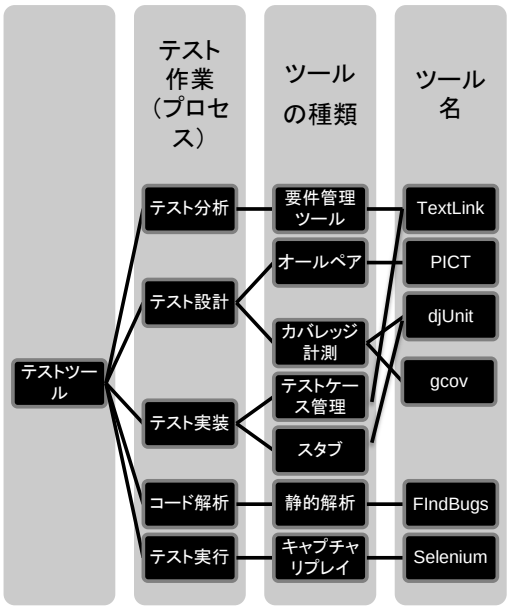


図 0-2 本書で採用したテストツールの分類

テスト作業（プロセス）は第三章、テストツールの種類は第四章で解説し、ツール名（テストツール）は、第六章にて紹介しています。

図を見ると分かるとおり、ひとつのツール種類に複数のテストツールが実在する場合だけでなく、ひとつの実在するテストツールが複数のツール種類を包含する場合もあります。このようにツール種類と、ツール名は、n 対 n の関係になり、単純な階層関係では表現できません。本書では、その関係を表現するために第五章で「テストツールマップ」を作り、両方の関係をマトリクスで表現しました。

本書にて触れなかった内容について

本書は、テストツールを有効活用していただくことを主眼にしています。そのため、本来のテスト全般、品質保証全般では必ず触れなければいけない内容について意図的に記載していないものがあります。例えば以下の作業が該当します。

- ツールではなく人が考えなければいけない作業（例えば、テスト戦略、テスト方針、テスト計画など）
- テスト以外の品質保証活動について（例えば、レビュー、モデル検査、形式検証、監査など）

本書の用語定義

本書で使われている用語定義は、ソフトウェアテストを体系的に学ぶための国際資格 ISTQB(International software testing qualifications board)の用語集を日本語訳した JSTQB 用語集（http://jstqb.jp/syllabus.html）をベースにしています。そのため、各組織にて一般に使われている用語と異なる場合があります。例えば以下の用語が該当します。

- 障害、バグ、不具合、イシュー、故障、欠陥などと呼ばれる用語は、テストで発見した段階を「インシデント」、ソフトウェアの問題として特定できた段階を「欠陥」に統一
- 単体テスト、コンポーネントテスト、モジュールテスト、単体試験と呼ばれる用語は「ユニットテスト」に統一

テストに関わる用語で不明なものがあれば、JSTQB 用語集をあわせて確認いただけるようお願いします。

また、開発やテストでの「行い」は、プロセス、アクティビティ、タスクといった概念を整理して使うことも多くありますが、本書では日本語での読みやすさを重視し、「作業」、「プロセス（複数の作業をまとめたもの）」と言う用語にて記述しました。

本書の活用方法

「まずは己を知る」ということで第二章の開発プロジェクト健康診断から始めていただくのもよいですし、テストツール活用の勉強会の資料として活用していただくのもよいかもしれません。またテストツールを購入する際のカタログとして利用していただくのも活用方法のひとつです。つまり、読者の皆さんの使いやすいように本書を活用していただいてかまいません。

どのように活用していただくとしても、本書で「ツールを使う以前に何が理解できていなければいけないか」を理解していただき、テストツールが開発現場にとって真に役立つものとなることが私たちの願いです。本書がその役に立つことを切に願っています。

ASTER テストツール WG 一同

第一章　なぜテストツールが必要なのか？

テストツールとは？

ソフトウェア開発における「テストツール」と聞いて皆さんは何を思い浮かべるでしょうか？プログラマの方であれば、身近なツールとしては統合開発環境(Eclipse など)に付属する「デバッガ」を思い浮かべる方が多いでしょう。プロジェクトマネージャやチームリーダといった管理業務に関わっている方は（本来はテストツールではないにも関わらず）表計算ソフトを思い浮かべる方が多いかもしれません。実は、これら以外にも、世の中には今すぐにでも利用できる有用な「テストツール」が数多く存在します。例えば、テスト管理系ではソフトウェア要件の管理から、テストケース管理、インシデント管理のためのツールがありますし、一方、テスト実行系では、テスト自動実行ツール、性能テストツール、静的解析ツール、動的解析ツールなどが存在します。

これらのツールは、ソフトウェア開発を効率化し、開発したソフトウェアの品質を高めることによりビジネスに貢献することを目的としています。今日では、あらゆる産業でソフトウェアによるイノベーションがビジネスにおける成功の鍵となっていること、加えて、経済のグローバル化による競争機会の増加や、円高による国内開発のコスト増を鑑みると、テストツールを用いたソフトウェア開発の効率化は日本のソフトウェア開発現場にとって必須であると言っても過言ではありません。しかしながら、日本では、テストツールが有効に活用されていないという現状があります。

テストツールが使われていない現状

日経システムズの調査[1]によると、ソフトウェア開発の現場でテストツールを使っていない理由のうち、第 1 位は「導入コストが高い」、第 2 位は「手作業で行ったほうが早い」、第 3 位は「どんなツールがあるのか知らない」となっています。第 1 位の「導入コストが高い」に関しては、実際のプロジェクト予算とソフトウェア価格のギャップがあることは否めませんが、一方で、テストツールの存在の認識不足や、投資対効果がユーザ側では見えにくい、または理解されておらず、結果としてテストツールの利用が予算化されていないため、このような現状に至っているとも考えられます。第 2 位の「手作業で行ったほうが早い」に関しては、実際にツールの使い勝手が悪いということも考えられますが、テストツールを適材適所で使っていない、もしくは第 3 位の「どんなツールがあるのか知らない」と同様に、テストツールの現場に対する認知度が不足していることも理由として考えられるでしょう。

上記調査結果をまとめると、日本のソフトウェア開発の現場でテストツールが利用されていない理由のひとつとしては、テストツールの存在、および、その効果に対するユーザの理解が不足していることが大きいと言えます。これはソフトウェアベンダーのみならず日本のソフトウェア業界における大きな問題です。ソフトウェアベンダーは、ユーザに対して、テストツールの存在およびその価値に対する理解を促進させると共に、ソフトウェア開発をより効率的なものに変革できるようサポートしていく必要があります。

テストツールを使うメリット

それでは、テストツールを用いると具体的にはどのようなメリットがあるのでしょうか？以下では、テストツールがソフトウェア開発をどのように効率化するのかをコード解析における静的解析ツールの例を用いて紹介します。

[1] 日経システムズの調査結果　http://itpro.nikkeibp.co.jp/article/COLUMN/20110512/360288/

静的解析ツールとは、簡単に言うと、ソースコードを解析し、ソースコード上に存在する問題を自動的に検出するものです（図 1-1）。ここでいう「静的」とはソフトウェアを実行することなくソースコードを解析することにより、問題を指摘することを意味しています。静的解析ツールは、文法やコーディングルールをチェックする機能に加え、メモリーリークや配列のオーバーラン、デッドロックといった実行時に複数の関数やクラスをまたがる複雑な条件下で発生する欠陥を発見する機能を持ったものもあります。

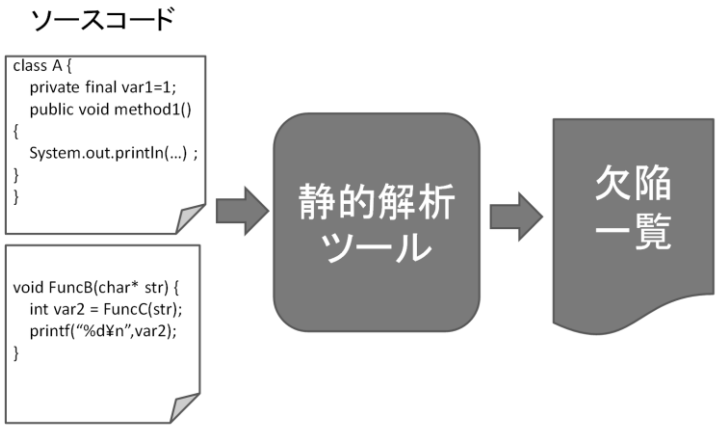


図 1-1 静的解析の仕組み

このような静的解析ツールが有用である大きな理由はふたつあります。ひとつ目は、ソースコード品質の向上および確保です。静的解析ツールは 100%のパスカバレッジによる欠陥の発見を特徴としており、目視によるコードレビューや動的テストでは見逃される可能性が高い欠陥を発見することができます。これによりコードレビューにおける欠陥の見逃しやテストケース漏れなどによるテスト結果のばらつきを抑えることができます。ふたつ目は、ソフトウェア開発（特に欠陥除去）の効率化です。ますます大規模・複雑化するソースコードを人手によりレビューやテスト実行をすることは工数の観点から困難であり、何らかの自動化技術は必須であると言えます。静的解析ツールは、ソフトウェアを実行することなく自動的に欠陥を発見できること、および、コーディング段階から使用することにより、欠陥の早期発見・修正が可能になるため、欠陥修正コストを減少させることができます。

以上の例に見られるように、テストツールの活用は、これまで人間が行っていた機械的な作業を自動化することで工数を削減し、人間はより高度な判断を必要とする作業に注力することが可能になると言えます。特に欧米ではアジャイル開発など、新しいソフトウェアエンジニアリングによる開発手法の浸透に伴い、このようなテストツールの活用が進んでいます。一方、日本の状況を鑑みると、前述の通り、より少ない人数・より短い開発期間で、より複雑なソフトウェアの開発が求められているにも関わらず、多くのソフトウェア開発の現場では効率化が進んでいないというのが現状ではないでしょうか。

これからの時代に必要とされるエンジニアとなるために

ところで、ダニエル・ピンク著の「ハイ・コンセプト」[2]という書籍をご存知でしょうか。この書籍の一節で、経済がグローバル化し、多くの仕事が発展国へとオフショアリングされていく中で、先進国で働く人々に対して、今の仕事をこのまま続けてよいか判断するために、以下に示す3つのチェックポイントが紹介されています。

[2] 「ハイ・コンセプト「新しいこと」を考え出す人の時代」 ダニエル・ピンク著 大前研一訳 三笠書房 2006

1. *他の国なら、これをもっとやすくやれるだろうか？*
2. *コンピュータなら、これをもっとうまく、早くやれるだろうか？*
3. *自分が提供しているものは、この豊かな時代の中でも需要があるだろうか？*

皆さんはこの3つのチェックポイントを読んで何を思われたでしょうか？日本のソフトウェア開発現場では、残念ながら上記チェックポイントに当てはまるケースが多いのではないかと感じています。言い換えると、日本のソフトウェアエンジニア・テストエンジニアがグローバル市場で生き残っていくためには、テスト分野で言えば、テスト技法を学び、自動化技術を始めとするテストツールを活用し、より高いレベルの仕事を遂行するようになることが必須であると言えるでしょう。

次章からは、日本のソフトウェア開発の現場を効率化するために、ソフトウェア開発プロセス全体を見据えた上で、特にソフトウェアのテストにおいて、どのようなアクションが必要であり、どのようなテストツールが適用可能であるかを紹介していきます。

第二章　あなたのプロジェクト、健康診断

あなたのプロジェクトは、テストツールを有効に活用し、日々の開発を効率的に行えている「健康なプロジェクト」でしょうか？以下の診断表で、セルフチェックをしてみてください。

同様の質問は ASTER の Web 上からも実施できる予定です。(http://www.aster.or.jp/business.html#toolwg)

診断表

①　テスト分析

No.	質問	選択
1	**要件に応じたテストケースを作成していますか？** *A: テスト対象の要件を理解してるよ。分からなかったらちゃんと調べるよ。* *B: 言われたことは理解してます。* *C: テスト対象の要件の一覧を見たことがないです。*	
2	**テストケースとテスト対象の要件の関連を正しく把握していますか？** *A: テスト対象の要件とテストケースの関連付けを行ってるよ。* *B: 部分的には行っているはず・・・。* *C: 頭では分かってます。*	
3	**要件変更があった場合、テストケースへの影響を把握できますか？** *A: 要件変更により影響があるテストケースを瞬時に識別できる。* *B: 要件変更の際に、これまでのテスト結果から影響範囲を想定してる。* *C: 要件変更があったことを知る術がありません。*	

② テスト設計

No.	質問	点数
4	**テストを実行する工数と、テストケースを作る工数ではどちらが工数がかかりますか？** *A: テスト分析とテスト設計次第である。テスト分析の結果によっては、テストケースの数を意図的に少なくするように設計し、テスト実行工数がかからないようにしている。* *B: テスト対象による。例えば、組合せパターンが多いテストは実行のほうが工数がかかる。* *C: テスト実行する工数のほうが断然かかる*	
5	**テスト技法（原因結果グラフ、直交表もしくはオールペア、状態遷移テストなど）を駆使してテスト設計していますか？** *A: 駆使してると自信を持って言える。* *B: 技法は現実的に使うのは難しい。* *C: 利用していない、初耳。*	
6	**テスト設計の結果として何をレビューしていますか？** *A: テスト設計した結果としてできるモデルや表を常に（仕様変更にあわせて毎回）レビューしている。* *B: テスト設計した結果としてできるモデルや表を一回はレビューしている。* *C: テストケースをレビューしている。* *D: テスト設計の結果をレビューする必要はない。*	

③ テスト実装

No.	質問	点数
7	**テスト環境やテストデータを作成する担当者がいますか？** *A:メンバー全員ができるよう仕掛けができているため、テスト環境やテストデータの作成はメンバー自身でやっていける。* *B: 担当者はいる。しかし、彼がいなくても問題なく対応できるよう標準化、手順化ができているので他の人でも作業できる。* *C: 担当者がいないのでいつも混乱している。* *D: 担当者はいる、彼の仕事っぷりは神業としか言えない。他の人にはまねできない。*	
8	**テスト設計したテストケースをどのビルドで誰にテストしてもらうか事前に割り振っていますか？** *A: テストケースをどう割り振ったかは全メンバーが分かるようにしているし、見直しの際もどこを見直しているかが分かるようにしているので、問題ない。* *B: 開発スケジュールの遅延や欠陥によって見直しがしょっちゅう入るので正直言って大変。助けてほしい。* *C: そういうことは、その日その日で考えているので問題ない。*	
9	**一連のテストケースを実行するための詳細なテスト手順を実行前に作っていますか？** *A: ケースバイケース。事前に詳細な手順が必要な時は手順を作るし、何度も行うものは事前に自動化できるよう自動スクリプトを書いている。* *B: 開発スケジュールの遅延や欠陥によって見直しがしょっちゅう入るので、テスト担当者に書いてもらい、実行結果として残してもらっている。* *C:手順は必ず作るもの。なので、テスト設計の一環でやっている。*	

④ コード解析

No.	質問	点数
10	**コードレビューを漏れなく効率的に行えていますか？** *A: 見るべきポイントを事前に定義したり、ツールを活用することにより、漏れなく効率的に行えている。* *B: 余裕がある時はできているが、忙しくなってくるとできないことがあり、効率化が必要であると感じている。* *C: 忙しくて殆どできないか、やっているが形骸化しており、やるだけ無駄だと思っている。*	
11	**デッドロックやメモリリーク、システムクラッシュなどの欠陥の発見のためどのようにテストしていますか？** *A: コーディングルールの遵守やツールの活用によりコーディング・ユニットテスト段階で可能な限りテストしている。* *B: ロングランテストやフリーテストなど、テスト段階で探索的に発見。* *C: 特定のテストは行っていない。*	
12	**ソースコードレベルでどのくらいシステムがテストされるべきか（テストカバレッジ）目標を定義し、実際どれくらいテスト実行されたか確認していますか？** *A: ツールを利用して確認している。* *B: やっているが開発担当者まかせ。* *C: そんなことやっていない。*	

⑤ テスト実行

No.	質問	点数
13	**機能テスト、構造テスト、非機能テスト、回帰テストを行っていますか？** *A: 内容を理解しており、テスト設計、テスト実装に沿って実行している。* *B: どれかはやるようにしている。* *C: テストはしているけど、どれかはわからない。*	
14	**毎回、同じオペレーションでテストをしていて嫌になっていませんか？** *A: 自動でやってるから平気だよ。* *B: 苦にならないよ。* *C: はい、だから手を抜いてるよ。*	
15	**製品リリース前になると大人数でシステムを使うテストをしていませんか？** *A: 仮想ユーザを発生させるツールを使ってる。* *B: 業務命令で多くの社員でやってる。* *C: やらずにリリースしちゃいます。*	

⑥ テスト結果管理

No.	質問	点数
16	**プロジェクト毎のテスト予実管理をしていますか？** *A: もちろん。それに、次回のテスト時には今回の実績データを使うよ。* *B: 予定と実績は管理しているけど実績データを活用できてないかも・・・。* *C: 毎回忙しいのでそれどころでないよ。*	
17	**テスト条件、テストシナリオ、進捗、テスト結果がそれぞれ紐づけられ状況を把握できることのメリットを理解していますか？** *A: 十分理解しており、紐づけもしているよ。* *B: 課題だとは思うが何もしていない。* *C: 理解できない。*	
18	**テスト進捗やテスト結果のメトリクスはすぐに見られる？** *A: もちろん、すぐに見られます！！* *B: 表計算ソフトが得意な社員のマクロ作成まで待ってください！* *C: メトリクスってなんですか？*	

⑦ テストウェア管理

No.	質問	点数
19	**テストスクリプトなどの管理は必要であると感じていますか？** *A: はい、もちろん実践しています。* *B: できてはいないがそう感じている。* *C: 必要ないでしょ。*	
20	**テストスクリプトやソフトウェアは履歴管理され、ビルドされたプログラムに対応づけられている** *A: はい、関連付けと管理を行っています。* *B: 履歴管理だけ行っている。* *C: 特に何もしていない。*	
21	**テストに使用するソフトウェアやスクリプトは再利用可能ですか？** *A: 管理ツールを利用している。* *B: プロジェクト毎にサーバのフォルダに保存している。* *C: そんなことする必要あるんですか？*	

⑧ インシデント管理

No.	質問	点数
22	**テスト中に欠陥が見つかったらどのようなアクションをとっていますか？** *A: インシデント管理を行うシステムに入力している。* *B: Word/Excel に記入している。* *C: 開発担当者に都度連絡。*	
23	**インシデントを管理した内容を活用できていますか？** *A: レポート内容が開発担当者に連絡が行ったり、分析結果によりリリース判断としている。* *B: これからの課題だと思っている。* *C: 何を活用すればよいか分からない。*	
24	**インシデントの数や改修された欠陥を管理し、グラフなどに視覚化して関係者間で共有していますか？** *A: やってますよ。そのおかげで状況を理解して仕事ができるようになりました。* *B: これからの課題だと思っている。* *C: そんなデータ、マネージャだけ見ればいいでしょう。*	

配点と診断結果

① テスト分析

No.	質問	点数
1	A を選択したあなたの配点は 4 点 B を選択したあなたの配点は 2 点 C を選択したあなたの配点は 0 点	
2	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
3	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
	診断結果説明 0-3: まずは要件管理プロセスを定着させましょう 4-7: 基本はできていますので、要件管理ツールを使い、もう一段上を目指してみてはいかがでしょう！？ 8-10: すばらしい！　この調子でいきましょう！	合計　　点

② テスト設計

No.	質問	点数
4	A を選択したあなたの配点は 4 点 B を選択したあなたの配点は 2 点 C を選択したあなたの配点は 0 点	
5	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
6	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点 D を選択したあなたの配点は 0 点	
	診断結果説明 0-3: テスト設計について学ぶところから始めましょう。 4-7: テスト設計をちゃんとやっているが、あまりにテスト設計に工数がかかりすぎていませんか。テスト設計ツールを活用するとよいかもしれません。 8-10: すばらしい！ この調子でいきましょう！	合計 点

③ テスト実装

No.	質問	点数
7	A を選択したあなたの配点は 4 点 B を選択したあなたの配点は 2 点 C を選択したあなたの配点は 0 点 D を選択したあなたの配点は 0 点	
8	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
9	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
	診断結果説明 0-3: テスト実装という作業が正しく認識できていないかもしれません。きちんとやるようにしましょう。 4-7: テスト実装は楽チンだと思っている節があります。そのため、現場は大変な思いをしてやり遂げているかもしれません。テスト実装関連ツールを活用するとよいかもしれません。 8-10: すばらしい！ この調子でいきましょう！	合計 点

④ コード解析

No.	質問	点数
10	A を選択したあなたの配点は 4 点 B を選択したあなたの配点は 2 点 C を選択したあなたの配点は 0 点	
11	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
12	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
	コード解析の診断結果説明 0-3: いつまで泥臭いことやるつもり？テストはコーディング段階のエラーを発見するものではありません！ 4-7: 努力しているみたいだけど、静的解析ツールを使ってコード解析したらもっといいよ！ 8-10: すばらしい！ この調子でいきましょう！	合計 点

⑤ テスト実行

No.	質問	点数
13	A を選択したあなたの配点は 4 点 B を選択したあなたの配点は 2 点 C を選択したあなたの配点は 0 点	
14	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
15	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
	テスト実行の診断結果説明 0-3: 品質問題が多いか、テスト工数がかかりすぎていませんか？テストプロセスの整理から着手してみましょう。 4-7: 惜しいです。あと一歩で品質問題を減らせる可能性があります。キャプチャ/リプレイツールや性能テストツールを使わないとできないこともありますよ。 8-10: すばらしい！　この調子でいきましょう！	合計　　点

⑥ テスト結果管理

No.	質問	点数
16	A を選択したあなたの配点は 4 点 B を選択したあなたの配点は 2 点 C を選択したあなたの配点は 0 点	
17	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
18	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
	テスト管理の診断結果説明 0-3: テスト結果データを蓄積するところから始めましょう。ツールはその後です。 4-7: 基本はできています。蓄積したデータを有効活用できるテスト管理ツールを使うことで一歩前進できるでしょう！ 8-10: すばらしい！　この調子でいきましょう！	合計　　点

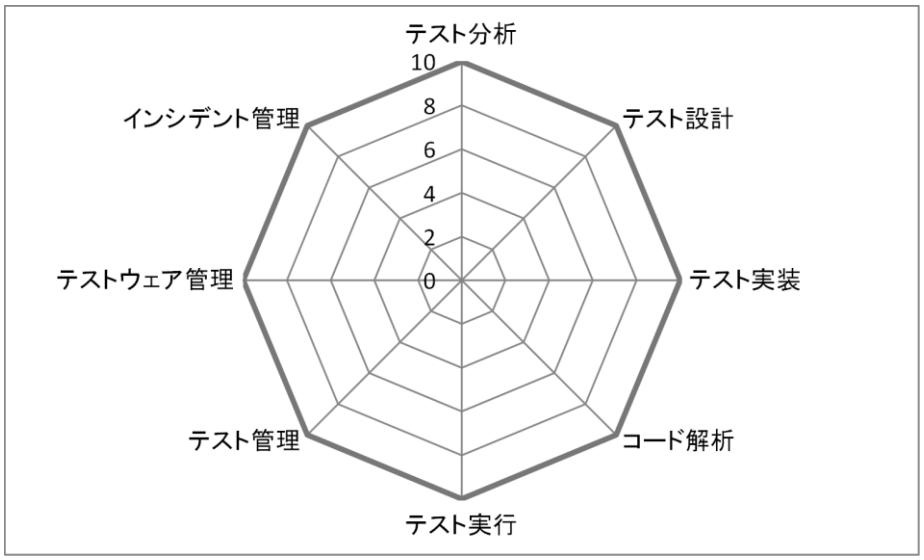
⑦ テストウェア管理

No.	質問	点数
19	A を選択したあなたの配点は 4 点 B を選択したあなたの配点は 2 点 C を選択したあなたの配点は 0 点	
20	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
21	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
	テストウェア管理の診断結果説明 0-3: まずはテストウェア管理プロセスを定着させましょう 4-7: 基本はできていますので、構成管理ツールを使い、もう一段上を目指してみてはいかがでしょう！？ 8-10: すばらしい！　この調子でいきましょう！	合計　　点

⑧ インシデント管理

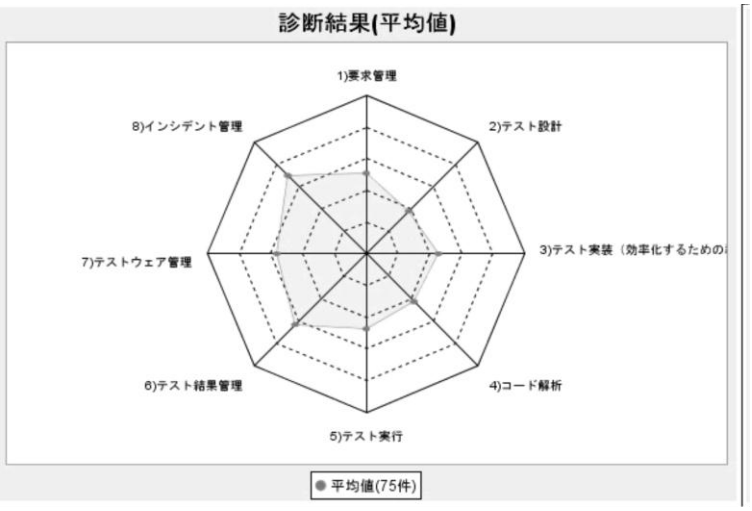
No.	質問	点数
22	A を選択したあなたの配点は 4 点 B を選択したあなたの配点は 2 点 C を選択したあなたの配点は 0 点	
23	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
24	A を選択したあなたの配点は 3 点 B を選択したあなたの配点は 1 点 C を選択したあなたの配点は 0 点	
	インシデント管理の診断結果説明 0-3: コミュニケーションロスの発生や問題点が改善されない危惧があります。インシデント情報を記録・共有するところから始めましょう。 4-7: 品質分析にかかる工数を削減できる可能性があります。インシデント管理ツールを使ってみてはいかがでしょうか。 8-10: すばらしい！ この調子でいきましょう！	合計　　点

各項目の合計点を下記のレーダーチャートに書き入れて傾向を見てみましょう。



この 8 項目は、テストを行う際に必要となる作業を表しています。レーダーチャートの傾向から、皆さんのプロジェクトではどのテスト作業に課題があるかが読み取れるでしょう。

以下のレーダーチャートは、JaSST'12 Tokyo「テストツールの処方箋」セッションにて実際に会場にてアンケートを実施した結果の平均になります。結果からは、「インシデント管理」は多くの方が上手く実施できているが、それに反し、「テスト設計」や「コード解析」があまり上手く実施できていないことが読み取れます。



本章のプロジェクト健康診断は ASTER の Web サイトでもご利用いただけるよう準備しております。オンラインで実施すると、設問に答えるだけで自動的に採点を行い、同時にレーダーチャートも表示してくれます（個人情報の入力などはございません）。また、回答いただいた結果は履歴として蓄積されます。その結果は今後の日本における開発環境の実態や傾向の分析のために活用し、その情報は随時公開していく予定です。よって、出来る限り多くの方に回答いただけると幸いです。

次節以降で、プロジェクトで必要となるテスト作業の解説を行います。また、各作業において活用可能なツール群のマッピングを行っておりますので、ご活用ください。

第三章　プロジェクトで必要なテストとは

はじめに

そもそも、ソフトウェア開発プロジェクトの中で、テストはどのように行われているでしょうか。

「テストとは何か？」という質問をすると、「開発した成果物が想定通りに動いていることを確認する作業」だと答える人もいますし、「ユーザに使ってもらう前に欠陥を見つける作業」と答える人もいます。他にも「単なる納品する前の決まりごと」であり、納品物リストにテスト結果があるからやっているだけだと言う人もいるでしょう。多分、テストを「単なる納品する前の決まりごと」であると考えていれば、テストにツールはあまり必要だと思われないかもしれません。

ツールとは、所詮作業を手助けする「道具」にすぎません。テストツールを使うことも重要ですが、テストとは何のために行っているのか、そのためにどのような作業が必要なのかをプロジェクトの中で共通の理解にする必要があります。テストのために必要な作業が分かると、その作業を便利にするツールの必要性も認識できるでしょう。

テストと言うと、開発されたソースコード、もしくはコンパイルしたバイナリをとりあえず動かしてみることや、時間の空いている人に「テストしておいて！」とお願いすることだけがテストである、つまり「テストを実行すること」だけがテストだと思うことが多いかもしれません。ちなみにそのようなプロジェクトの計画書に書かれるガントチャートには、テストはコーディングの後からしか線が引かれません。（図 3-1 参照）

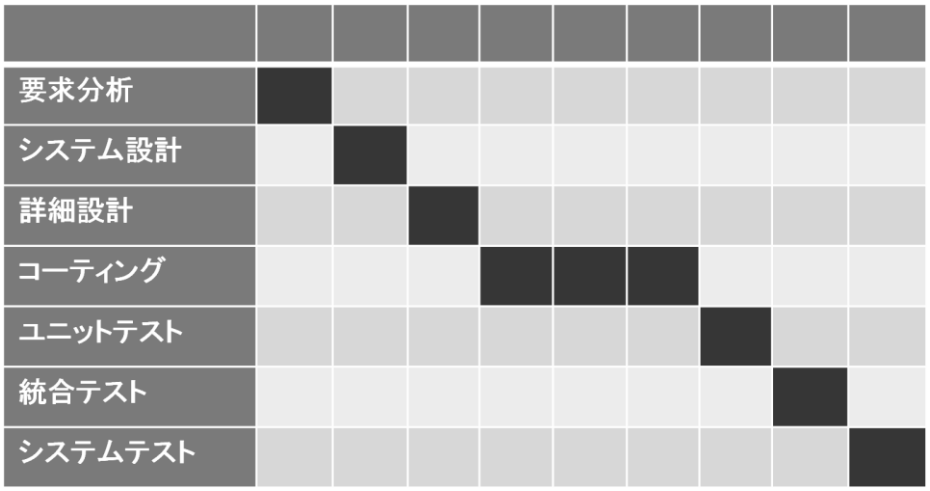


図 3-1　テスト実行だけがテストだと考えるプロジェクトのガントチャート

テスト実行だけがテストだと考えているプロジェクトでは、テスト実行を支援するツール、つまりテスト自動実行ツール以外のテストツールは必要性も認識できませんし、それらのテストツールを導入しても何に使えばよいかわからないでしょう。

テストは「実行」するだけでは上手くいきません。テストが、「開発した成果物が想定通りに動いていることを確認する作業」だとすれば、「想定通り」が何かわからないまま動作させてもそれでよいのかがわかりません。「ユーザに使ってもらう前に欠陥を見つける作業」だとすれば、欠陥が見つかるように動かしてみないといけませんし、見つかった欠陥を直さなければなりません。

ISTQB（ソフトウェアテストを体系的に学ぶための国際資格）では、テストについて以下のように記されています。

テストの活動は、テスト実行の前後にも存在する。例えば、計画、コントロール、テスト条件の選択、テストケースの設計と実行、実行結果のチェック、テスト完了基準の検証、テストプロセスやテスト対象システムに関する報告、テストのまとめや終了作業(テストフェーズが完了した後)がある。テストにはドキュメント(ソースコードを含む)レビューや、静的解析を実施することも含む。(ISTQB FL シラバスより)

本書では、テストに必要な作業として、以下の 8 つの作業を定義します。

- テスト分析
- テスト設計
- テスト実装
- コード解析
- テスト実行
- テスト結果管理（モニタリングとコントロール、報告書作成）
- テストウェア管理
- インシデント管理

上記のテストに必要な作業を、V 字モデル（ソフトウェア開発のライフサイクルとテストの関係を表している）に当てはめると以下の図のようになります。

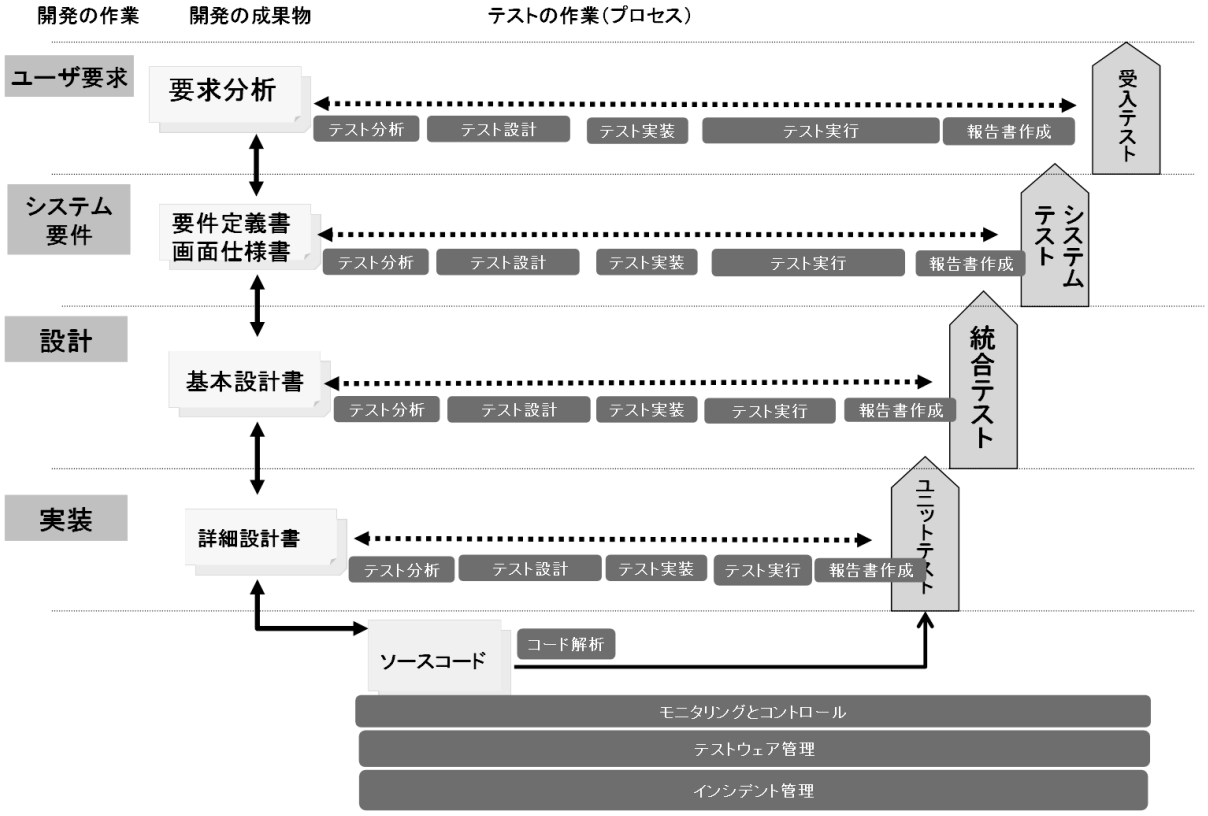


図 3-2 V 字モデルとテストに必要な作業の対比

図 3-2 を見ると分かるとおり、作業によっては、V 字モデルの右側のユニットテスト、統合テスト、システムテスト、受け入れテスト毎に行うものもあります。（これらは総称して「テストレベル」と呼びますので、本書でも以降からは「テストレベル」と呼びます。）

一般的には、どのテストレベルでもテスト分析～報告書作成までのプロセスが必要ですが、テストレベル毎にまるで同じように作業を行い、同じアウトプットを作成しなくてもかまいません。テストレベルによって各作業の大変さが異なるからです。（これらのバランスを考えることはテストを計画する際に行います。）

以降では、本書で定義するテストに必要な 8 つの作業について、その作業では何をするのか、開発プロセスのどの時期で行うべきか、どのようなツールが作業を支援するかを解説します。

なお、本解説は JSTQB シラバスをベースに解説しています。詳細は JSTQB シラバスを参考にしてください。

テスト分析

何をするのか

テスト分析では、テスト対象を理解してテストすべきこと（JSTQB 用語集では、これをテスト条件と呼びます）を明確にしていきます。テスト対象は、テストレベル毎に異なります。図 3 で定義した一般的なテストレベルの例で見ると以下のようになります。

- ユニットテストでは、開発プロジェクトの中で定めた開発の最小単位となるコンポーネント（モジュール、関数、クラス、メソッドなど）をそれぞれが独立した状態で見たものがテスト対象になり、コンポーネントについて書かれたドキュメント（図 3-2 の例では詳細設計書）をベースにテスト分析を行います。ユニットテストでは、テスト対象となる独立したコンポーネントが受け付ける入力のパターンや出力のパターンをテストします。

- 統合テストでは、ユニットテストで確認したコンポーネントを連携させた状態での各コンポーネントの結合がテスト対象になり、コンポーネント間の結合について設計したドキュメント（図 3-2 の例では基本設計書）をベースにテスト分析を行います。コンポーネント間のインターフェイスやソフトウェアが稼動を予定している環境（オペレーティングシステム）上での確認など、ユニットテスト時とは異なりそれぞれの連携部分および連携により実現できることに着目し、テストすべきポイントを明確にしていきます。

- システムテストでは、統合テストまで済んだソフトウェア（つまり、システムとして完成したもの）をテスト対象とします。テスト分析をするためのベースとなるドキュメントは、図 3-2 の例で言えば要件定義書や画面仕様書になります。システムテストは、ユニットテストや統合テストと異なり、テストする範囲が広大になりますので、欠陥の出そうなところやユーザがよく使いそうなところがどこかを考えてテストするポイントを明確にしていきます。

- 受け入れテストでは、システムテストまで済んだソフトウェアがテスト対象になります。そういう意味ではテスト対象はシステムテストと同じです（ただし、ソフトウェアの搭載された環境が本番環境になるなど、場合によって違いがあります）。実際にシステムを利用するユーザが、システムの機能性や操作性などがビジネス要件を満たしているかを確認する場合が多くなるので、もともとのユーザ要求や実際の運用を明記したドキュメントを、テスト分析を行うためのベースにしてテストすべきポイントを明確にします。

テスト対象の開発ドキュメントに書かれている要求と、テストすべきこと（つまり、テスト条件）は、紐付け（追跡可能性の維持）をしておきます。要求とテスト条件の関係は多岐にわたり、また、いろいろなタイミングで明らかになっていくものであり、後で見直すと言ったことも行うからです。テスト分析では、開発ドキュメントには書かれていない情報（例えばユーザがよく使う機能はどれであるとか、どこで欠陥が多く出ているかといったこと）も加味します。それらの情報も、要求と紐付けしていかなければなりません。

開発プロジェクトのどの時期で行うべきか

- 開発時、V 字モデルの左側（開発の成果物）が作られている時、もしくは作られた直後からテスト分析を行い、テスト条件を明確にしていきます。
- 開発時、V 字モデルの左側（開発の成果物）に変更が入るたびにテスト分析内容を見直します。つまり、開発プロセスのすべての時期で見直しは続きます。
- テスト分析（もしくはテスト設計からテスト分析へのフィードバック）の結果、開発の成果物に不備が見つかる場合があります。（記述内容が曖昧であったり、一貫性に欠けていたりなど）その場合は、開発成果物とテストの両方を見直します。これは開発プロセスのどの時期でも行うことができますが、見直しにかかる作業量を考えると、なるべく早い時期が望ましいと言えます。

この作業が上手くいっていないとどんなことが起きるか（症状）

- テスト条件を明確にしない（テスト対象を十分に理解していない）ままテスト設計を行うと、テストの抜け漏れ、もしくは過剰にテストしてしまうといった問題が起きます。
- 要件とテスト条件の関係を紐付けしておかないと、どの要件に対してどの程度テストしているのかがわからなくなります。
- 仕様変更や仕様追加が入った時に、どのテスト条件、テストケースを見直さなければいけないかがわからなくなり、誤ったテストケースが残ってしまいます。

どのような種類のツールを使うのか

- 要件管理ツール

テスト設計

何をするのか

テスト分析にて明確になったテスト条件は、そのままテストできるものもありますが、そのままでは何をテストすればよいかわからない場合も多くあります。例えば、入力の取りうる範囲が広く、全部やるのは大変な場合や、入力および事前状態を組み合わせると、組合せパターンの数が多くなりすぎるような場合です。このような場合は、テスト条件を確認する具体的なパターンを適切な量になるようにしなければいけません。この作業をテスト設計と呼びます。

テスト設計では、テスト条件を漏れなく確認できるテストケースを作っていきます。「漏れなく」と言っても、考えられるすべてのパターンを完全にテストするのは困難です。そのため、テストをする目的から導かれる観点に基づいて、その観点から考えるとどれだけのパターンを作るべきか？ということを考えます。ISTQB では、テストすべき観点は以下のような「テストタイプ」として定義されています。

- 機能のテスト：ソフトウェアの利用者にとって、テスト対象のソフトウェアが機能仕様通り動作するかを確認することが目的のテストであり、ソフトウェアの内部構造は意識せず、その機能の利用者による入力と利用者が得る出力を基にテストケースを作ります。

- ソフトウェアの構造やソフトウェアのアーキテクチャのテスト：ソフトウェアの開発担当者にとって、ソフトウェアの「内部の作り」がソフトウェア設計通り動作するかを確認することが目的のテストであり、内部構造（プログラムであればフローチャート、アーキテクチャ設計であればコーリングシーケンスなど）を網羅するようテストケースを作ります。

- 非機能要件のテスト：機能以外の要件を満たしているかを確認することが目的のテストです。例えば、性能や可用性などを確認するテストであれば、急激な負荷状況の変動や、予期せずシステムがダウンするといった状況を考慮してテストケースを作ります。

- 変更部分のテスト（再テスト、回帰テスト）：欠陥修正や仕様変更、仕様追加によって、すでにテスト済みのソフトウェアに影響がないことを確認することが目的のテストです。ここでは、すでにテスト設計をしてあり、テスト実行まで済んでいるテストを何度も実行することになります。

上記のようにテストタイプ毎に観点が異なるため、テスト設計の結果として作られるテストケースも異なるものになります。また、これらのテストタイプは、どのテストレベルに対しても使うことができます。

テストケースを作るためのテクニック（技法）は数多く提唱されており、その技法を活用することで、一定のルールに基づいてテストケースを作成することができます。技法は、ブラックボックステスト技法（仕様を網羅するテストケースを作る方法）、ホワイトボックステスト技法（内部構造を網羅するテストケースを作る方法）、経験ベースのテスト技法（探索的テストに代表される、ピンポイントにテストケースを作る方法）の 3 種類に分類できます。（技法に関して、詳しくは JSTQB シラバスを参照してください。）

開発プロジェクトのどの時期で行うべきか

- テスト設計は、どのテスト技法を使うかで、開始可能なタイミングが異なります。ブラックボックステストのような仕様を網羅するテスト技法であれば、仕様が分かるようになってからテスト設計を行います。ホワイトボックステストのような内部構造を網羅するテスト技法であれば、内部構造が分かるようになってからテスト設計を行います。経験ベースのテスト技法は、ある程度ソフトウェアを動作させたり、欠陥の傾向が見えてきたりした後でテスト設計ができるようになります。

- テスト分析内容を見直したら、テスト設計内容（テストケース）も見直しが必要です。つまり、開発プロセスのすべての時期で見直しは続きます。

- テスト設計の結果、開発の成果物に不備が見つかる場合があります（記述内容が曖昧であったり、一貫性に欠けているなど）。その場合は、テスト分析へフィードバックし、開発成果物とテストの両方を見直します。これは開発プロセスのどの時期でも行うことができますが、見直しにかかる作業量を考えると、なるべく早い時期が望ましいと言えます。

この作業が上手くいっていないとどんなことが起きるか（症状）

- テスト設計をせずにテストを実行すると、入力のパターンを考えたり、どのような条件の下にこのテストを合格とするのかを考えたりしながらテストを実行することになります。そうなると以下の問題が起きます。
  - テスト実行工数が長くなる。（テスト実行時にテスト設計の作業を同時に行うため）
  - テスト実行者によって結果が異なる。（期待する結果を実行者がその場で決めるため）
  - 本来見なければいけないところを見逃す。（いろいろなことを考えながらの作業となるため）

- テスト設計をしないと、テスト設計結果が明確に残らず、また、誰にもレビューされなくなるので、テストケースが極端に少なかったり、極端に多くなったりしてしまいます。

- テスト設計の際に、テスト技法を誤って適用すると、適切でないテストケースが作られてしまいます。

どのような種類のツールを使うのか

- 原因結果グラフツール
- 組合せテスト支援ツール（直交表、オールペア）
- 状態遷移テストツール
- 動的解析ツール
- カバレッジ計測ツール
- その他のテスト設計支援ツール

テスト実装

何をするのか

テスト分析とテスト設計でテストケースが作られると、「何を」「どのパターンで」テストするかが明確になりますが、テストケースだけあってもそのままテストを実行することはできません。テストを実行できるようにするための一連の作業をテスト実装と呼びます。

テスト実装では以下の作業を行います。

- テスト環境の準備：テスト対象のソフトウェアを動かすためのシステム構築を行います。ハードウェアやネットワーク環境のセットアップ、オペレーティングシステム(OS)など、テスト対象のソフトウェアを動かすためのソフトウェアのインストールとセットアップが含まれます。シミュレーション環境の構築もテスト環境の準備の一部といえます。

- テストデータの準備：テストを実行するための事前条件となるデータを作り、テスト対象のソフトウェアにセットアップします。

- テストケースの実行順序の決定：テスト設計時に作ったテストケースの実行順序を決めます。決める際には、優先順序や開発全体のスケジュールを考慮して、作成したテストケースをどのビルドで行うか決定します。そして、そのビルドでテストを実行するメンバーにテストケースを割り振ります。

- テスト実行スクリプトの生成：テストを実行するメンバーに割り振ったテストケースを、効率よく一連の流れで実行できるように順序立て、操作手順（テスト実行スクリプト）を作成します。手動でテストする場合は、事前に詳細な操作手順を作成し、どのタイミングでどのテストケースを確認するかを決めておきますが、あまり操作手順を意識しなくてもよいテストケースであれば、操作手順は実行するメンバーにテスト実行時に決めてもらうこともあります。自動テストを行う際のスクリプティングもこの作業に含まれます。

テスト実装は、どのテストレベルでも必ず行いますが、テストレベルによって作業量が異なります。例えば、ユニットテストでは、開発担当者の開発環境でテストをすることが多いので、環境の準備はあまり手間にはなりませんが、統合テスト、システムテストでは、テスト用の環境を構築することが多いため、環境の準備をスケジュールに組み込んでおかないと作業が間に合わなくなります。

開発プロジェクトの進捗に左右されず、テストができるように準備することもテスト実装の重要な作業です。複数のシステムが連携する大規模システム（システムオブシステムズ）のテストでは、他のシステムの開発日程が遅れると、自分たちで開発しているシステムのテストが開始できないといった問題が起きることがあります。その場合、予めシミュレーション環境を用意しておき、日程に左右されないようにします。

開発プロジェクトのどの時期で行うべきか

- テスト環境の準備やテストデータの準備はテスト実行を開始する前に完了するように日程を計画します。そういう意味では、開発全体の日程とは別に計画ができます。しかし、上述したように開発の日程遅延が準備日程に影響を与えることがありますので、予め日程遅延が起きた時の対策を採っておくことが必要です。例えば以下のようなことが考えられます。
  - シミュレーション環境やスタブとなるソフトウェアを用意しておき、開発日程に関係なくテストの開始を可能にする。
  - データ作成、環境構築、および、データの投入といった作業の時間を短縮できるように手順化、もしくはその手順を自動化する。

- テストケースの実行順序やテスト実行スクリプトも基本的にはテスト実行を開始する前に完了するように日程を計画しますが、以下のようにテスト実行中にも実装が必要になります。
  - テストケースの実行順序は、開発日程の遅延や、欠陥によるテスト待ちで変更することがあるため、テスト実行作業中も見直しをします。
  - テスト実行スクリプトは、テスト対象となるソフトウェアが構築されて実際に操作可能にならないと作れない場合があります。
  - テスト実行中であっても、仕様変更があれば、テストケースとテスト実行スクリプトは見直しを行います。

この作業が上手くいっていないとどんなことが起きるか（症状）

- テスト環境やテストデータの準備に時間がかからないようにするための手順化、標準化が必要です。そうしないと以下のような問題がおきます。
  - 環境設定にいつも時間がかかり、それが原因で日程遅延する。
  - 環境の作り間違えで、テストやり直しになる。
  - テスト環境やテストデータの準備が専任者の個人芸になってしまうと、その専任者不在時に誰も準備ができなくなる。
- テストケースの実行順序を事前に準備しないと以下のような問題がおきます。
  - テスト実行時にテストケースを選択するため、メンバーの割り振りに時間がかかってしまう。
  - 進捗状況を把握できなくなる。（テスト結果管理が上手くいかなくなる）
- テスト実行スクリプトの準備の程度によって以下のような問題がおきます。
  - バッチ処理など操作手順をひとつでも間違えるとテストやり直しになるケースは、事前準備しておかなければ、かえってテスト実行工数が増えてしまう。
  - テスト実行の操作手順を事前に作りこみすぎると、仕様変更のたびに見直す工数が膨大になる。

どのような種類のツールを使うのか

- シミュレータ（スタブツール）
- ラボイメージ
- テストデータジェネレータ
- テストケース管理ツール

コード解析

何をするのか

大規模で複雑なソフトウェアの開発においては、すべてのソースコードを人手によりレビューすることは困難です。近年では、静的解析ツールを用いたソースコード解析（以下、コード解析）が、高品質なソフトウェアを構築するための重要な鍵のひとつとなってきています。コード解析は、人手によるコードレビューを補完、置換するものであり、ツールを用いてソースコードを解析することにより、特定の欠陥や欠陥につながる可能性のあるソースコード上の特徴を自動的に発見したり、テストの実行を助けたりすることができます。これにより、人手によるコードレビューと比較して、欠陥発見に必要な時間を短縮したり、レビューの実施回数を増やしたりすることが可能となります。加えて、欠陥発見能力のばらつきを抑えることもできます。また、人手によるレビューの対象を、ツールでは発見が困難な仕様の問題などに集中させることが可能となるため、より効率的なコードレビューを実施することにも役立ちます。

コード解析の用途はさまざまですが代表的なものは以下の4つです。

- メモリリークやランタイムエラーなどプログラミングエラーの発見
- MISRA[3]などコーディングルールの遵守確認
- 行数や複雑度などのメトリクス情報の取得
- ソースコードからクラス図やコールグラフなどを生成するソフトウェアの構造解析

また、実際にコード解析の運用段階においては、プロジェクトマネージャやコード解析ツールの運用管理者は解析結果を上手く活用しているか（例えば、コード解析ツールで発見された欠陥はちゃんと修正されているか）の定期的な確認が必要です。運用上問題がある場合は、開発担当者への負荷を考慮した上で、必要に応じた運用方針の見直しが必要でしょう。

開発プロジェクトのどの時期で行うべきか

用途や用いるツールよって開発プロジェクトのどの時期にコード解析を行うべきかは異なります。

- コーディングルールのチェックやメトリクス情報を取得するコード解析ツールを利用する場合は、コーディングの初期段階から継続的に利用することにより、危険なコードの書き方を回避し、ソースコードの品質を高めることができます。
- プログラミングエラーの発見を行うコード解析ツールは、遅くともユニットテスト段階から利用することにより、ユニットテストでは見逃される可能性のあるエラーを発見し修正することができます。また、統合テストフェーズでは、テスト実行前に静的解析ツールで見つかるような問題を先に潰しておき、テストに必要な作業負荷を低減することができます。加えて、テスト期間中にソースコードに変更が発生した場合は、コード解析ツールを実行することにより、副作用の混入を防ぐこともできます。
- 構造解析ツールは、既存のソフトウェアの構造を理解するために、ソフトウェアの設計段階で用いることがあります。またコーディング段階においては、設計通りの実装が行われているかチェックする上で用いることもあるでしょう。
- 継続的インテグレーションを採用している開発プロジェクトにおいては、ビルド、ユニットテストと共にコード解析（静的解析ツール）を実行することにより、欠陥の早期発見を行うことが望ましいでしょう。

[3] Motor Industry Software Reliability Association が開発した C 言語のためのソフトウェア設計標準規格

この作業が上手くいっていないとどんなことが起きるか（症状）

- コーディング段階でコード解析を実行せずに、最終納品のタイミングで初めてコード解析を行うと、修正時間が取れなかったり、修正に伴う変更のテストが実行できなくなります。結果として、開発スケジュールに遅れの発生や、欠陥が修正されないままソフトウェアが出荷されるなどの問題が生じます。
- コード解析ツールの運用計画が不十分な場合、コード解析ツールの解析結果が上手く活用されず、結果として品質向上に寄与しないという状態に陥ります。
- コード解析で指摘されるコードレベルの問題が統合テストやシステムテスト段階まで残っていると、テスト担当者と開発担当者のコミュニケーションや原因の調査に時間がとられ、テスト実行の工数が非常に増加します。

どのような種類のツールを使うのか

- 静的解析ツール
- 構造解析ツール

テスト実行

何をするのか

テスト実行とは、通常「テスト」と呼ばれている作業のことを指します。担当者が自らテスト手順をひとつひとつ行いながらソフトウェアの機能テストなどを人手で行う作業を「テスト手動実行」と呼び、ツールを活用してテスト実行を自動で行うことを「テスト自動実行」と呼びます。本節では、「テスト自動実行」に焦点を当てて記述していきます。テストは、システムおよびソフトウェアの開発プロセスの中で何度も繰り返し実行されるものです。

特に以下のようなテストは何度も繰り返されます。

- 回帰テスト： ソフトウェアへの機能追加や問題修正などの対応により、既存コードに手が加わった場合においても、期待される振る舞いが変わらないことを確認するテストです。テストを実行する上では、その修正による影響範囲が明確になっていることが重要で、その範囲を重点的に確認しますが、それと同様にその範囲外も既存機能が正しく稼動するか確認する必要があります。つまり、変更が加わらないコードに対して繰り返しテストを行うことでソフトウェア品質のベースラインを担保することができます。

- 性能テスト・ストレステスト： ソフトウェアに対する性能要求を満たしているか確認するテストです。一般的には、トランザクションあたりのスループットやレスポンスタイムを計測します。また、複数ユーザによりさまざまな処理が並列実行されるような高負荷状態時でも、設計上性能に関するボトルネックがないかどうか、システムの CPU 使用率、メモリ使用率、および、ディスク I/O などと合わせて検証し評価します。

- セキュリティテスト： ソフトウェアの脆弱性を確認するテストです。外部からの不正アクセスを防御できているか実際に稼動するソフトウェアに対してアタックをかけ、セキュリティに関する考慮漏れがないか検証します。

上記のような繰り返し実行されるテストは、ツールを使いテスト実行を自動化することにより、大幅にテスト実行の効率化が図れます。入力項目が多い場合やそのバリエーション/パターンが多岐にわたる場合にもツールを使用するメリットがあります。また、たとえテストを繰り返し行わないとしても、同じ処理を複数回実行するもの（例えば性能テストでの大量ユーザの同時使用など）であれば自動実行を行うメリットがあります。しかしながら、「テスト自動実行」については以下の点に注意する必要があります。

- すべてのテストケースが自動化できるわけではない。
- まずは単調なテストシナリオから自動化を試みて、その範囲を徐々に増やしていくのがよい。
- テスト途中に人の判断が必要なテストシナリオは自動化できない。

開発プロジェクトのどの時期で行うべきか

- テスト対象となるソフトウェアを構築した後に、設計通り実装できているかどうかそれぞれの機能について確認を行います。開発時だけでなく、保守フェーズに入っても活用されます。テスト実行時はソフトウェアとして、ただエラーがなく動くことを確認するだけでなく、テスト結果として得られた内容が期待通りの値や表示を返してきたかを検証します。
- 性能テストおよびストレステストは、システムテスト時に実行することが多くなります。（ユニットや統合のテストレベルでも性能の確認を行いますが、最終的な性能はシステムとして確認すべきだからです。）統合

テストにて基本的な機能が確認された後、複数のコンポーネントが統合された状態で性能面でのボトルネックがないか検証します。

- セキュリティテストは一般的にはシステムテスト時に実行されます。ただし、近年ではコード解析によりセキュリティ面の強化が重要視されており、ユニットテスト時からシステムテストにかけて使用されます。早期に問題を見つけ修正することにより、作業の手戻りを抑止し同時に品質向上に貢献します。

この作業が上手くいっていないとどんなことが起きるか（症状）

- テスト作業の効率化が進まず、さらに品質低下につながります。
  - 何度も同じことを行うテストを人間がやっていると何倍も工数がかかる。
  - 人間が実行する場合、時間的な制約が出てくる。（夜間テストなどが困難）
  - 同じことを何度も人間にやらせると間違えたり、実行漏れが発生したりする。
- 性能テストが十分でない場合、本番リリース後に重大なトラブルが発生する可能性があります。
  - 急激な負荷により、最悪の場合、システムダウンにつながることがあり、ビジネスの損失や信頼度の低下につながる。
- セキュリティテストが十分でない場合、不正アクセスによりデータ漏洩などが発覚し、世間や顧客からの信頼度が低下し、企業にとって多大な損出を出してしまう恐れがあります。

どのような種類のツールを使うのか

- ユニットテストツール
- キャプチャ/リプレイツール
- 性能テストツール
- セキュリティテストツール
- テスト自動実行支援ツール

テストウェア管理

何をするのか

テストウェアとは、開発プロセスを通して生成されるテスト関連成果物のことです。テスト対象のソフトウェアによって内容も、粒度も、変わることがあります。主なテストウェアは、以下のようになります。

- **テストケース**: テストに必要な前提条件、操作、その結果を記載したものです。テストケースは、テスト条件を満たすように作成し、管理する必要があります。テストケースは、テスト計画、テスト設計、テスト実装、テスト実行、テスト結果で利用、作成、参照されます。テストケースは、通常自然言語で記述されます。

- **自動テストスクリプト**: テストを完全、または、部分的に自動化した際の、自動実行を定義したスクリプトです。バッチ処理や、製品コードと同一のプログラミング言語、または、自動テスト専用のスクリプト言語で記述されることがあります。これらは、自動化されたテストケースにおいては、密接に関係づけられます。

- **セットアップ/クリーンアップ**: テストを実行する前後の処理を指します。特に、テストに必要な前提条件を整えるためのセットアップと、テスト実行後の環境のリセットを行うクリーンアップがあります。これらは、バッチ処理、製品コードと同一のプログラミング言語、または、自動テスト専用のスクリプト言語で記述されることがあります。

- **テストデータ**: テストを実行する際のデータを指します。テストに必要な前提条件の一部とも言えます。テストデータを整備することで、テストの精度と結果を均一化することが見込めます。テストデータには、データが格納されるデータベースのスキーマなども含まれます。

- **テスト環境の構成**: テストを実行する環境の構成です。例えば、3 層アプリケーションの場合は、テスト環境の構成は、「クライアント - アプリケーションサーバ - データベースサーバ」となり、これらを複数組み合わせたテスト環境を準備する必要があります。テスト環境の構成には、テスト対象のソフトウェア（ビルド）、テストで利用するソフトウェア/ハードウェアも含まれます。

- **テスト支援ドキュメント**: テストを円滑に実行、持続可能にするためのドキュメントを指します。例えば、テスト実行のための確認チェックリストや、環境整備ガイドなどです。これらは必要に応じて作成し、更新する必要があります。

テストウェア管理とは、テストウェアを適切に管理することです。テストウェアの構成管理を行い、適切なテストウェアをいつでも、誰でも取り出せるようにすることを目的とします。

テストウェアは、開発プロセス全般で継続的に利用されるため、関係者間で共有されている必要があります。テスト効率の向上と煩雑さの軽減のため、テストウェアの再利用性の向上を考慮するとよいでしょう。また、状況に応じて頻繁に更新されるものが多いため、バージョン管理は必須で、これらテストウェアはそれぞれに関連があるため、テストウェア間の追跡可能性も意識しなければなりません。

したがって、テストウェアの構成を適切に管理することが求められます。テストウェアはソフトウェア製品コードとも密接に関連するため、ソフトウェア構成・変更管理と統合して管理する必要があります。

開発プロジェクトのどの時期で行うべきか

- テスト分析の開始時より、ソフトウェア構成を管理していく必要があります。これらは、その後の開発プロセス全般で参照されるため、共有と維持が行えるよう構成を識別する必要があります。
- テスト分析、結果分析により、頻繁に更新が必要な場合があります。そのため、テストウェアのバージョン管理を行い、変更を追跡可能にしておく必要があります。
- テストウェアは、テスト効率の向上と煩雑さの軽減のため、再利用可能な設計を行う必要もあります。再利用可能なテストウェアかどうかを判断、識別するためにも、テストウェアは、開発プロセス全般で意識する必要があります。
- テストウェアは、ソフトウェア製品コードとも密接に関係するため、開発担当者による管理や更新の可能性もあります。そのため、ソフトウェア製品コードと同一の構成・変更管理を行う必要も考慮に入れる必要があります。

この作業が上手くいっていないとどんなことが起きるか（症状）

- テストの設計、実装、実行が共有されず、属人化し、テストの信頼性の低下、テスト効率の低下を引き起こします。例えば、セットアップスクリプトを各自が独自のものを作成している、テスト環境を手動でそれぞれのやり方で行っているなどです。
- テストの実行が省力化されず、効率の低いテストが繰り返されます。その結果、予定されたテストが実行できず、テスト成果物も蓄積されず、開発プロセス全般にわたって効率化が阻害されます。例えば、テストスクリプトを共有しておらず、それぞれが都度作成する場合があります。

どのような種類のツールを使うのか

- 構成管理ツール

テスト結果管理

何をするのか

テストを実行する際には、テストが計画通りに正しく行われているかを見極めるために、テストの計画と実績の管理が必要になります。テストの実績はテスト結果から判断しますので、これをテスト結果管理と呼びます。テスト結果管理をすることで、計画からの逸脱を早期に発見し、軌道修正することができます。テスト結果管理は次のような手順で行います。

- テスト結果の記録/計測/収集：テストを実行したら、その結果を記録します。例えばテストが未実行か実行済か、あるいはテストに成功したか失敗したかといった情報があります。
- テスト結果の可視化：数値などの生データだけでは、状況や結果を把握しづらいことが多くあります。その場合には、グラフや表などに加工することで理解しやすくします。
- テスト結果の分析/評価：得られた結果を基に、テストの実行結果に問題がないかどうかを確認します。問題がある場合は、原因を分析し、対策を検討、実行します。
- テスト結果の報告/共有：分析、評価した結果を文書にまとめて、顧客やプロジェクトマネージャなどの利害関係者に報告します。また、結果は開発プロジェクト内の関係者とも共有しておきます。

テスト結果管理で確認、評価すべき事項やその手法のうち、代表的なものを以下に挙げます。

- テストの進捗状況の把握
  - 日々のテストケースの実行結果を記録し、未実行のテストケース数がどれだけ残っているかを、その予定数とともに折れ線グラフに表すのが一般的です。つまり、通常は右下がりの折れ線になります。これにより、テストの進捗が順調か否かを判断できます。
- テストケースの十分性
  - 例として、管理単位（モジュール、機能など）毎に、その規模（コード行数やファンクションポイント）に応じて必要とされるテストケース数の基準を、予め設定しておく方法があります。その場合、上限値や下限値のように幅を持った基準を設定します。そして、実行したテストケース数がその基準の範囲内に収まっているかどうかによって、テストケース数の十分性を判断します。
  - 「コード解析やテスト実行」でのカバレッジ計測の結果を利用して、テストの実行漏れがないかどうかを判断します。
- 欠陥が潜在する可能性
  - テストケースと同様に、管理単位毎に規模に応じて検出すべき欠陥の数の基準を設定しておきます。そして、検出した欠陥数がその基準の範囲内に収まっているかどうかによって、欠陥を検出し尽くしたと言えるかどうかを判断します。
  - テストの実行期間や実行したテストケース数に対して、検出された累積の欠陥数を折れ線グラフに表し、欠陥の検出状況の推移を把握することができます。グラフは通常は右上がりになりますが、ある程度の期間が経ったところでグラフが水平になれば、欠陥を検出し尽くしたと判断できます。

開発プロジェクトのどの時期で行うべきか

- テスト実行が始まると同時に開始し、テストが終了するまで行います。
- どのような管理や評価をするかについては、テスト実行前に決めておく必要があります。
- テスト実行だけでなく、テスト設計やテスト実装も管理対象とし、テストケース作成やテスト実行スクリプト作成の進捗を管理することも推奨されます。

この作業が上手くいっていないとどんなことが起きるか（症状）

- テストを終了してよいかどうかを正しく判定できません。具体的には次のようなことが起こりえます。
  - テストが不足しており、欠陥が残っているにも関わらず、テスト終了と誤って判断してしまいます。これにより、欠陥を本来検出すべき工程よりも後で欠陥が顕在化することになり、その原因分析や修正に伴う手戻り工数が発生します。また、欠陥が残ったまま製品をリリースしてしまった場合には、欠陥情報の周知、修正パッチの提供、クレーム対応、さらには顧客やエンドユーザからの損害賠償請求といったさまざまな損失を被る可能性があります。
  - 計画していた品質を十分に満たすだけのテストをしたにも関わらず、過剰にテストをして無駄に工数を費やすことになります。これにより、コストの超過や納期の遅延につながる可能性があり、たとえ品質が十分であっても顧客満足度が下がることになります。
- テストが順調に進んでいるのかどうかを把握できません。例えば次のようなことが起こりえます。
  - テスト期間の終盤になって、テストが計画通りに進んでいないことに気付き、期限までに間に合わせようと残業や休日出勤などで対応しようとします。それにより、テスト担当者の疲労度が増し、心身ともに不安定になります。さらに、このような状況で実行するテストにはミスが起こりやすいため、欠陥を見逃してしまい、適切な品質を保証できない可能性があります。
- テスト結果を関係者間で共有できません。例えば次のようなことが起こりえます。
  - あるテストレベルにおいて欠陥が検出されたものの、原因が特定できないために修正できないまま、条件付きで次のテストレベルに進むことがあります。その場合、未修正の欠陥があることをテスト結果レポートに明記しておき、その欠陥に関連する箇所のテストの扱いについて、次テストレベルの担当者と共有しなければなりません。しかし、それを怠ると、次テストレベルの担当者は欠陥があることを知らずにテストしてしまい、本来実行すべきでないテストに無駄な労力を使ってしまうことになります。

どのような種類のツールを使うのか

- テスト結果管理ツール
- テスト結果レポートツール

インシデント管理

何をするのか

インシデントとは、ソフトウェアの開発～運用含めたライフサイクルすべてにおいて発生する「調査が必要な事象」のことを指します。インシデントは、調査した結果、システムやコンポーネントの欠陥であることが分かると修正を要することになります。（調査した結果、システムの欠陥ではなく運用ミスや仕様の誤解釈であることもあります。）

インシデント管理とは、発生したインシデントが解決するまでの状況を見失わないように追い続けることです。テストを行うと、テスト対象の動作不備を見つけることができますが、見つけた動作不備はシステムの欠陥かどうかの調査から始まり、欠陥であった場合は修正されるまで追い続け、最後は修正されたことを確認するための再テストを行うことになります。これらの作業を「ステータス」と言う項目で分類し、ステータス毎に誰が何を行うのかを決めます。各ステータスで必要な作業が完了したら次のステータスへ進めます。このようにステータスを進めていくことで、インシデントの対応が完了するまでの間、いつ誰が何を行うべきかがすぐに分かるようになります。

インシデント管理をする際に、それぞれのインシデントに対し、事象の詳細、欠陥修正をすることになった担当者、発生原因、重要度といった情報を付与していくことで以下の関係者に有益な情報を提供することができます。

- ***開発担当者やその関係者に対し、必要に応じて問題を特定、抽出、解決できるようフィードバックをする。***
- ***テストリーダに対し、テスト実行中のシステムの品質や、テストの進捗を追跡する手段を提供する。***
- ***テストプロセス改善のためのヒントを提供する。***

（ISTQB シラバスより引用）

また、インシデント管理は、開発組織によって、チケット管理、欠陥追跡、障害管理、バグ管理、不具合管理、バグトラッキングなどさまざまな呼び方がありますが、本書では、それらを総称してインシデント管理で統一します。

開発プロジェクトのどの時期で行うべきか

- 本来は、ソフトウェアライフサイクルのすべての時期において、成果物の検証や妥当性確認を行う際にはインシデント管理が必要です。（ソフトウェアライフサイクルとは要求分析～テストといった開発プロセスだけでなく、運用後と欠陥対応、仕様追加対応の開発も含めた期間を指しています。）

- ただし、インシデント管理の公式度合いは開発プロジェクトの計画に基づいて柔軟に変更するのが現実的です。例えば、ユニットテストの時期に発見したインシデントは、開発者個人だけで問題の解決までの一連の作業を管理してかまいませんが、システムテスト以降や、運用に入ってから発見したインシデントは、ステータス毎に担当を割り当てて、プロジェクトに関係する全員が各状況を閲覧できるべきです。

この作業が上手くいっていないとどんなことが起きるか（症状）

- インシデントは計画的に発見できるわけではないため、インシデント対応に関係する作業は割り込み作業として行うことが多くなります。また、対応完了するまでに多くの人が関わるため、すぐに自分が作業でき

るわけではなく、誰かの作業を待ってから着手することになります。（例えばテスト担当者がインシデントを登録した後、修正済みの成果物を再テストするまでには何日もかかります。）その中で、インシデント管理を怠ると、以下のような問題が起こります。

- プロジェクト全員がいつまでに誰が何を行わなければならないかが把握できなくなり、メンバー個人が何をしなければならないかを見つけるところに時間がかかり、作業の速度が著しく低下する。
- テスト担当者が発見したインシデントが開発者に報告されずいつまでも対応されない。（もしくは対応を忘れたままリリースされてしまう。）

- インシデント管理を公式に行わないと、インシデント管理で得られる有益な情報が収集できない、もしくは関係者に情報がいきわたりません。例えば以下のような情報が該当します。
  - 開発の進捗に関する情報。（あと修正が必要な欠陥があと何件残っているか？）
  - 成果物の品質を把握するための情報。（テストをしても欠陥が見つからなくなっているか？）
  - 開発プロジェクトを振り返るための情報。（どんな欠陥が多く見つかったのか？）

どのような種類のツールを使うのか

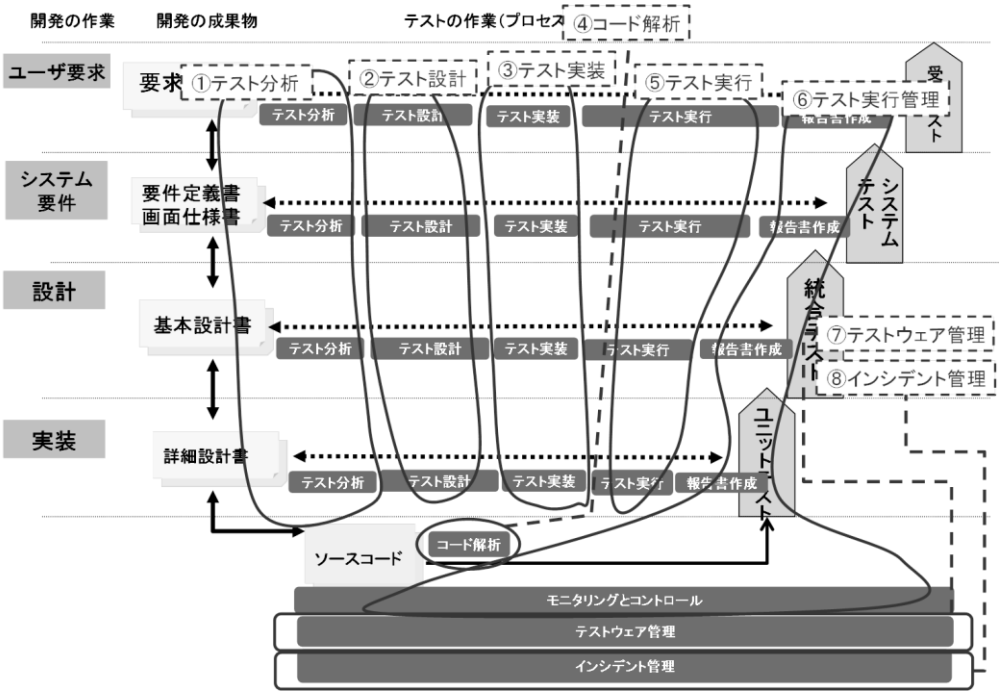
- インシデント管理ツール

第四章　テストツール体系

本章の内容

第三章では、ソフトウェア開発において、テストをする時にはどのような作業が必要なのかを解説しました。各作業の解説にてその作業を支援するツールの種類を明記しましたが、本章では、それらのツールの種類毎の解説をします。第三章の作業における課題を解決する「ツールの効能」と、「ツール使用上の注意」を併記していますので導入検討時の一助にしていただけます。

テストツール全体像



No.	テストの作業	作業を支援するツールの種類
①	テスト分析	要件管理ツール
②	テスト設計	状態遷移テストツール、組合せテスト支援ツール（直交表、オールペア）、原因結果グラフツール、動的解析ツール、カバレッジ計測ツール、その他のテスト設計支援ツール
③	テスト実装	スタブツール、シミュレータ、ラボイメージツール、テストデータジェネレータ、テストケース管理ツール
④	コード解析	静的解析ツール、構造解析ツール
⑤	テスト（自動）実行	ユニットテストツール、キャプチャ/リプレイツール、性能テストツール、セキュリティテストツール、テスト自動実行支援ツール
⑥	テストウェア管理	構成管理ツール
⑦	テスト結果管理	テスト結果管理/テスト結果レポートツール
⑧	インシデント管理	インシデント管理ツール

テスト分析

要件管理ツール

ツールの概要

テスト対象の要件を体系立てて管理するツールです。殆どの製品で以下の機能が搭載されています。

- Word、Excel などの文書からデータをインポートして要件定義として管理する。
- 要件を構造化して視覚的に管理し、要件の確認を容易にする。
- ベースラインの要件と仕様変更後の要件の差異、およびその影響分析をわかりやすく提示する。
- 要件を基にテスト条件を管理する。（テスト条件生成、トレーサビリティ確保）
- 各要件に該当する開発成果物（モデルやソースコードなど）のトレーサビリティを確保する。

ツールの効能

- 要件からテスト実行結果まであらゆる開発成果物のトレーサビリティ管理の自動化を支援し、要件カバレッジの確認ができる。
- 要件からテスト条件が自動生成できるツールの場合、その後の開発において特定要件のテストカバレッジがどのくらいなのかなどの紐付けが容易になる。
- 要件変更の影響分析を行うことにより、その後の開発工程にどのくらい影響が出るのか視覚的に提示できるので、顧客への認知が行いやすくなる。（あらゆる成果物の変更のインパクトを解析）
- 膨大な要件数の管理が可能となる。
  - 重要度や優先度、カテゴリやリリース時期などのフィールドをキーに柔軟にアイテムの切り分け、および切り分け条件の保存や共有が可能となる機能を搭載していることが多い。
  - データベースシステムであることが多く、データベースの能力で何千万件も登録が可能となる。そのため、ファイルでの管理では分散してしまう情報を一元管理できる。
- 国際スタンダード認証取得時のエビデンス確保の手段として活用できる。

ツール使用上の注意

- 要件管理として基本となるベースラインを固定化すること。これができないと、顧客とのレビュー後の要件の差異を分析できなくなる。この場合、ツールを使ってもベースがぶれているため効果が期待できない。
- トレーサビリティ管理が重要であることを理解し、要件を登録する際は、登録するそのタイミングで他の要件やテストとの関連を明確にしてツールに登録すること。後でやろうとすると調査しなければならない関連が増えすぎてしまい作業負荷となり関連の登録がおざなりになる。
- トレーサビリティをとる粒度を明確に定義すること。ツールによっては、ドキュメントの 1 行 1 行に対して、その関連を紐付けすることも可能だが、ただ網羅的に管理すると言うだけで細かくしすぎると、その細かさゆえに膨大な関連ができてしまい、関係が維持できなくなる。また、粒度がばらばらのまま紐付けをしていっても関係が理解できなくなる。

テスト設計

状態遷移テストツール

ツールの概要

ブラックボックステスト技法のひとつである、状態遷移テストに基づいたテスト設計作業を支援するツールです。

状態遷移テストとは、テスト対象となるコンポーネントやシステムが取りうる「状態」と、状態が遷移するきっかけとなる「イベント」や遷移するための「条件」を図や表形式で設計するためのテスト設計技法です。この技法を使うことでテスト対象が取りうる動きとその遷移条件を有限の個数でモデルとして表現でき、網羅基準が明確になります。状態遷移を図で表す方法と表で表す方法がありますが、それぞれ以下のようなメリットがあります。

- 状態遷移図を用いることで、テスト対象が取りうる動きを俯瞰して理解できる。
- 状態遷移表を用いることで、各状態が受け付けるイベントとの組合せを漏れなく表現できる。

また、この技法は、ユーザシナリオ、画面設計、アーキテクチャ設計、プログラムの詳細設計などに対するさまざまなテストで利用することができます。

ただし、状態遷移モデルからテストケースを作成するのは手動で行うことも可能ですが、間違いが起こりやすく、かつ作業量も多くて大変になります。状態遷移テストツールはこういった一連の作業を自動化するツールです。

※　技法の詳細は参考文献に掲載されているテスト設計の書籍を参照してください。

ツールの効能

- 状態遷移モデルからテストケースを生成する作業が自動になるため、短時間で間違いなく正確にテストケースが生成できる。
- 仕様変更に伴う状態遷移モデルの修正や追記が容易になる。
- 状態遷移モデルによってテストケースがいくつ必要になるか計算するのが自動でできるため、網羅率の計算が容易になる。
- 状態遷移テストツールとテスト自動実行ツール、テストデータジェネレータツールを組み合わせることで、モデルベースドテスト（テスト用に記述したモデル作成からテスト実行までを自動化する）が実現できる。

ツール使用上の注意

- モデルを描くのはツールではなく技術者が行わなければならないため、以下のことに注意してモデリングすること。
  - 状態遷移図を作る際に各状態の抽象度を合わせておかないと、状態間の接続（リンク）が多くなりすぎて収拾がつかなくなったり、遷移条件を表現しきれなくなる。
  - 状態数やイベント数が多くなりすぎて（状態爆発とも呼ぶ）、実質網羅性を確保したテストケース数に収まりきらなくなる場合もある。
  - 抽象度を合わせるには、内部状態を持つ状態ならば、その状態の内部の状態遷移を隠蔽化（コンポジット）することでメタに表現し、他の状態と抽象度を合わせることができる。また個々のイベントをすべて取り上げるのではなく、性質が似ているイベントを有効同値クラスと見なすことで複数のイベントを1イベントとして扱うなど、同値分割が利用できることもある。

  - イベントに対する自己遷移（状態が変わらない）時の挙動や、ある状態におけるイベントをありえないものと判断する時に判断ミスが起こりやすい傾向があるため、他の技術者にレビューしてもらうなどしてテスト設計ミスを防ぐ必要がある。
  - 状態遷移図・表により状態遷移の流れは把握できても時系列での動きは確認できない。時系列を考慮したテストケースを作成するには、シーケンス図やタイミング図といった別のグラフを併用してテスト設計しなくてはならない。
- 組織内で統一した記法ルールとツールの支援する記法を合わせること。状態遷移図の記法にはいくつか種類があるが、（例： UML、ムーア図、ハレル図など）ツールはどれかひとつの記法しかサポートしていない場合がある。
- 技法の理解なく単にツールを用いたテスト設計を行わないこと。理解なくテスト設計を行ってしまうと、上記注意事項に対する考慮がされない可能性が高くなる。

組合せテスト支援ツール（直交表、オールペア）

ツールの概要

ブラックボックステスト技法のひとつである、組合せテスト技法（直交表やオールペア）に基づいたテスト設計作業を支援するツールです。

直交表は、仕様として相互関係を持たない複数の機能を組み合わせて動作させた時に、組合せによる問題の有無を検証するための技法です。相互関係がないため「直交している」といえます。この技法では機能のことを「因子」、各機能が持つ要素やオプションといった選択肢のことを「水準」と呼びます。

例えば、音楽再生機能があったとして、再生可能なフォーマットが MIDI、MP3、WAV、AIFF、ATRAC3 ならば、この因子は5つの水準を持っていると表現します。

直交表を用いたテストでは複数の機能を組み合わせたテストケースを作成した時に、あるふたつの機能の組合せ（2 因子）において、水準どうしの組合せの出現回数を数えた時に、どの組合せでも同じ回数だけ現れるという特徴があります。

オールペアは、任意の 2 パラメータ（ふたつであることからペア）を取り上げた時に、すべての値の組合せパターンを含むテストケースを作成します。（オールペアでは直交表で言うところの因子をパラメータ、水準を値と呼びます。）直交表との違いは、水準どうしの組合せの出現回数の公平性を確保しないことにあります。

直交表もオールペアも機能間の組合せに対して網羅度の高いテストケースを作成できるテスト設計技法です。この技法を使うことで以下のような効果があります。

- 機能間の組合せに対して網羅度の高いテストケースを作成できる。
- 合理的に組み合わせるべきテストケースを選択できる。
- 水準に対して、適切に同値分割を適用した場合、網羅度を確保しつつテストケース数を削減できる。

ただし、直交表やオールペアを使って組合せパターンを作成するのは手動で行うことも可能ですが、間違いが起こりやすく、かつ作業量も多くて大変になります。組合せテスト支援ツールはこういった一連の作業を自動化するツールです。

※　技法の詳細は参考文献に掲載されているテスト設計の書籍を参照してください。

ツールの効能

- 直交表やオールペアのモデルからテストケースを生成する作業が自動になるため、短時間で間違いなく正確にテストケースが生成できる。
- 仕様変更に伴う直交表やオールペアのモデルの修正や追記が容易になる。
- 直交表やオールペアのモデルによってテストケースがいくつ必要になるか計算するのが自動でできるため、網羅率の計算が容易になる。

ツール使用上の注意

- 直交表やオールペアのモデルを描くのはツールではなく技術者が行わなければならないため以下のことに注意してモデリングすること。
  - 直交表は、因子数の組合せが増える毎に一次関数的にテストケース数が増える。

- ペアワイズは組合せ条件を絞っている分、直交表に比べれば因子数の増加によるテストケース数の増加は緩やか（対数的）になる。
- 直交関係にない機能間の組合せに適用した場合、組み合わせるべきテストケースに抜け漏れが出る。
- 仕様としてありえない機能間の組合せ項目（直交表だと「禁則」、オールペアだと「制約」と呼ぶ）をテストケースから除外する必要がある。

- 技法の理解なく単にツールを用いたテスト設計を行わないこと。理解なくテスト設計を行ってしまうと、上記注意事項に対する考慮がされない可能性が高くなる。

原因結果グラフツール

ツールの概要

ブラックボックステスト技法のひとつである、原因結果グラフに基づいたテスト設計作業を支援するツールです。

原因結果グラフ（CEG: Cause-Effect Graph）は、仕様の因果関係を「原因、中間、結果」というノードとして階層的に分け、ノードの論理的な関係性をノード間を接続するリンクで表現し、ノード間に制約がある場合には制約条件を表現することで整理・分析する技法です。テスト設計としては、原因結果グラフをデシジョンテーブルに変換しテストケースを導出するという流れとなります。

この技法を使うことで、自然言語で表現された複雑な仕様をグラフとしてシンプルかつわかりやすく表現できます。また、曖昧に定義された仕様箇所を明らかにし仕様間の論理的な関係性を整理していくことで、曖昧な仕様を無くしていくことができます。特に技術者とテスト対象のユーザ間における「想定外」に対する認識の相違を、技術者が開発段階で埋めることに効果を発揮します。

原因結果グラフからデシジョンテーブルを手動で変換することも可能ですが、間違いが起こりやすく、かつ作業量も多くて大変になります。原因結果グラフツールはこういった一連の作業を自動化するツールです。

※　技法の詳細は参考文献に掲載されているテスト設計の書籍を参照してください。

ツールの効能

- 原因結果グラフからデシジョンテーブルを生成する作業が自動になるため、短時間で間違いなく正確にテストケースが生成できる。
- 仕様変更に伴う原因結果グラフの修正やノードや制約の追記が容易になる。
- 原因結果グラフのモデルから自動生成されたデシジョンテーブルでテストケースがいくつ必要になるか計算可能になるため、網羅率の計算が容易になる。

ツール使用上の注意

- 原因結果グラフを描くのはツールではなく技術者が行わなければならないため以下のことに注意してモデリングすること。
  - 論理式に対する基本的な理解が必要。
  - 仕様が曖昧な状態から原因結果グラフを導く場合、テスト対象に関する知見を持っていないと、結果として論理関係や制約条件の抜け漏れが生じ、結果、不十分なテストになってしまう。
  - テスト技法としては、中級以上のスキルが求められるため、使い慣れるまではメンバー間で勉強会を開いたり、相互レビューを行ったりするなどして、スキルの獲得と定着を図る必要がある。

動的解析ツール

ツールの概要

ソフトウェアの構造やアーキテクチャをテストする際に、テスト設計やテスト実行の結果確認を支援するツールです。

ソフトウェアの内部の作りがソフトウェア設計通りに動作するかをテストする際には、内部構造をモニタリングし、モジュールやコンポーネント間の呼び出し（コーリングシーケンス）、メモリの解放や競合状態に対する対処がソフトウェア設計通り動作しているかを確認します。

ソフトウェア設計通りに内部で動作しているかを確認するのは、プログラムを一行ずつ実行し、モニタリングツールで都度確認を行うことも可能ですが、時間もかかり大変な作業になります。動的解析ツールはテスト結果を確認する部分を自動化します。一般にツールは実行履歴（トレース）の取得、結果をテキスト・表・グラフなどを介してレポートするなどの自動化を支援します。

内部構造のテスト設計支援だけでなく、テストにて検出したインシデントから欠陥を特定する際のデバッグツールとして活用することもできます。

ツールの効能

- メモリリークやメモリ破壊など、一度実行した限りでは目に見えないソースコード上の欠陥を検出する。
- 並列処理における競合状態（レースコンディション）や、デッドロックの発生の可能性など、テスト実行時には問題が発生しなかったが、タイミングなど条件次第では、インシデントとなりうるソースコード上の欠陥を検出する。
- 実行時の関数やメソッド呼び出しの頻度、および、実行時間を計算し、パフォーマンスのボトルネックとなりうる箇所を検出する。
- 実行されないコンポーネントを特定して追加のテストを考察できる。

ツール使用上の注意

- 動的解析ツールはテストケースの作成を直接支援せず、テスト設計の過不足を確認する際に有効なツールとなることを考慮すること。
- 発見したいコード上の欠陥を評価するために、どのようなテストケースが必要になるかは、別途テスト設計をすること。
- テスト実行環境にツールが対応しているか確認すること。
- テスト対象のプログラム言語にツールが対応しているか確認すること。

カバレッジ計測ツール

ツールの概要

プログラムの制御構造を網羅するホワイトボックステスト技法を支援するツールです。

ホワイトボックスのテスト設計では、テスト実行をした結果、網羅すべプログラム構造（全ステートメント、ブランチ、条件など）をどの程度動作させてテストできているかを計測することが必要になります。プログラム構造の網羅度合いの計測は、手動で行うことも可能ですが、プログラムコードに対して実際に動作したかを手動で確認するのは間違いが起こりやすく、かつ作業量も多くて大変になります。カバレッジ計測ツールはこういった一連の作業を自動化します。一般にツールは実行履歴（トレース）の取得、網羅された箇所の比率（%）を算出、結果をテキスト・表・グラフなどを介してレポートするなどの自動化を支援します。

ツールの効能

- カバレッジの計測を自動で行うことで、計測結果の精度を確保できる。
- 実行されないコード領域を特定して追加のテストを考察できる。
- 国際スタンダード認証取得時のエビデンスとして活用されている。

ツール使用上の注意

- カバレッジ計測ツールはテストケースの作成を直接的に支援をしないことを考慮すること。カバレッジ計測ツールの結果を見て、実行できていないコードを通るテストケースを適切に追加しないと、テストを大量にやってもホワイトボックスとしての網羅率が上がらない。
- テスト実行環境にツールが対応しているか確認すること。
- テスト対象のプログラム言語にツールが対応しているか確認すること。
- プローブ効果（ツールによる実行履歴取得のために起こる性能の低下など）が起こる可能性を考慮すること。

その他のテスト設計支援ツール

ツールの概要

本章で紹介したブラックボックステストのテスト設計技法（状態遷移テスト、直交表、オールペア、原因結果グラフ）に該当しないテスト設計作業を支援するツールです。

テスト対象の要件、仕様をツールベンダーが独自に開発したテスト設計の考え方に基づいてモデル化し、モデルを網羅するテスト設計仕様やテストデータ、およびテストテストケースを生成します。

モデルベースドテストツールと呼ばれる、テストケースの生成から自動テスト用スクリプトへの変換と自動実行をすべて支援するツールでは、複数の種類のテスト設計技法を作業を支援する（状態遷移モデルだけでなく他の複数の種類のモデルからテストケースを自動生成）もあります。

ツールの効能

- モデルからテストケースを生成する作業が自動になるため、短時間で間違いなく正確にテストケースが生成できる。
- 仕様変更に伴うモデルの修正や追記が容易になる。
- モデルによってテストケースがいくつ必要になるか計算するのが自動でできるため、網羅率の計算が容易になる。
- テスト自動実行ツール、テストデータジェネレータツールを組み合わせることで、モデルベースドテスト（テスト用に記述したモデル作成からテスト実行までを自動化する）が実現できる。

ツール使用上の注意

- ツールベンダーが独自に開発したテスト設計の考え方の理解なく単にツールを用いたテスト設計を行わないこと。理解なくテスト設計を行ってしまうと、テストケースに抜け漏れが起きていても判断できなくなる。

テスト実装

スタブツール

ツールの概要

スタブの作成を効率化、もしくは自動で行うツールです。

スタブとは、テスト対象のシステムやコンポーネントをテストするため、テスト対象から呼び出される特定目的のための最小限度のコンポーネントやシステムのことです。スタブによって、他のコンポーネントやシステムがなくてもテスト対象がどのように振る舞うかがテスト可能になります。

スタブは、開発担当者やテスト担当者が自ら開発することが可能ですが、スタブツールを使うと、ごくわずかの設定でスタブを使うことができるようになったり、もしくはスタブを自動生成することができます。

スタブを自動生成するツールは、ソースコードの静的解析結果からユニットインターフェイスのパラメータ、グローバル変数（入出力）、戻り値、変数タイプと用法を得て、コールされる関数を模倣したものを生成します。

ツールの効能

- スタブ作成の工数が短縮できる。
- スタブツールを共有することにより、インターフェイスの定義の標準化が容易になる。
- 静的解析ツール、カバレッジ計測ツール、ユニットテストツールなどと連携して以下のことを行うことが可能となるツールもある。
  - カバレッジ解析の自動化。
  - 回帰テストの自動化支援。（ソースコードの変更に応じてテストデータに変更が必要な箇所を指摘。）

ツール使用上の注意

- スタブから戻されるデータの正しさはツールでは検証できないため予め確認しておくこと。確認を怠るとスタブでは正しく動作していたものが実物に差し替えたとたん正しく動作しないということにつながってしまう。
- テスト実行環境にツールが対応しているか確認すること。
- テスト対象のプログラム言語にツールが対応しているか確認すること。

シミュレータ

ツールの概要

シミュレータとは、テスト対象のシステムやコンポーネントをテストするため、テスト対象を利用する相手の代表的な振る舞いを模倣する物理的、あるいは抽象的な装置、およびコンポーネントやシステムのことです。

シミュレータによって、他のコンポーネントやシステムがなくてもテスト対象によってテスト対象自身とテスト対象が利用する相手がどのように振る舞うかがテスト可能になります。

組込みソフトウェア開発では、ソフトウェアを搭載するハードウェアをシミュレーションすることでハードウェアがない環境でソフトウェアをテストすることができます。エンタープライズシステム開発では対向装置（テスト対象システムが利用する別システム）をシミュレーションすることで、本番環境ではない環境でシステム連携のテストができます。

ツールの効能

- 下記のような完成したシステムとなるためのテスト環境がなくてもテスト実行ができる。
  - 組込みソフトウェアで言うところの実機。（テスト対象ソフトウェアを搭載する実ハードウェア）
  - エンタープライズシステムで言うところの対向装置。（テスト対象が連携する別システム）
- 自動テスト実行ツールやスタブツールと連携することで、実世界では実現しづらいテストを実行できる。
  - 実システムよりも速く動作させる、もしくは大量に処理させることで、通常の動作では見つけることのできない欠陥を検出できる。（例えば性能テストのような非機能テストの実行）
  - 実システムよりも遅く動作させる、もしくは少量を処理させることで、通常の動作では見つけることのできない欠陥を検出できる。（例えば状態間の遷移が瞬時に済んでしまうエラー処理のテストの実行）

ツール使用上の注意

- シミュレータで確認できないことと、できることを明らかにしてテストを行うこと。あくまでもシミュレーションなので、フローティングの誤差、処理速度の違いなど実機との差異が生じる場合がある。
- シミュレータを使わなくても実行できるテスト（例えば計算ロジックの正確さや制御フローなど）までシミュレータでテスト実行しないこと。実システムをシミュレーションすることで、多くのテストが実現可能になるが、シミュレータで全部やろうとすると、その期間にテスト実行すべきテストケースが集中しテスト実行が遅延することがある。
- テスト実行環境にツールが対応しているか確認すること。

ラボイメージツール

ツールの概要

ラボイメージの作成や管理（保存したイメージを利用しやすいように一覧化するなど）、テスト環境への復元を効率化するツールです。

ラボイメージとは、本番運用環境にできるだけ近いように構築したテスト環境を再利用可能にしたイメージを指します。イメージとは、ディスクなどに記録されているソフトウェアの内容や構成をそのまま圧縮して複製したものです。例えば、3層 Web アプリケーションの場合、想定するクライアント層、アプリケーションサーバ層、データベースサーバ層の構成、これらのネットワーク構成と帯域をテスト環境としてイメージ化します（クライアント層では、OS、Web ブラウザの組合せなど多岐にわたるのが通常です）。

ラボイメージには、各層の OS, プリインストールするミドルウェアやエージェント、構成設定など可能な限り本番運用環境と同一のものを準備します。テスト実装作業では、準備したイメージをテスト環境に復元するだけで、すぐさまテスト実行が開始できます。ラボイメージツールはこれらの作業を効率化します。

最近では、ラボイメージを仮想化技術で容易に作成、構成し、再利用ができるツールも出てきています。仮想化により、テスト前のクリーンなイメージの保管や、テスト後の環境を問題の分析に利用するなどラボイメージの活用範囲を拡張することができます。

ツールの効能

- 本番運用環境と同等なテスト環境を作成する時間が短縮できる。
- 個別にテスト環境を準備する必要がなくなり、テスト担当者の環境設定にかかる負担（手作業、ミスによるやり直し、構成把握の手間暇など）が軽減される。
- 仮想化技術を活用したツールでは、以下のような活用が可能になる。
    - テスト環境の構成を一括で復元する。（例えば 3 層 Web アプリケーションの場合のクライアント層、アプリケーションサーバ層、データベースサーバ層のイメージをテスト用に一括で復元するなど。）
    - テストにてインシデントを検出した環境をイメージ化する。（開発者にわたすことでデバッグが効率化する、再テストをする際のテスト環境をインシデント検出時と同様の状態で復元するなどが可能になる。）

ツール使用上の注意

- 作成したラボイメージを復元するためのシステム環境が正しく動作するかを事前に検証すること。ラボイメージの復元に失敗してしまうと、結局テスト環境を作成するのは通常の手順を踏まなければならなくなるが、ラボイメージがあると思って通常の手順を行えるだけの工数を確保していないと、他の作業時間を圧迫してしまう。
- ラボイメージの管理を考慮すること。複数のラボイメージを保管しても、適切なタイミングに利用したいラボイメージがどれなのかがわからなくなると、テスト環境作成時間の短縮ができなくなる。

テストデータジェネレータ (利用データを基にデータ生成するツール)

ツールの概要

テスト対象のアプリケーションが使用するマスタデータやトランザクションデータなどを一定条件に基づき生成するツールです。

利用条件に基づくすべてのデータを生成するツールや、情報漏えいを防ぐために本番データに含まれる機密情報や個人情報をマスキングするなどして、本番データを流用するツールがあります。

ツールの効能

- テストデータの準備にかかる時間を短縮できる。
- 本番データの特徴を維持することにより、本番環境により近い環境でのテストが可能。

ツール使用上の注意

- データの件数や質(要素数や分布/ばらつきなど)が本番環境と合うようにすること。これらが異なるとテスト結果が OK となっても、本番では正しく動作しない場合がある。
- テストデータを生成する時間や、データ量などを予め見積もっておくこと。見積もっておかないと、スケジュールやストレージコストに影響を及ぼす場合がある。

テストデータジェネレータ (プログラムの期待値からデータ生成するツール)

ツールの概要

テスト実行時に使用するテストデータ（テストベクタとも呼ばれます）を特定のテスト技法（例：ペアワイズやデシジョンテーブル）や網羅基準に基づき生成するツールです。

例えば MBD(Model Based Development)[4]において制御モデルに対する特定の網羅基準を満たすテストデータを自動生成するツールや、ホワイトボックステスト技法のテスト設計に基づいた網羅基準（ステートメントカバレッジやブランチカバレッジ、MC／DC）を満たすテストデータを自動生成するツールなどがあります。

ツールの効能

- 網羅基準を満たすテストデータを効率的に作成できる。
- 機能（因子）間を組み合わせるテストデータの総数を合理的に削減できる。
- テスト自動実行ツールと連携することで、網羅基準を満たしたテスト実行が行える。

ツール使用上の注意

- 網羅基準を全て満たすテストデータを生成できない場合を意識すること。生成箇所については、人間がデータを作成する必要がある。
- テストデータを生成する時間や、データ量などを予め見積もっておくこと。見積もっておかないと、スケジュールやストレージコストに影響を及ぼす場合がある。

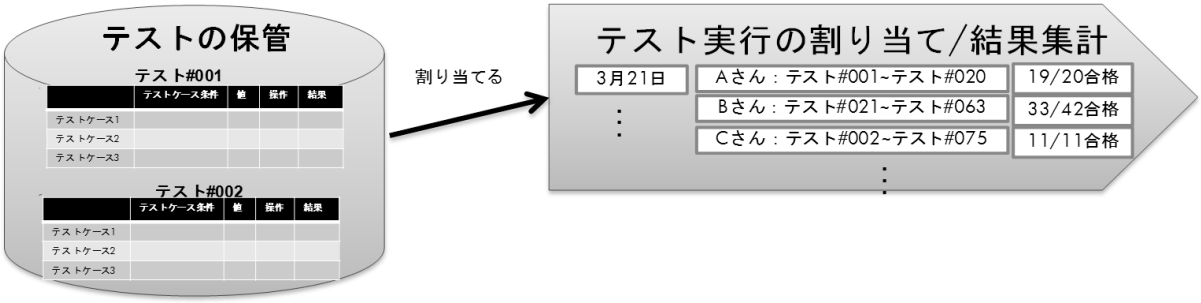
[4] 開発の初期段階にて機能をモデルで作成し、開発の上流工程から下流工程においてモデルを検証しながら開発プロセスを進めていく開発手法のこと。

テストケース管理ツール

ツールの概要

テストケースの情報を管理するツールです。主に以下の機能が搭載されています。

- テスト設計で作ったテストケースを保管し、検索やメンテナンスを容易にする機能。
- 保管してあるテストケースから、テスト実行するテストケースを選択し、実行単位にまとめる機能。
- 実行単位にまとめたテストへテスト結果を登録する機能。
- 「保管したテストケース」と「テスト実行順序/テスト実行スクリプト」を分けて管理する機能。（下記図のように、別に管理することで、テスト実行の進捗に合わせて保管してあるテストを修正する必要がなくなりテストケースの保守性が向上する。）



ツールの効能

- テスト実装作業にて発生する以下のような問題を回避することができる。
  - テスト実行順序/テストスクリプトの管理が容易になる。テスト実行に割り振られたテストケースがどれであるかがバラバラのファイルで管理されるのではなく、一元化して参照できるので、担当者間で実行するテストが重複したり、漏れが出るといった問題が起こりづらくなる。
  - 保管したテストケース（テスト設計で作ったテスト）の保守性が高まる。テスト実行の計画に合わせて、保管したテストケースを実行回数分コピーしたり、実行順でソートし直してファイルを分割するといったことを行わなくなるため。
- 要件管理ツール、インシデント管理ツール、テスト自動実行ツールと連携することで、以下のことが自動化できる。
  - 要件カバレッジ（テストが要件をどの程度満たしているか）の計測の自動化。
  - 要件やテストケース毎の欠陥の傾向分析。
  - テスト結果の自動収集。（エビデンスの自動取得）

ツール使用上の注意

- テストケースの記載ルールについて関係者間で合意をすること。ツールは個人によりフォーマットが異なるという問題は回避できるが、スプレットシートのように個人の自由で列を追加したりできないため、テストケースが書きづらいという問題が起き、ツールを利用しなくなることがある。
- テストケースのコピー＆ペーストはほどほどにすること。開発したテストケースと実行するテストケースを分離することで保守性が高まるがその意味を理解せずにコピーを繰り返すと、スプレットシートを使うのと何も変わらないため効果が出ない。

コード解析

静的解析ツール

ツールの概要

ソースコードを解析し、ソースコード上に存在するさまざまな問題を、プログラムを実行せずに発見、またメトリクス情報の収集を行うツールです。

静的解析ツールでの確認方法は大きく以下の 3 種類に大別できます。

- ルールチェック。（問題だと思われるコーディングのパターンに該当するかを確認。）
- 制御フローやデータフローの解析。（コードの最初から最後までのパスを検出し、一連のフローを確認。）
- メトリクスの収集と分析。（コードの行数や複雑度といったメトリクスを分析し、閾値をこえる係数になるかを確認。）

ツールの効能

- コーディングルールの遵守確認。（例　SEC、MISRA）
- コードクローンの検出。
- メモリリークや、配列のオーバーラン、デッドロックといったランタイムエラーの検出。
- バッファオーバーフローなどセキュリティ脆弱性を生むプログラミングエラーの検出。
- ソースコードの行数やプログラムの複雑度など、メトリクス情報の取得し、欠陥が発生しやすい箇所の特定と改善に役立てる。

ツール使用上の注意

- 静的解析の実行は、開発作業におけるコーディング段階や欠陥修正のためのコード変更後に早期に行うこと。ツールで検出した欠陥は、レビューされ、必要に応じて修正されるべきであるが、開発の終盤で初めて静的解析を実行すると、見つかった欠陥の多くが残された工数では対処しきれずに修正されないか、もしくは欠陥を修正するためにプロジェクトのスケジュール変更を余儀なくされることが多々ある。
- 静的解析ツールが持つルールチェックをすべて適用しないこと。ツールにはさまざまなエラーを発見する機能があるが、あまりに発見される欠陥数が多すぎる場合は、開発担当者へ与える負担が大きくなりすぎ、結果としてツールの解析結果が活用されないという事態に陥ることがある。
- ツールから得られる結果をどのように活用するのか、以下に示すような観点からの計画を行うこと。（やみくもに静的解析ツールを利用しても効果が出ない。）
    - 用途（コーディングルールの遵守確認か、ランタイムエラーの発見か？）の確認。
    - ツールの選定。
    - 運用ルールの確認。（開発プロセスのどの段階で、どれ位の頻度で利用するのか？解析結果の利用者は誰で、どのように活用するのか？など）
    - 解析結果を利用してソースコードを変更する場合のワークロードはどれ位かかるのか？
    - 利用者はツールを利用するために十分な知識・経験があるか？

構造解析ツール

ツールの概要

ソースコードを解析し、ソフトウェアの構造をマトリクスやグラフなどにより可視化する。また、複雑度など、メトリクス情報の収集を行うツールです。

ツールの効能

- 機能拡張や欠陥修正など、既存のソフトウェアに対して変更を行う際に、設計資料が不十分な場合、ツールの解析結果を用い、ソフトウェアの構造の現状理解に役立つ。
- コーディング段階において、設計通りにコーディングされているか（例えば、意図しないコンポーネント間の依存関係が発生していないかなど）を確認できる。
- ソースコードの行数やプログラムの複雑度、コンポーネント間の依存関係といったメトリクス情報を取得し、欠陥を検出しやすい箇所を特定できる。

ツール使用上の注意

- 誰が・いつ・どのような結果を得るために構造解析ツールを使用するのか明確化すること。用途・効能によって選択するツールが異なる。

- 大規模なソフトウェアの構造解析を行う場合、使用するツールによっては、解析に必要なシステムリソースが確保できない、解析は可能だが非常に大きく見づらいグラフができてしまい結果として役に立たないなどの問題が発生することがある。そのような場合には、少ないシステムリソースで解析ができるような解析オプションの調整や、解析対象の範囲の調整が必要になる。

テスト（自動）実行

ユニットテストツール

ツールの概要

ソフトウェア開発の主にユニットテストにて実行するテストを支援するためのツールです。テスト対象を実行させるためのテストドライバ（テスト対象を呼び出し、入力値を与えて機能を実行させる上位コンポーネントやシステムの代わりとなる）と、その補助ツール（テストコードを容易に書くためのライブラリやフレームワークなど）で構成されています。

期待値と実行時のテストデータを比較してテスト結果の合否判定を自動的に行います。

ツールの効能

- テスト実行を自動化することができ、以下のようなテストを繰り返す際の工数を削減することができる。
  - ユニットテスト以降の工程で発生したインシデントに対応した際に、容易に再テストができるため、デグレード（すでにテスト済みの箇所が欠陥となること）を抑制することができる。
  - 手動では実行することが困難な量のテストを短時間で実行できるため、テスト対象の信頼性を上げることができる。（例えば、大量にテストすることで内部構造の境界に対するテストができるなど。）
- 複雑な手順のテストをミスなく実行することができる。
- デバッガと組み合わせて用いることで、デバッグ作業の効率を高めることができる。

ツール使用上の注意

- テスト実装工数を適切に確保すること。テストコード作成工数に加え、スタブ/テストデータの準備に対する考慮が必要となる。（工数を少なく見積もり、自動テストの実行に間に合わなくなる。）
- 期待値が明確でない仕様の対処を明確にすること。テストコードを書く時に期待値がかけないことで想定以上にテストコードの実装工数がかかってしまうことがある。
- ユニットテストの範囲はテスト計画～テスト設計作業で決めておくこと。決めておかないと以下のような問題が発生する。
  - 過剰にテストコードを書いてしまって工数が超過する。
  - 局所的なテストコードだけを書きテスト対象の品質を十分に確認できない。
  - 統合テストとの責務配分が曖昧で、双方のテストレベル間でテストケースの重複が発生したり、どちらでもテストせず確認漏れとなる。
- テストドライバ用のテストコードを構成管理し、継続的に利用できるようテストコードを保守すること。保守をしていないとテスト対象のバージョンアップに伴いテストコードが動かなくなり、無駄になる。

ユニットテストツール(組込み)

ツールの概要

組込みソフトウェア開発用のユニットテストツールは、前述した一般的なユニットテストツールの解説に加え、以下のことの考慮したツールになることが多くなります。

- テストデータ（テストベクタ）を大量に必要とするため、テストデータジェネレータをセットで使ったユニットテストの自動実行を行うツールが多くなります。
- スタブだけでなくシミュレータをセットで使ってユニットテストの自動実行を行うことが多くなります。例えばICE（インサーキットエミュレータ）によりターゲット環境上で行うものや、ISS（インストラクションセットシミュレータ）を用いたシミュレーション環境上で行うものなどがあります。

ツールの効能

- 組込みソフトウェア開発におけるユニットテストツールの効能は、「ユニットテストツール」と同様となる。

ツール使用上の注意

- 組込みソフトウェア開発特有のユニットテストツール使用上の注意としては、以下のことがある。
  - ユニットテスト実行前に、MPU、OS、コンパイラ、ターゲット環境などに依存する制約条件を確認しておかないと、ユニットテスト自体が実行できなかったり、テスト結果の精度を確保できなくなる。

キャプチャ/リプレイツール

ツールの概要

テスト対象となるシステムやコンポーネントのユーザインターフェイス（キーボードやマウスなど）からの操作を記録（キャプチャ）し、テストコード（スクリプトと呼ぶことが多い）として保存します。このスクリプトを再生（リプレイ）し、テスト対象を記録時と同じように動作させることが可能になるツールです。

ユーザインターフェイス（例えば、画面など）を記録し再生するメカニズムは以下の 3 種類に大別できます。

- XY 座標でどこをマウスがクリックしているか/キー入力しているかを記録、再生。
- ユーザインターフェイス上の要素（ボタン、入力フィールドなど）のプロパティを読み込み、一意に認識したものの操作を記録、再生。
- ユーザインターフェイス上の要素を OCR や画像認識といった技術で読み込み、一意に認識したものの操作を記録、再生。

操作のキャプチャ（記録）と操作のリプレイ（再生）以外に以下の機能があります。

- 期待結果通りに動作していることを判定する「チェックポイント」を予め設定し、実行時の結果と比較検証することができる。結果は、合格、不合格のようなログとして記録される。
- スクリプトをモジュール化し、モジュールを組み合わせて複数のテストシナリオに対応させる。
- テスト入力と期待結果を CSV ファイルやスプレッドシートに格納し、格納した全テストデータをスクリプト上で変数化した値として代入することで、ひとつのスクリプトを繰り返すだけで全テストを実行することができる。（このスクリプト作成技術は「データ駆動テスト」と呼ばれている。）

ツールの効能

- 以下のような同じテストを複数回実行する際の工数を削減できる。
  - ユーザインターフェイスが既存と同じところのテスト実行を自動化。（回帰テスト効率化）
  - 仕様変更に対して副作用を確認するためのテスト実行を自動化。（回帰テスト効率化）
  - テストデータの作成作業を自動化。
- テスト結果のエビデンス取得にかかる工数を削減できる。
- 複雑な操作を記録させることで、テスト実行時の操作ミスを防ぐことができる。

ツール使用上の注意

- テスト実装工数を適切に確保すること。（工数を少なく見積もり、自動テストの実行に間に合わなくなる）
- 期待値が明確でない仕様の対処を明確にすること。テスト実装時に期待値がかけないことで想定以上にスクリプトの実装工数がかかってしまうことがある。
- スクリプトは再利用したり、モジュール化を考慮すること。テストテストシナリオとスクリプトを1：1で作成するとシナリオ分のスクリプト作成工数が必要となる。
- ツールの特性を理解すること。
  - チェックポイントの選択。（例えば、画像間の比較検証はピクセルが合わず NG となる場合がある。）
  - ユーザインターフェイス要素の認識。（テスト対象の仕様や利用技術により認識できないことがある。）

性能テストツール

ツールの概要

システムに負荷を与え、そのシステムを構成するソフトウェアやハードウェアが期待通りの性能(応答性能、可用性、拡張性)を満たしているか検証するツールです。

性能テストでは、例えば Web 上でのネット販売サイトのテストにて、1000 人単位の同時アクセスをした時の応答をテストしなければならないことがあります。そのテスト実行を人手でのみすることは体制や環境において現実的ではありません。性能テストツールはこのような大量の負荷を容易かつ正確にシミュレートし、性能テストを行います。

性能テストは、クライアント層とサーバ層とのやり取り（プロトコル）をキャプチャ（記録）して、リプレイ（再生）するのが一般的です。ツールがどのプロトコルの記録に対応しているかによって、さまざまなシステム（Web アプリケーション、クライアント/サーバやシンクライアント、スマートフォンなど）の負荷テストを支援できます。

性能テストツールは、大量の負荷のシミュレート以外に以下の機能があります。

- キャプチャ/リプレイツールと同じ類の機能、例えばチェックポイントや、データ駆動テストといった機能がある。
- 応答時間を分析する機能。例えばリアルタイムに応答時間をグラフにプロットして表示したり、応答結果の標準偏差を計算したりする機能を搭載しているものが多い。

ツールの効能

- テスト対象のシステムやコンポーネントのレスポンスやスループットの確認ができる。
- オペレーティングシステムやミドルウェア、もしくはハードウェアのサイジングが適切かを確認できる。
- 高負荷時や並列動作時に発生するインシデント（欠陥）の検出ができる。

ツール使用上の注意

- 負荷量だけでなく、負荷を生成する条件(タイミングや使用データ、ビジネスフローなど)を想定している実運用を考慮したものにすること。これらが想定している実運用時と異なると正しい結果が得られない。
- レスポンスコードやサーバのログだけでなく、実際に受信するコンテンツを検証対象とすること。実際に受信するコンテンツを検証対象としないとアプリケーションエラーを見逃してしまう場合がある。
- テスト対象システムのアーキテクチャに詳しいメンバーの手助けが得られるかを事前に確認すること。負荷テストを行った結果、どこにボトルネックがあるのか分析したり、どのようにチューニングしたらよいかといった部分ではテスト対象システムのアーキテクチャに詳しいメンバーの手助けが必要となる。

セキュリティテストツール

ツールの概要

テスト対象が脆弱性の課題に対処できていることをテストするツールです。

セキュリティテストでは、テスト対象のシステム/コンポーネントの脆弱性に関わる欠陥が無いことを確認します。代表的な脆弱性の欠陥としては、SQL インジェクション、クロスサイトスクリプティング、および、バッファオーバフローなどがあげられます。

脆弱性の課題は、新しいセキュリティの突破方法の発見によって日々増えていきます。そのため、これまでは欠陥ではなかったことが新しい課題の発見により欠陥として対処しなければならなくなります。脆弱性をテストするには日々増えていく脆弱性に関わる欠陥に対する高い専門性が求められます。また、新しい脆弱性の課題が発見されるたびに、テスト済みのすべての機能に対して新しい脆弱性に関わる欠陥が無いかをテストしなければならないため、手動テストだけでは工数面から対処が困難になります。セキュリティテストツールは、テスト対象が最新の脆弱性の課題に対処できていることをすべての機能に対して詳細に確認することができます。

ツールによっては、単にセキュリティテストを自動実行するだけでなく、発見した脆弱性に対する修正案の提示や検証結果のレポート作成（業界基準やコンプライアンスにも対応していることの報告のため）を支援刷る機能がついているものもあります。

ツールの効能

- 人間では確認しきれない多くの脆弱性の課題を自動的に検証できるため、網羅性の確保や効率化につながる。
- 最新の脆弱性に対応したセキュリティルールファイルを利用することで最新の脆弱性に継続的に対応可能となる。
- 業界基準やコンプライアンスに対応しているエビデンスとなる。

ツール使用上の注意

- コード修正の手戻りを防ぐため、開発段階でのコード解析によるセキュリティテストの実行を検討すること。
- 最新のセキュリティルールファイルを使用すること。セキュリティルールファイルが古いと、ツールを使ってテスト実行してもセキュリティの欠陥が取り除けない。

テスト自動実行支援ツール

ツールの概要

ユニットテストツールやキャプチャ/リプレイツールにて作成した自動テストの実行やテスト結果の収集を制御するツールです。自動テストの実行をリモート実行（複数の自動テストを予め指定した環境上でスタートさせ、テスト結果を自動的に収集）します。

自動テストをスタートさせる方法は、タイマー予約（ある時間になったらスタート）する方法や、特定のイベントをきっかけにスタート（例えば、ソースコードのコンパイルなど）させる方法などがあります。また、複数の自動テストの実行順に条件分岐や繰り返し処理といった制御処理をしたり、エラーが出たときの対処をしたりといった機能を持ったものもあります。

自動テストの実行支援だけでなく、他の開発作業との連携を効率化させる機能を持ったツールもあります。（例えば、ソースコードのチェックアウト→コンパイル→デプロイ→テスト実行→テスト結果収集→各種メトリクス分析といった一連の開発作業を連携させて自動実行させる技術である、「継続的インテグレーション」を実現するツールが該当します。）

ツールの効能

- 夜間、休日といった時間に自動テストを実行し続けることができる。
- 複数の環境で多くの自動テストを実行するのを１人で行うことができる。
- 目的別に複数の自動テストをまとめて管理できる。（例えば、「回帰テスト用のセット」としていくつものテストを常に実行するなど。）

ツール使用上の注意

- 連続して実行する自動テスト間の関係を考慮すること。例えば以下のような問題が起きることがある。
  - 最初に実行した自動テストによりテスト対象の状態が変わってしまい、次の自動テストが途中で止まってしまう。この場合は、複数の自動テストの間に、状態を元に戻すための処理スクリプトを入れる。
  - テスト対象に欠陥が見つかったために想定外の状態遷移/画面遷移を起こして自動テストが止まる。この場合はエラーが起きたときの対処をするスクリプトを入れる。
  - 期待結果の格納先が同一になっているために前のテストの結果を次のテストが上書きしてしまう。
- 複数の自動テストを実行し続けることでテスト対象に想定外の負荷がかかり、テスト対象の処理が遅くなってしまい自動テストのスピードについていけなくなり、テストが止まることがある。この場合は、テスト実行のスピードを遅くしたり、テスト対象の処理を待つための同期処理スクリプトを入れたりする。

テストウェア管理

構成管理ツール

ツールの概要

ソフトウェア開発で作成するドキュメント類やソースコード、データファイルなど、一連の成果物を管理するためのツールです。

構成管理ツールは単にファイル自体を管理するだけのファイルサーバとは異なり、ファイルの変更内容や編集者などの履歴情報も管理しており、いつ・誰が・どのような変更を行ったかという情報を確認することができます。

また、構成管理ツールを適切に利用することで、複数の開発担当者による分散開発でのデグレードや衝突、あるいは複数のバージョンのシステムを並行して開発する際のマージ漏れなどを抑えることができます。

近年では、単一のリポジトリを利用する構成管理ツールの他に、複数のリポジトリを各拠点に設置して利用する分散構成管理ツール（分散バージョン管理ツール）と呼ばれるツールも提供されています。

ツールの効能

- 特定のバージョンにおけるソースコードやドキュメントを特定して、取得することができる。
- ソースコードやドキュメントの変更履歴や変更理由を継続的に残すことができる。
- 誤ってファイルを削除した場合や、誤編集した場合でも、容易に以前の内容を復元することができる。

ツール使用上の注意

- 構成管理ツールへ登録する成果物の構成は予め検討しておくこと。思いつきで構成管理ツールへ成果物を登録していると、その構成がよくわからなくなり、管理が困難になる。
- 成果物を構成管理ツールへ登録する際には、成果物の品質レベル（ビルドが成功する、テストが合格するなど）について、利用者間で認識を合わせておくこと。
- マージ漏れやマージミスといった問題が起きやすいため、ブランチの作成やマージを行う場合には、一定のルールを設けておくこと。

テスト結果管理

テスト結果管理/テスト結果レポートツール

ツールの概要

テスト実行結果を収集し、テスト実行の状況を把握、および品質分析や進捗報告などのレポートを作成するツールです。

各種グラフ（信頼度成長曲線、要件カバレッジ、日程毎の進捗など）や報告書（インシデント一覧、テスト結果一覧）などを自動作成します。これにより、テストを終わらせてよいのか、未実行のテストはどれなのかを容易に判断できます。

状況の把握としては主に以下の方法があります。

- 日程計画に対してどのテストが合格しているか、合格していないテストがどれかを特定する。
- テスト対象の要件、仕様に対してどのテストが合格しているか/していないかを特定する。
- インシデントの発生状況とテストの合格状況を組み合わせてテストが足りているかを判断する。

ツールの効能

- 静的解析ツール、テスト自動実行ツール、テストケース管理ツールやインシデント管理ツールなどとセットでテスト結果管理/テスト結果レポートツールを使うと、以下のようなテスト結果管理作業の工数削減につながる。
  - テスト結果状況がリアルタイムで把握できるようになり、どのテストを実行しなければならないかが容易に把握できる。
  - テスト結果の状況把握に工数がかからなくなる。
  - レポート作成が自動でできるようになり、常に進捗や品質を把握して作業ができる。
  - レポート作成に工数がかからなくなる。

ツール使用上の注意

- テストケースの管理/インシデントの管理ができていること。（テストケース管理/インシデント管理ができていないと集計した結果が有用なものにならない可能性がある。）
- どのようなレポートを作ると状況の分析や判断が容易になるかは別途検討を行うこと。

インシデント管理

インシデント管理ツール

ツールの概要

インシデントに対し、事象やステータス、調査すべきメンバー、期日や対処方法などを記録、蓄積、整理するためのツールです。

インシデント管理は、スプレッドシートやメールにて管理するといった方法もありますが、そのような管理には以下のような課題があります。

- 複数人で同じインシデントに対して情報を登録、編集することが困難である。
- インシデント数が増えると一覧性が極端に落ちる。
- 必須の入力項目を設定することが容易にはできない。

これらの不都合が生じると、結果として対応漏れが起きてしまいます。インシデント管理ツールは、このような不都合に対処するためのツールとなります。

また、収集されたインシデントの属性情報を分析する機能を併せ持つものも多くあります。またテスト結果管理/レポートツールと併用することで、進捗の把握、品質の分析、開発プロセスのさまざまな側面の改善に役立つ分析結果を生成できます。

ツールの効能

- インシデントの記録、蓄積、整理などの管理および対応状況の追って調べる作業を省力化できる。
- テスト自動実行ツールや静的解析ツールと併用することで、インシデントの記録作業を自動化できる。
- 登録時や対応時に付属、または記入されるデータを分析する作業を自動化できる。また、テスト結果管理ツールと併用することで、テスト結果とあわせた分析結果を自動生成できる。
- 膨大なインシデント数の管理が可能となる。
    - 重要度や優先度、カテゴリやリリース時期などのフィールドをキーに柔軟にアイテムの切り分け、および切り分け条件の保存や共有が可能となる機能を搭載していることが多い。
    - データベースシステムであるため、データベースの能力で何千万件も登録が可能となる。そのため、組織全部のインシデントを一元管理し、傾向分析を行うといった活用ができる。

ツール使用上の注意

- ツール利用前にステータス毎の担当者を明確にすること。そうしないと登録されたインシデントに対して誰が対応するかが不明になり放置されるといった問題が起きる。
- インシデントに付与する情報を増やしすぎないこと。インシデント管理ツールに登録するための工数が増えてしまい、ツールを利用しなくなることがある。
- フィールドの値の意味（重要度や欠陥原因分析など）について関係者間で合意しておくこと。そうしないと、情報が入力されず蓄積されていかなかったり、入力内容に誤解が多く分析に利用できないといった問題が起きる。

第五章　テストツールマップ

本章の内容

第六章では、非常に多岐にわたるテストツールが網羅的に紹介されています。この関係で、目的のツールを機能レベルから検索する場合に不便を感じます。この章では、左記の問題を解決するため、機能レベルで、第六章に掲載されているツールを表形式で分類することで、検索の利便性を提供しています。第六章を参照する前に、本章で、必要とするツールの機能を絞り込むことで、効率よく目的のツールを参照することができます。

この表の見方

本章で使用しているツールの分類基準及び、略号は以下の通りです。

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM

上の表で、

- 2 行目は、三章で紹介されているテスト作業の分類です。
- 3 行目は、四章で紹介されているテストツールの分類です。
- 4 行目は、本章で使用している 3 行目のツール分類の略号です。

各ツール名の行に、〇印がついていれば、そのツールが、四章で紹介したテストツール分類の持つ機能をカバーしていることになります。

テストツールマップ

テスト分析

要件管理ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
Caliber RM® / Caliber RDM™	○																					有	マイクロフォーカス
HP QualityCenter/ApplicationLifecycleManagment	○										○							○	○	○	○	有	ヒューレットパッカード
IBM Collaborative Lifecycle Management – RRC / RTC / RQM	○									○	○							○	○	○	○	有	アイビーエム
IBM Rational® DOORS®	○																					有	アイビーエム
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
Oracle Test Manager	○										○								○	○	○	有	オラクル
T-VEC	○	○	○				○	○			○									○		有	富士設備
Microsoft(r) Visual Studio(r) Team Foundation Server (TFS)	○					○				○	○								○	○	○	有	マイクロソフト
TestLink	○										○								○	○		無	オープンソース
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Professional with MSDN(r)	○										○			○				○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Test Professional with MSDN(r)	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
Xupper Ⅱ /TI Ver9.0	○						○				○								○	○		有	ケンシステム

テスト設計

状態遷移テストツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有／無	
StateMatrix		○																				無	オープンソース
T-VEC	○	○	○				○	○			○									○		有	富士設備

組み合わせテスト支援ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有／無	
PICT			○																			無	オープンソース
T-VEC	○	○	○				○	○			○									○		有	富士設備
カバレッジマスターwinAMS			○			○		○	○		○			○				○		○		有	ガイオテクノロジー

原因結果グラフ作成支援ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有／無	
CEGTEST				○																		無	オープンソース

動的解析ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
Coverity Dynamic Analysis					○																	有	コベリティ
DevPartner Java					○	○						○										有	マイクロフォーカス
DevPartner® Studio Professional Edition					○	○						○										有	マイクロフォーカス
Eclipse TPTP					○									○						○		無	オープンソース
ENdoSnipe Ver4.6					○											○						有	アクロクエスト
HP Diagnostics					○																	有	ヒューレットパッカード
IBM Rational® Application Developer					○	○						○		○								有	アイビーエム
IBM Rational® AppScan®					○							○					○					有	アイビーエム
IBM Rational® Functional Tester					○										○			○				有	アイビーエム
IBM Rational® Performance Tester					○											○						有	アイビーエム
IBM Rational® PurifyPlus™					○	○								○								有	アイビーエム
IBM Rational® Rhapsody®					○	○			○			○		○								有	アイビーエム
IBM Rational® Service Tester for SOA Quality					○				○					○								有	アイビーエム
IBM Rational® Software Architect					○	○						○		○								有	アイビーエム
IBM Rational® Test Virtualization Server					○				○					○	○	○		○				有	アイビーエム
IBM Rational® Test Workbench					○				○					○	○	○		○				有	アイビーエム
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
MC-Checker					○				○		○			○				○		○		有	ガイオテクノロジー
Oracle JRockit Mission Control					○											○						有	オラクル
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Insure++ 7.1					○	○																有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト

カバレッジ計測ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
DevPartner Java					○	○						○										有	マイクロフォーカス
DevPartner® Studio Professional Edition					○	○						○										有	マイクロフォーカス
djUnit						○			○													無	オープンソース
IBM Rational® Application Developer					○	○						○		○								有	アイビーエム
IBM Rational® PurifyPlus™					○	○								○								有	アイビーエム
IBM Rational® Rhapsody®					○	○			○			○		○								有	アイビーエム
IBM Rational® Software Architect					○	○						○		○								有	アイビーエム
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Insure++ 7.1					○	○																有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
SilkCentral® Test Manager						○					○							○		○	○	有	マイクロフォーカス
Microsoft(r) Visual Studio(r) Team Foundation Server (TFS)	○					○				○	○								○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
カバレッジマスターwinAMS			○			○		○	○		○			○				○		○		有	ガイオテクノロジー

その他のテスト設計支援ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有／無	
T-VEC	○	○	○				○	○			○									○		有	富士設備
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Test Professional with MSDN(r)	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
XupperⅡ/TI Ver9.0	○						○				○								○	○		有	ケンシステム

テスト実装

テストデータジェネレータ

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
CasePlayer2								○				○	○									有	ガイオテクノロジー
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
Oracle Data Masking Pack								○														有	オラクル
Oracle Test Data Management Pack								○														有	オラクル
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
Parasoft Virtualize 9.3								○	○	○	○											有	テクマトリクス
T-VEC	○	○	○				○	○			○									○		有	富士設備
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
カバレッジマスターwinAMS			○			○		○	○		○			○				○		○		有	ガイオテクノロジー

シミュレータ/スタブ

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
djUnit						○			○													無	オープンソース
EasyMock									○													無	オープンソース
HP Service Virturaization									○													有	ヒューレットパッカード
IBM Rational® Rhapsody®					○	○			○			○		○								有	アイビーエム
IBM Rational® Service Tester for SOA Quality					○				○					○								有	アイビーエム
IBM Rational® Test Virtualization Server					○				○					○	○	○		○				有	アイビーエム
IBM Rational® Test Workbench					○				○					○	○	○		○				有	アイビーエム
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
MC-Checker					○				○		○			○				○		○		有	ガイオテクノロジー
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
Parasoft Virtualize 9.3								○	○	○	○											有	テクマトリクス
カバレッジマスターwinAMS			○			○		○	○		○			○				○		○		有	ガイオテクノロジー

ラボイメージ

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
IBM Collaborative Lifecycle Management – RRC / RTC / RQM	○									○	○							○	○	○	○	有	アイビーエム
Oracle VM										○												無	オラクル
Parasoft Virtualize 9.3								○	○	○	○											有	テクマトリクス
Microsoft(r) Visual Studio(r) Team Foundation Server (TFS)	○					○				○	○								○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Test Professional with MSDN(r)	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト

テストケース管理ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有／無	
HP QualityCenter/ApplicationLifecycleManagment	○										○							○	○	○	○	有	ヒューレットパッカード
IBM Collaborative Lifecycle Management – RRC / RTC / RQM	○									○	○							○	○	○	○	有	アイビーエム
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
MC-Checker					○				○		○			○				○		○		有	ガイオテクノロジー
Oracle Test Manager	○										○								○	○	○	有	オラクル
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
Parasoft Virtualize 9.3								○	○	○	○											有	テクマトリクス
SilkCentral® Test Manager						○					○							○		○	○	有	マイクロフォーカス
T-VEC	○	○	○				○	○			○									○		有	富士設備
Microsoft(r) Visual Studio(r) Team Foundation Server (TFS)	○					○				○	○								○	○	○	有	マイクロソフト
TestLink	○										○								○	○		無	オープンソース
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Professional with MSDN(r)	○										○			○				○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Test Professional with MSDN(r)	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
XupperⅡ/TI Ver9.0	○						○				○								○	○		有	ケンシステム
カバレッジマスターwinAMS			○			○		○	○		○			○				○		○		有	ガイオテクノロジー

コード解析

静的解析ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
CasePlayer2								○				○	○									有	ガイオテクノロジー
Coverity Static Analysis												○					○					有	コベリティ
DevPartner Java					○	○						○										有	マイクロフォーカス
DevPartner® Studio Professional Edition					○	○						○										有	マイクロフォーカス
FindBugs												○										無	オープンソース
HP FortifySCA												○					○					有	ヒューレットパッカード
IBM Rational® Application Developer					○	○						○		○								有	アイビーエム
IBM Rational® AppScan®					○							○					○					有	アイビーエム
IBM Rational® Rhapsody®					○	○			○			○		○								有	アイビーエム
IBM Rational® Software Architect					○	○						○		○								有	アイビーエム
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
PROMA-C DevNavi Ver2.0												○							○		○	有	アクロクエスト
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト

構造解析ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有／無	
CasePlayer2								○				○	○									有	ガイオテクノロジー
Coverity Architecture Analysis													○									有	コベリティ
Lattix 7.3.1													○									有	テクマトリクス
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
Understand 2.6													○									有	テクマトリクス
Visual Studio Premium with MSDN	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト

テスト自動実行

ユニットテストツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
Eclipse　TPTP					○									○						○		無	オープンソース
HP Service Test														○	○							有	ヒューレットパッカード
IBM Rational® Application Developer					○	○						○		○								有	アイビーエム
IBM Rational® Policy Tester®														○			○					有	アイビーエム
IBM Rational® PurifyPlus™					○	○								○								有	アイビーエム
IBM Rational® Rhapsody®					○	○			○			○		○								有	アイビーエム
IBM Rational® Service Tester for SOA Quality					○				○					○								有	アイビーエム
IBM Rational® Software Architect					○	○						○		○								有	アイビーエム
IBM Rational® Test Virtualization Server					○				○					○	○	○		○				有	アイビーエム
IBM Rational® Test Workbench					○				○					○	○	○		○				有	アイビーエム
Junit														○								無	オープンソース
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
MC-Checker					○				○		○			○				○		○		有	ガイオテクノロジー
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Visual Studio Professional														○								有	マイクロソフト
Microsoft(r) Visual Studio(r) Professional with MSDN(r)	○										○			○				○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
カバレッジマスターwinAMS			○			○		○	○		○			○				○		○		有	ガイオテクノロジー

キャプチャ／リプレイツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有／無	
HP QuickTest Professional/ Functional Testing															○			○				有	ヒューレットパッカード
HP Service Test														○	○							有	ヒューレットパッカード
HP　Sprinter															○					○	○	有	ヒューレットパッカード
IBM Rational® Functional Tester					○										○			○				有	アイビーエム
IBM Rational® Test Virtualization Server					○				○					○	○	○		○				有	アイビーエム
IBM Rational® Test Workbench					○				○					○	○	○		○				有	アイビーエム
Oracle Application Replay Pack															○	○						有	オラクル
Oracle Functional Testing															○			○				有	オラクル
Oracle Real Application Testing															○	○						有	オラクル
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
Quality Commander															○	○		○				有	日本ノーベル
Selenium															○			○				無	オープンソース
SilkTest®															○			○				有	マイクロフォーカス
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Test Professional with MSDN(r)	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト

性能テストツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
Apache Jmeter																○						無	オープンソース
ENdoSnipe Ver4.6					○											○						有	アクロクエスト
HP LoadRunner/Performance Center																○						有	ヒューレットパッカード
IBM Rational® Performance Tester					○											○						有	アイビーエム
IBM Rational® Test Virtualization Server					○				○					○	○	○		○				有	アイビーエム
IBM Rational® Test Workbench					○				○					○	○	○		○				有	アイビーエム
Oracle Application Replay Pack															○	○						有	オラクル
Oracle JRockit Mission Control					○											○						有	オラクル
Oracle Load Testing																○						有	オラクル
Oracle Real Application Testing															○	○						有	オラクル
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
Quality Commander															○	○		○				有	日本ノーベル
SilkPerformer®																○						有	マイクロフォーカス
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト

セキュリティテストツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
Coverity Static Analysis												○					○					有	コベリティ
HP　FortifySCA												○					○					有	ヒューレットパッカード
HP　WebInspect																	○					有	ヒューレットパッカード
IBM Rational® AppScan®					○							○					○					有	アイビーエム
IBM Rational® Policy Tester®														○			○					有	アイビーエム
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
Microsoft(r) Visual Studio(r) Premium with MSDN(r)	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft(r) Visual Studio(r) Ultimate with MSDN(r)	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト

テスト自動実行支援ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
HP QualityCenter/ApplicationLifecycleManagment	○										○							○	○	○	○	有	ヒューレットパッカード
HP QuickTest Professional/ Functional Testing															○			○				有	ヒューレットパッカード
IBM Collaborative Lifecycle Management – RRC / RTC / RQM	○									○	○							○	○	○	○	有	アイビーエム
IBM Rational® Functional Tester					○										○			○				有	アイビーエム
IBM Rational® Test Virtualization Server					○				○					○	○	○		○				有	アイビーエム
IBM Rational® Test Workbench					○				○					○	○	○		○				有	アイビーエム
Jenkins																		○		○		無	オープンソース
MC-Checker					○				○		○			○				○		○		有	ガイオテクノロジー
Oracle Functional Testing															○			○				有	オラクル
Quality Commander															○	○		○				有	日本ノーベル
Selenium															○			○				無	オープンソース
SilkCentral® Test Manager						○					○							○		○	○	有	マイクロフォーカス
SilkTest®															○			○				有	マイクロフォーカス
Microsoft(r) Visual Studio(r) Premium with MSDN®	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Professional with MSDN®	○										○			○				○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Test Professional with MSDN®	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Ultimate with MSDN®	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
カバレッジマスターwinAMS			○			○		○	○		○			○				○		○		有	ガイオテクノロジー

テストウェア管理

構成管理ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
AccuRev 5.2.1																			○		○	有	テクマトリクス
HP QualityCenter/ApplicationLifecycleManagment	○										○							○	○	○	○	有	ヒューレットパッカード
IBM Collaborative Lifecycle Management – RRC / RTC / RQM	○									○	○							○	○	○	○	有	アイビーエム
Oracle Test Manager	○										○								○	○	○	有	オラクル
PROMA-C DevNavi Ver2.0												○							○		○	有	アクロクエスト
StarTeam																			○		○	有	マイクロフォーカス
Microsoft® Visual Studio® Team Foundation Server (TFS)	○					○				○	○								○	○	○	有	マイクロソフト
TestLink	○										○								○	○		無	オープンソース
Microsoft® Visual Studio® Premium with MSDN®	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Professional with MSDN®	○										○			○				○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Test Professional with MSDN®	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Ultimate with MSDN®	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
Xupper Ⅱ /TI Ver9.0	○						○				○								○	○		有	ケンシステム

テスト結果管理

テスト結果管理/テスト結果レポートツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
Coverity Integrity Control																				○		有	コベリティ
Eclipse TPTP					○									○						○		無	オープンソース
HP QualityCenter/ApplicationLifecycleManagment	○										○							○	○	○	○	有	ヒューレットパッカード
HP Sprinter															○					○	○	有	ヒューレットパッカード
IBM Collaborative Lifecycle Management – RRC / RTC / RQM	○									○	○							○	○	○	○	有	アイビーエム
Jenkins																		○		○		無	オープンソース
LDRA tool suite	○				○	○		○	○		○	○	○	○			○			○		有	富士設備
Mantis																				○	○	無	オープンソース
MC-Checker					○				○		○			○				○		○		有	ガイオテクノロジー
Oracle Test Manager	○										○								○	○	○	有	オラクル
Parasoft C++test 9.2					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft dotTEST 9.0					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft Jtest 9.1					○	○		○	○		○	○		○			○			○		有	テクマトリクス
Parasoft SOAtest 9.0					○			○	○		○	○			○	○	○			○		有	テクマトリクス
SilkCentral® Test Manager						○					○							○		○	○	有	マイクロフォーカス
T-VEC	○	○	○				○	○			○									○		有	富士設備
Microsoft® Visual Studio® Team Foundation Server (TFS)	○					○				○	○								○	○	○	有	マイクロソフト
TestLink	○										○								○	○		無	オープンソース
Microsoft® Visual Studio® Premium with MSDN®	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Professional with MSDN®	○										○			○				○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Test Professional with MSDN®	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Ultimate with MSDN®	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト
Xupper Ⅱ /TI Ver9.0	○						○				○								○	○		有	ケンシステム
カバレッジマスターwinAMS			○			○		○	○		○			○				○		○		有	ガイオテクノロジー

インシデント管理

インシデント管理ツール

	RM	ST	CT	CE	DA	CA	TD	TG	SM	LI	TM	SA	SR	UT	CR	PT	SC	AT	CM	RR	IM	有/無	
AccuRev 5.2.1																			○		○	有	テクマトリクス
HP QualityCenter/ApplicationLifecycleManagment	○										○							○	○	○	○	有	ヒューレットパッカード
HP　Sprinter															○					○	○	有	ヒューレットパッカード
IBM Collaborative Lifecycle Management – RRC / RTC / RQM	○									○	○							○	○	○	○	有	アイビーエム
IBM Rational® ClearQuest®																					○	有	アイビーエム
Mantis																				○	○	無	オープンソース
Oracle Test Manager	○										○								○	○	○	有	オラクル
PROMA-C DevNavi Ver2.0												○							○		○	有	アクロクエスト
SilkCentral® Test Manager						○					○							○		○	○	有	マイクロフォーカス
StarTeam																			○		○	有	マイクロフォーカス
Microsoft® Visual Studio® Team Foundation Server (TFS)	○					○				○	○								○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Premium with MSDN®	○				○	○	○	○			○	○	○	○	○		○	○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Professional with MSDN®	○										○			○				○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Test Professional with MSDN®	○						○			○	○				○			○	○	○	○	有	マイクロソフト
Microsoft® Visual Studio® Ultimate with MSDN®	○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○	有	マイクロソフト

第六章　テストツールカタログ

本章の内容

本章では、実際に入手可能なテストツールを紹介しています。本章で紹介するツールは、ASTER テストツール WG に参加されているツールベンダー、および代理店が提供しているツールを提供元自身で記載していただきました。オープンソースツールに関しては、ASTER テストツール WG のメンバー自身が利用経験があるツールを各ツールの種類毎ひとつ選択し、記載しました。

この表の見方

記載内容は以下のようになります。

対象ＯＳ	対象言語	動作形態	有償／無償
ツールが動作する OS	ツールが対象としているプログラム言語（言語に関係しないツールは「－」と記載。	動作形態(クライアントアプリケーション/システム/プラグイン/Web 上のサービス..etc.)	有償
提供元	URL		
日本でのツールの提供元	ツールを具体的に知るための情報が掲載されている Web サイト or ツールの配布をしている We サイト		
概要			
ツールの概要と、ツールのイメージを理解するための図			

各ツールがどのツールの種類に該当するか分かるように下記のような表に〇をつけています。各〇がついた機能についての詳細は併記した Web サイトで確認するか、各ツールベンダーにお問い合わせください。

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
該当する部分に〇がついています。（そのツールが複数のツール種類を兼ね備えている場合は複数の〇がつきます）																			

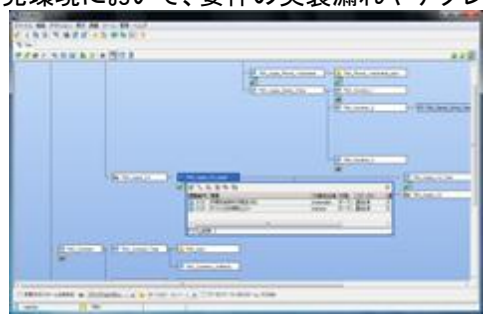
AccuRev 5.2.1

対象ＯＳ	対象言語	動作形態	有償／無償
Windows 7/Vista (SP1 以降)/XP (SP2 以降)/Server 2008 (SP1 以降)/Server 2003 (SP2 以降)、Linux (32bit/64bit、kernel 2.6 以上)	ー	クライアントアプリケーション、コマンドライン インターフェイス、Eclipse プラグイン、Visual Studio プラグイン、Web アプリケーション	有償

提供元	URL
テクマトリックス	http://www.techmatrix.co.jp/quality/accurev/index.html

概要

開発プロセスを可視化する、プロセス指向のソフトウェア構成管理・変更管理ツールです。分散/並行開発、アジャイルなどあらゆる開発モデルに適用でき、成果物管理と開発プロセスを融合します。統合された課題管理ツールと連携することで、課題とコード変更の間のトレーサビリティを実現し、派生開発や保守開発など複数の開発ラインが並行する開発環境において、要件の実装漏れやデグレードの防止に役立ちます。



開発プロセスを可視化する StreamBrowser

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
																		○		○

Apache JMeter

対象ＯＳ	対象言語	動作形態	有償／無償
―	―	Java アプリケーション(GUI)	無償
提供元	URL		
Apache Software Foundation	http://jmeter.apache.org//		

概要

Web アプリケーションを提供するサーバなどに対し「Test Plan」と呼ばれるシナリオファイルを記述することで、柔軟なパフォーマンステストが行える負荷テストツールです。テスト対象は Web サーバに限らず、データベースや FTP、LDAP サーバなどに対するテストもサポートしています。「Test Plan」の子となる「Thread Group」というくくりで同時アクセス数やループ数などを設定しながらテスト対象サーバの限界性能を探ることができます。

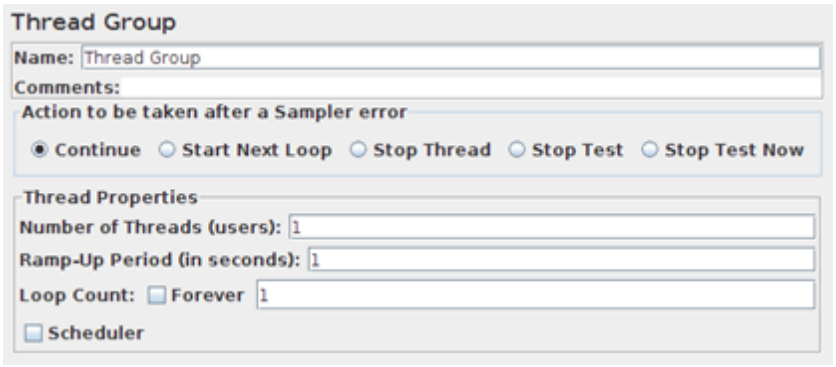


図　:「Thread Group」の設定画面

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計							テスト実装				コード解析		テスト実行				テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
															○					

Caliber RM® / Caliber RDM™

対象ＯＳ	対象言語	動作形態	有償／無償
サーバ：Windows Server 2003、2008 クライアント：Windows XP、Vista、7	（管理ツールのため開発言語に依存しない）	Client/Server クライアントは、Web、Windows、Eclipse/VisualStudio	有償

提供元	URL
マイクロフォーカス	http://www.microfocus.co.jp/products/caliber/caliber/

概要

Caliber 製品ファミリーは、詳細かつ正確な要件定義を行うことを可能にします。これにより、各部門をまたがるプロジェクトの関係者が、期限、予算、仕様通りにプロジェクトを進められるよう、効果的な協調作業が行えます。可視化されたシナリオを、エンドユーザーと IT 部門の間で共通の理解可能な簡単な表現方法で定義していくことが可能です。シナリオはストーリーボードとして実行できるので、関係者は何が抜けているかを把握することができ、要件を再定義し検証することにより、コストのかかる手戻りの発生を排除することができます。
要件の検証が終わると、テストケースや、UML による設計情報を自動生成することができるので、ソフトウェア開発の速度と精度が大幅に向上します。また、要件は開発プロジェクトの全工程を通じて管理および追跡可能なため、プロジェクトの進行をリスクにさらすことなく、絶えず変更される要件に対し、影響範囲を分析し迅速に対応することができます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○																				

CasePlayer2

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP, Windows Vista, Windows 7（32 ビット）	C/C++	クライアント（ノードロック）/サーバ（フローティング）	有償
提供元	URL		
ガイオ・テクノロジー株式会社	http://www.gaio.co.jp/product/dev_tools/pdt_caseplayer2.html		

概要

CasePlayer2 は、ANSI 準拠 C 言語、C++、組み込み向け C 言語（非 ANSI）、アセンブラのソースコードを解析し、フローチャートなどのプログラム仕様書を作成するツールです。「仕様書ブラウザ」を搭載し、ソースコードと各仕様書間の連携を自由に取ることができます。過去のソースコードのロジック解析や、新たに開発したプログラムソースの仕様書作成作業を強力に支援します。ソースコードの静的解析機能として、「外部変数」の参照/代入の一覧作成機能や、C 言語コーディング規約「MISRA-C」のルールチェック機能を搭載しています。単体テストツール「カバレッジマスターWinAMS」との連携によりブランチカバレジ、MC/DC カバレッジの最適化テストケースの自動作成をサポートします。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
							○				○	○								

CEGTEST

対象ＯＳ	対象言語	動作形態	有償／無償
ー	ー	Web アプリケーション	無償
提供元	URL		
加瀬　正樹氏	http://softest.cocolog-nifty.com/labo/CEGTest/		

概要

原因結果グラフ(Cause-Effect Graph)と呼ばれるテスト設計技法を利用したテスト実装が Web アプリケーション上で容易に行えるツールです。複雑な論理関係や制約を図で表すことで自動的にデシジョンテーブルに変換してくれます。この変換作業が現場への導入の高いハードルになっていた問題を一気に解決したツールです。

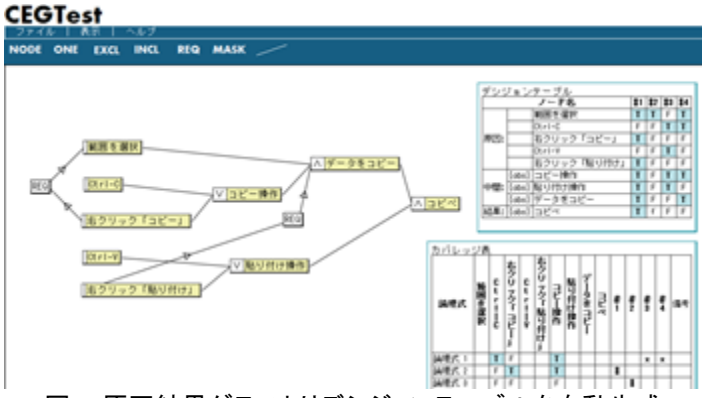


図　:原因結果グラフよりデシジョンテーブルを自動生成

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
			○																	

Coverity Architecture Analysis

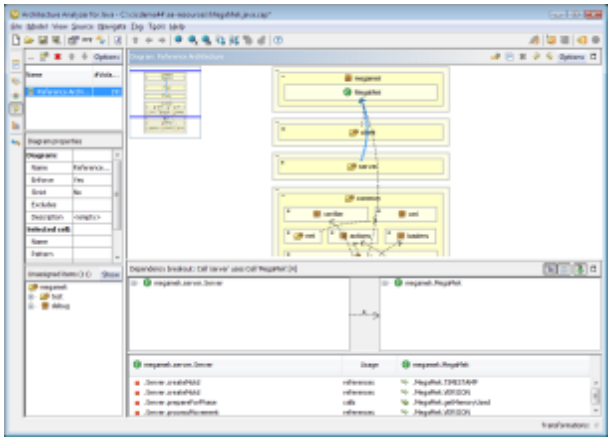
対象ＯＳ	対象言語	動作形態	有償／無償
Windows, Linux, Solaris , Mac OS X	C/C++, Java	スタンドアロンアプリケーション	有償

提供元	URL
コベリティ日本支社	http://www.coverity.com/html_ja/products/architecture-analysis.html

概要

Coverity Architecture Analysis は、大規模な C/C++および Java ソースコードのアーキテクチャ構造と依存関係を自動的に視覚化する構造解析ツールであり主な特徴は以下のとおりです。

- 設計文書が不十分なレガシーコードに対する開発者の理解を助ける
- 依存関係をチェックし、アーキテクトが定義した設計ルール違反を自動的に検出する
- コードの複雑度等のメトリクス情報を用いて、過度に複雑なコードの存在箇所を指摘する



ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
												○								

Coverity Dynamic Analysis

対象ＯＳ	対象言語	動作形態	有償／無償
解析エンジン:Windows, Linux, Solaris , Mac OS X 不具合管理サーバー:Windows, Linux	Java	解析エンジン：コマンドライン 不具合管理サーバー：Web システム	有償

提供元	URL
コベリティ日本支社	http://www.coverity.com/html_ja/products/dynamic-analysis.html

概要

Coverity Dynamic Analysis は Java プログラムにおけるデッドロックや競合状態の発生等、マルチスレッドアプリケーションにおける問題を効率的発見することができる動的解析ツールであり、以下を特徴としています。

- 現在使用しているテストプログラムを Coveirty の Java エージェント上で実行することにより解析する仕組みのため、テスト環境を殆ど変更せずに動作させることが可能
- 実際に発生した問題だけでなく、構造上発生しうるマルチスレッドに関する問題の検出が可能

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○																

Coverity Integrity Control

対象ＯＳ	対象言語	動作形態	有償／無償
:Windows, Linux	C/C++/C#/Java	Web システム	有償
提供元	URL		
コベリティ日本支社	http://www.coverity.com/html_ja/products/integrity-control.html		

概要

Coverity Integrity Control はコベリティの解析ツール群によるテスト結果を用いて、ソースコードの品質、セキュリティリスク、および開発効率を可視化し、ソフトウェア開発におけるリスクを低減するための統合ソリューションです。あらかじめ設定しておいた標準ポリシー（品質目標）用いて、ソースコードの品質や開発プロセスの状態の管理、監視、およびレポートを自動化します。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果「管理／レポート」ツール	インシデント管理ツール
																			○	

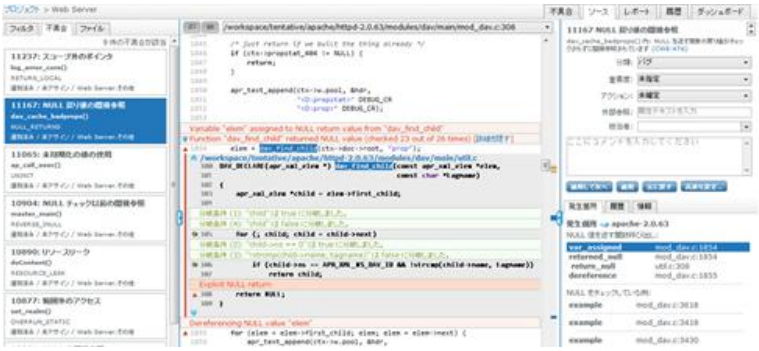
Coverity Static Analysis

対象ＯＳ	対象言語	動作形態	有償／無償
解析エンジン:Windows, Linux, AIX, Solaris , Mac OS X 等 不具合管理サーバー:Windows, Linux	C/C++, Java, C#	解析エンジン：コマンドライン、または、統合開発環境(MS Visual Studio, Eclipse へのプラグイン) 不具合管理サーバー：Web システム	有償

提供元	URL
コベリティ日本支社	http://www.coverity.com/html_ja/products/static-analysis.html

概要

Coverity Static Analysis はリソースリークやデッドロック、システムクラッシュ等、システム全体への重大な影響を及ぼす不具合を発見する静的解析ツールです。全パス解析により動的テストやコードレビューでは見つけにくい不具合を効率的に発見できるため、家電等の組み込み機器、社会インフラ、自動車、ゲーム、金融等、全世界1100社以上のお客様に品質向上、開発効率向上のためにご利用いただいております。



ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
											○					○				

DevPartner® Java Edition

対象ＯＳ	対象言語	動作形態	有償／無償
Windows Server2003 / 2008 WindowsXP / Vista / 7 / Solaris / AIX / HP-UX / RedHat	Java	ブラウザ/専用クライアント	有償

提供元	URL
マイクロフォーカス	http://www.microfocus.co.jp/products/devpartner/devpartner_fm/devpartnerjavaedition/

概要
DevPartner Java Edition は、Java アプリケーションのテストをサポートする、性能解析ツールです。J2EE/J2SE に対応し、静的ソースコード解析、パフォーマンス分析、メモリ分析、カバレッジ分析の 4 つの強力な機能により、ミッションクリティカルなシステムでの高品質化、開発期間の短縮、生産性の向上を実現します。

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
				○	○						○									

DevPartner® Studio Professional Edition

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008 WindowsXP/Vista/7	VB.NET、C#.NET、C++(マネージ)、ASP.NET C++(アンマネージ)	VisualStudio 2003 / 2005 / 2008 / 2010 のアドインとして提供	有償
提供元	URL		
マイクロフォーカス	http://www.microfocus.co.jp/products/devpartner/devpartner_fm/devpartnerstudiopro/		

概要

DevPartner Studio Professional Edition は、静的ソースコード解析、パフォーマンス分析、メモリ分析、カバレッジ分析、実行時エラー検出などで構成され、コーディングからテストまで各工程をトータルにサポートしています。また、これらの機能を利用すれば、各工程でのアプリケーション品質を測定・可視化し、テスト漏れの防止や品質の埋め込みを可能にします。これにより、「バグが少ない」「パフォーマンス劣化が少ない」「運用時において高い信頼性を保つ」といった高品質なアプリケーションを、効率よく開発することができます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○	○						○									

djUnit

対象ＯＳ	対象言語	動作形態	有償／無償
（Eclipse の実行環境に依存）	Java	Eclipse プラグイン、Ant	無償
提供元	URL		
株式会社デジック	http://works.dgic.co.jp/djwiki/Viewpage.do?pid=@646A556E6974		

概要

本ツールには、Java の単体テストにおけるカバレッジを計測する機能と、モックオブジェクトの生成をサポートする機能（Virtual Mock Objects）があります。カバレッジ計測機能では、実行されたテストによるステートメントカバレッジ、デシジョンカバレッジの値を計測し、グラフとして表します。また、テスト対象クラスの未実行の行をマーキングします。一方、Virtual Mock Objects 機能では、テスト対象クラス外のメソッドへの入力値に応じて戻り値を自由に変更するといった複雑なテストを容易に実行することができます。

図　djUnit によるカバレッジ計測結果：カバレッジの値と未実行の行が分かる

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
					○			○												

EasyMock

対象ＯＳ	対象言語	動作形態	有償／無償
Windows、Linux 等（JUnit の実行環境に依存）	Java	JUnit と合わせて動作	無償
提供元	URL		
	http://www.easymock.org/		

概要

JUnit を使用した Java の単体テストにおいて、擬似的なオブジェクト（モックオブジェクト）の作成をサポートするツールです。JUnit のテストコード中で EasyMock のライブラリを使用することで、モックオブジェクトを生成し、テスト対象クラス外のメソッドに引数を与えて、期待する戻り値が返ってくるように設定することができます。これにより、本来はサーバ上で動的に生成されるオブジェクトが必要なテストでも容易にテストすることができます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
								○												

Eclipse TPTP

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP、Windows Vista、Windows 7、Linux	Java	Eclipse プラグイン	無償
提供元	URL		
	http://www.eclipse.org/tptp/		

概要

一連の Java アプリケーションに対して、その各要素（クラス、メソッドなど）のパフォーマンス（性能）を測るツールです。アプリケーション実行にかかる時間や実行順序、回数などを測定、記録したり、メモリやスレッドの分析をしたりといった機能があります。また、GUI エディタを使用した JUnit テストコードの作成、編集や、リモートホストでのテストの実行、テスト結果のレポート作成などの機能も備えた統合的なテスト環境です。

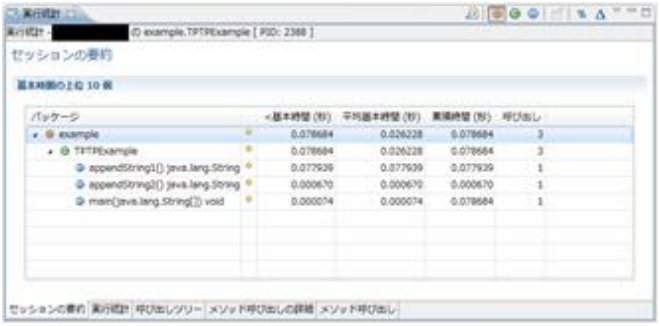


図　TPTP の実行結果：各メソッドの実行回数や実行時間が分かる

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○									○						○	

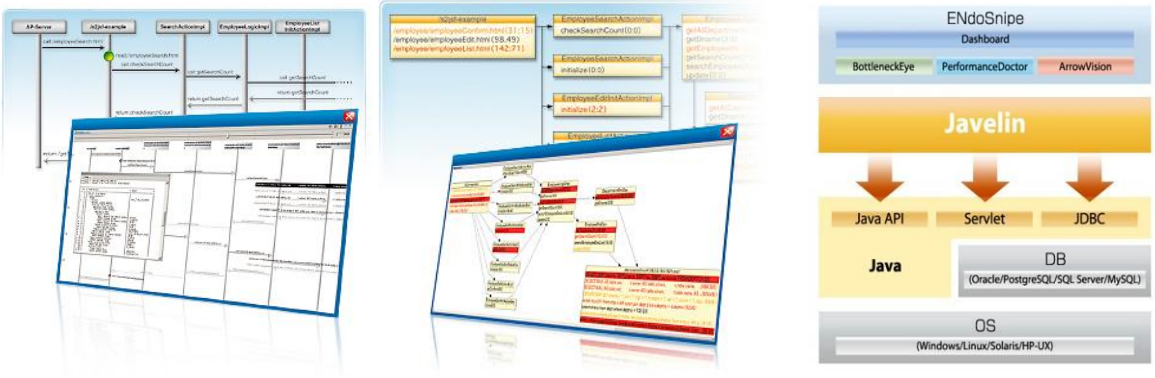
ENdoSnipe Ver4.6

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP/Vista/7 Windows Server Linux/Solaris/HP-UX	Java Oacle 10g R2 以上、PostgreSQL 8.0 以上、SQL Server 2005 以上、MySQL 5.0 以上	Eclipse プラグイン、Web ダッシュボード	有償

提供元	URL
Acroquest Technology	http://endosnipe.acroquest.co.jp/index.html

概要

ENdoSnipe は、Java システムの動作状況を診断し、システム内部に潜んでいるパフォーマンス問題や品質問題の原因を自動的に検出し、「見える化」するソフトウェアです。
そのため、パフォーマンス低下やシステムダウンの傾向などの品質問題を早期に解決することができます。



3つの見える化 「ボトルネックが見える」「システムの動作が見える」「性能問題の芽が見える」

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○											○					

FindBugs

対象ＯＳ	対象言語	動作形態	有償／無償
Windows、GNU/Linux 、MacOS X 等	Java	コマンドライン、独自 GUI、Ant、Eclipse プラグイン、Maven 等	無償
提供元	URL		
	http://findbugs.sourceforge.net/		

概要

Java 言語で記述されたソースコードに対して静的解析を行うツールです。Java のコーディングの慣習に違反しているためにバグを引き起こす可能性のある記述箇所を自動的に検出します。約 300 個のチェックルールがデフォルトで用意されている他、独自のチェックルールの作成もできます。性能、セキュリティ、マルチスレッドなどさまざまな観点のチェックルールがあり、各ルールはその重大度に応じて Error、Warning、Info の 3 段階のレベルが設定されています。

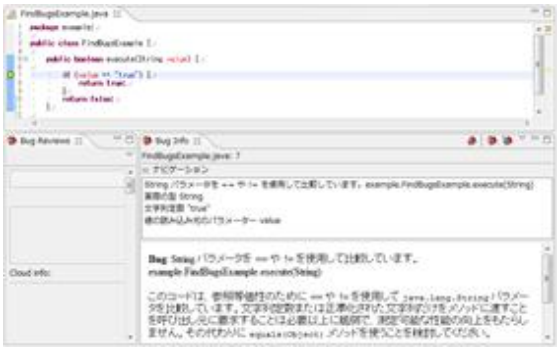


図　Eclipse 上での FindBugs の実行結果：チェックされた箇所と違反内容が分かる

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
											○									

HP Diagnostics

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008(Server) Windows/Unix/Linux(agent)	J2EE, .net, OracleDB, SQLServer, SAP, Websphere MQ, CICS	Web システム	有償
提供元	URL		
日本ヒューレットパッカード	http://www8.hp.com/jp/ja/software/software-product.html?compURI=tcm:191-936858&pageTitle=diagnostics-software		

概要

アプリケーションの分析を行うプロファイラツールです。監視対象側に Agent(probe)をインストールし、Diagnostics サーバで複数のアプリケーション監視を一元的に行うことが可能です。負荷が非常に低いため、テスト時/本番運用時共にご使用頂けます。メソッドの処理時間や呼び出し回数、CPU 使用率、メモリーリークを起こしているオブジェクトや処理時間のかかっている SQL 等分析することが可能です。負荷テストツールの HP LoadRunner や運用時のサービス監視ツールである HP BAC と連携させることも可能です。

図　リクエストのブレークダウン：メソッドの処理時間や引数、実行された SQL 等が分かる

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○																

HP Fortify SCA

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008(Server) Windows/Unix/Linux/Mac OS	ASP.NET、VB.NET、C# (.NET)、C/C++、Classic ASP (with VBScript)、COBOL、CFML、HTML、Java、JavaScript/AJAX、JSP、PHP、PL/SQL、Python、T-SQL、Visual Basic、VBScript、XML、ABAP/4	クライアントツール	有償

提供元	URL
日本ヒューレットパッカード	https://www.fortify.com/products/hpfssc/source-code-analyzer.html

概要

アプリケーションのソースコードを論理的に検査し、脆弱性を生み出すコーディングがないかを診断します。470 種類以上の脆弱性および品質に関わる問題点を発見し、その問題の解説、推奨の修正方法やプライオリティ情報を提供します。チェックパターンはインターネットを通じて定期的に更新しています。Web アプリ、PC アプリ、組み込み系など幅広い分野のアプリケーションの検査、また、Struts など外部ライブラリを使った実装方法のチェックも可能です。統合開発環境（IDE）へ組み込みや、API やコマンドラインを使った連携が可能なため、既存の開発環境に容易に適応できます。PCI-DSS, OWASP などのガイドラインに対応しておりコンプライアンス対策に有効です。

図　脆弱性の根本原因、脅威が発生する論理的なパスが分かる

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計							テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果「管理／レポート」ツール	インシデント管理ツール
											○					○				

HP LoadRunner/Performance Center

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008 WindowsXP/Vista/7	http/https, COM/DCOM, Citrix, java, .net, RemoteDesktop, SAP, Oracle EBS, Winsock 等(通信プロトコルに依存)	クライアントツール	有償

提供元	URL
日本ヒューレットパッカード	http://www8.hp.com/jp/ja/software/software-product.html?compURI=tcm:191-935779&pageTitle=loadrunner-software

概要

アプリケーションの操作/通信を記録し、多重実行する負荷テストツールです。負荷テストで必要とされるさまざまな機能が用意されています。スクリプト作成時間の短縮、実ユーザの操作に近い負荷の再現、エージェントレスでの統計情報の取得、分析/レポート作成を短縮する豊富な機能で負荷テスト全体の効率を上げることが可能になります。

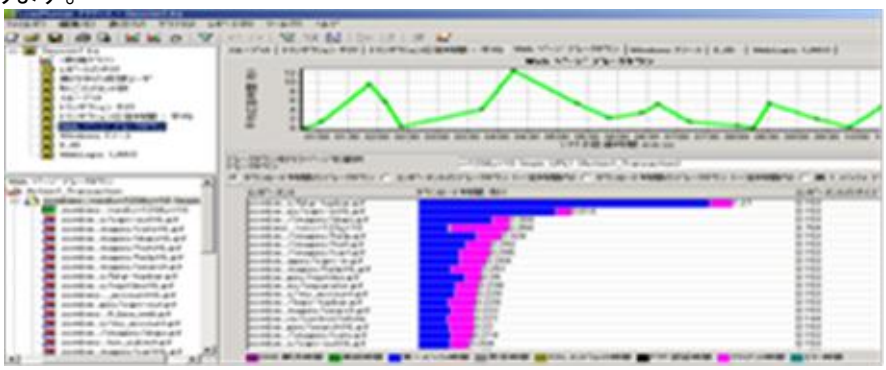


図　Web ページ分析：Web ページ内のどこで時間がかかっているかを把握できる

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
															○					

HP Quality Center/Application Lifecycle Managment

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008(Server) WindowsXP/Vista/7(Client)	（管理ツールのため開発言語に依存しない）	Web システム	有償
提供元	URL		
日本ヒューレットパッカード	http://www8.hp.com/jp/ja/software/software-product.html?compURI=tcm:191-936102		

概要

要件とテストと不具合を単一のリポジトリで管理する Web システムです。個々のツールとしての機能を備えつつ、要件とテストと不具合の関連付けにより要件カバレッジを把握したり、要件の変更時に影響するテストを分析したりすることができます。また、最上位エディション（製品名が ALM と変わります）では、複数プロジェクトにまたがるレポートを作成したり、他のプロジェクトのテスト資産を共有できます。また、開発ツール（ソースコード管理、ビルド管理、静的解析ツール）と連携して、品質を可視化します。

図　要件カバレッジ：要件に対してテストがどこまで合格しているかが分かる

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○										○							○	○	○	○

HP QuickTest Professional/ Functional Testing

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008 WindowsXP/Vista/7	Web, Windows API, VB, .net, Java, SAP, Oracle EBS, Powerbuilder, Delphi, ターミナルエミュレータ等	クライアントツール	有償

提供元	URL
日本ヒューレットパッカード	http://www8.hp.com/jp/ja/software/software-product.html?compURI=tcm:191-936981&pageTitle=functional-testing

概要

GUI 操作を記録し、再生する機能テストツールです。初心者にもわかりやすいコーディング不要のキーワードビューや上級者が高度にカスタマイズすることを可能とするエキスパートビューの両方を提供しているため、幅広い技術者にご使用頂けます。座標に依存しないオブジェクト単位でのスクリプトが作成できるため、画面変更やメンテナンスに強いスクリプトを作成できます。さらに豊富な種類のチェックポイント(テストの成否の判断)やデータ駆動テスト（図のようにスプレッドシートに直接　パラメータ化する入出力データを書き込める）、エラー発生時の回復処理等、自動テストで必要とされる多くの機能が提供されています。

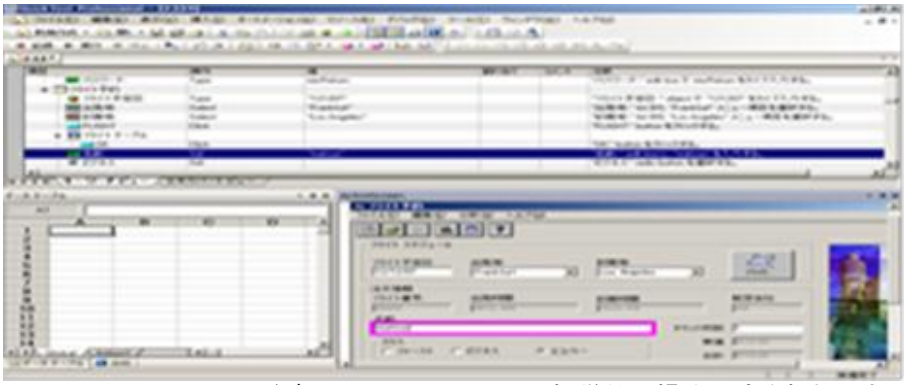


図　QTP 画面：キーワードビューと ActiveScreen で視覚的に操作内容を把握できる

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計							テスト実装				コード解析		テスト実行				テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
														○			○			

HP Service Test

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008 WindowsXP/Vista/7	WSDL、JMS、HTTP、REST 等	クライアントツール	有償
提供元	URL		
日本ヒューレットパッカード	http://www8.hp.com/jp/ja/software/software-product.html?compURI=tcm:191-937085&pageTitle=service-test-software		

概要

画面のないサービスコンポーネントの機能テストを行うツールです。直感的に使用できるように実行可能なアクティビティをツールボックスからテストフローにドラッグ&ドロップで実装し、テストが視覚的に構成されるため、コードを殆ど使用することなく、Web サービスのテストスクリプトを作成できます。また、上級者はイベントハンドラでカスタマイズすることができ初級者から上級者まで効率的にテストすることが可能です。

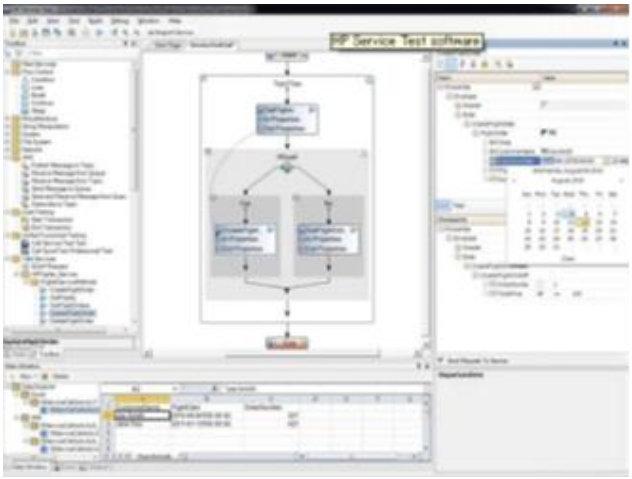


図　テストスクリプト:テストフローで視覚的にスクリプトの作成ができる

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
													○	○						

HP Service Virtualization

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008 WindowsXP/Vista/7	WSDL、JMS、HTTP、REST等	Web システム	有償
提供元	URL		
日本ヒューレットパッカード	http://www8.hp.com/jp/ja/software/software-product.html?compURI=tcm:191-1016176&pageTitle=service-virtualization		

概要

サービスを仮想環境としてレスポンスを返すシミュレーションツール(スタブ)です。現在多くの環境が複雑なシステムを構成しており、テスト環境作成時時にはアクセスが限定的されたサービスが存在したり、アクセス毎に課金が発生し必要とされる等、テストを容易に行えない状況があります。HP ServiceVirtualization は仮想的にリクエストに対する応答を生成し、これによりシステム全体を早期にテストすることが可能になります。また、応答時間も自由にカスタマイズすることもでき、負荷テストを想定した利用も可能です。

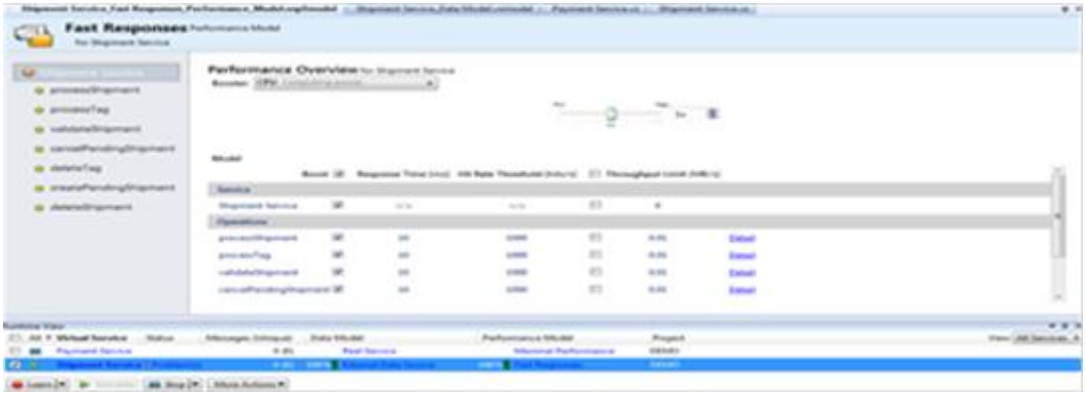


図　パフォーマンスモデル：応答時間も自由に設定できる

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
								○												

HP Sprinter

対象ＯＳ	対象言語	動作形態	有償／無償
WindowsXP/Vista/7	通常利用では特になし。パワーモード利用時(.net、java、IE、Firefox、SAP、WPF 等)	クライアントツール	有償

提供元	URL
日本ヒューレットパッカード	http://www8.hp.com/jp/ja/software/software-product.html?compURI=tcm:191-936957&pageTitle=sprinter-software

概要

手動テストの支援ツールです。テスト担当者は事前登録された作業指示に従ってテストを行い、インタラクティブにテストのステータスと結果を登録することができ、テストの品質を均一化します。また手動テストの大半の時間を占める、結果画面のエビデンス取得を効率よく行うことができます。さらにパワーモードでは画面に対するデータ入力作業、画面操作のマクロ記録、ログ記録、複数環境での同時実行作業を支援するミラーリング機能等を使用して作業の効率を高めることが可能です。

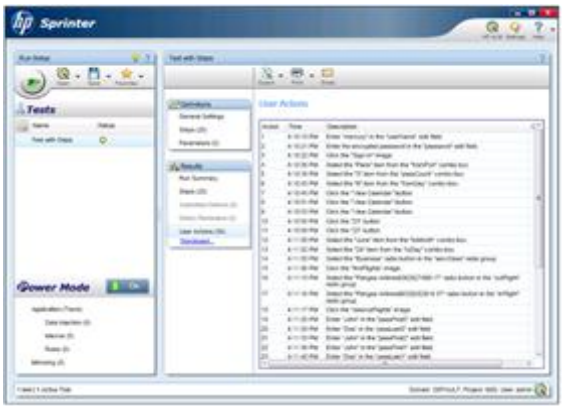


図　ログ記録：操作内容がログとして取得され、エクスポートすることも可能

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
														○					○	○

HP WebInspect

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008(Server) Windows	Web ページであれば開発言語に関係なく対応可能	クライアントアプリケーション	有償
提供元	URL		
日本ヒューレットパッカード	https://www.fortify.com/products/web_inspect.html		

概要

アプリケーションへ自動で疑似攻撃を実施・監視し、その反応から脆弱性を発見します。Web アプリケーションの構造を自動的に検出(高度なクロール機能)し、発見した脆弱性の重大度と対応方法を提示します。エグゼクティブサマリー、PCI-DSS 等、さまざまなレポート出力機能があり、コンプライアンス対策に有効です。
開発言語に依存せず、また最新の AJAX、Flash などのリッチな Web アプリケーション、Web サービスにも対応します。攻撃パターンはネットを通じて常に最新の状態で利用可能です。世界で大手企業を中心に1000社以上の導入実績があります。HP Fortify SCA と組み合わせ、より付加価値の高いハイブリッド検査が可能です。

図　発見した脆弱性を整理して表示する

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
																○				

IBM Collaborative Lifecycle Management – RRC / RTC / RQM

対象ＯＳ	対象言語	動作形態	有償／無償
RHEL/SLES/ Win2003/2008/2008R2/ IBM AIX/IBM i/IBM zOS/ Solaris 10 SPARC	（管理ツールのため開発言語に依存しない）	Web システム	有償

提供元	URL
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/announce/lifecycle/

概要

トレーサビリティ確保による品質の向上やリアルタイム・プランニングによる納期の短縮など、統合されたアプリケーション・ライフサイクル管理 (ALM) 機能によってソフトウェア開発チーム全体の生産性を高める統合ソリューション、コラボレーティブ・ライフサイクル・マネージメント（CLM）を提供します。

Rational® Requirements Composer （RRC)は、従来の製品にはなかった要求を”練り上げる場”を提供します。もちろん従来の重要な機能である、要求を”共有・管理する場”も提供します。必要な関係者を巻き込み”人々の思い”を RRC のエディター機能により表現し、具体化、詳細化を行いながら”要件”へと落とし込むことが可能になります。

Rational Team Concert™ (RTC)は、作業管理・変更管理を軸に、計画管理、ソースコード管理、ビルド管理を統合した All-in-One 製品です。作業とソースコードやビルド結果を紐づけることで、チームの「いつ、誰が、何をしたか／何を作成したか」といったライフサイクルにわたるトレーサビリティを実現します。

Rational® Quality Manager (RQM)は、ソフトウェア品質に関わる「要求、計画、テストケース、結果、障害」までの開発ライフサイクル全体を集中管理します。シームレスな情報共有とトレーサビリティの実現、自動化によるスケジュール短縮、またリアルタイム・ダッシュボード機能により予防的なリスク・マネージメントを実現し、高品質なソフトウェアをより迅速的かつ効率的に構築します。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○									○	○								○	○	○

IBM Rational® Application Developer

対象ＯＳ	対象言語	動作形態	有償／無償
Win2003/XP/Vista/2008/7 SLES9.0/10.0/11.0 RHEL5/6	Java	クライアントツール	有償
提供元	URL		
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/design/rad/		

概要

Rational® Application Developer は Java EE 6.0 に対応し、Web やビジネス（業務）アプリケーションの開発/実行/単体テスト/デバッグ/Java 静的コード解析/動的分析など開発全般をカバーする Java 統合開発環境です。3 世代の WebSphere Application Server （WAS） V6.1,V7.0,V8.0 が同梱されており、古いバージョンの WAS 上のアプリのマイグレーションにもお使いいただけます。Rational Team Concert™との連携することで、複数メンバーによる開発もサポートします。

図　Rational® Application Developer の主な機能

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○	○						○		○							

IBM Rational® ClearQuest®

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/2003/Vista/2008/7 その他多数(Linux)	（管理ツールのため開発言語に依存しない）	Web システム	有償
提供元	URL		
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/scm/cq		

概要

Rational® ClearQuest® の変更追跡機能、プロセス自動化機能、レポート機能、ライフサイクル・トレース機能を使用することによって、ソフトウェア開発ライフサイクルの可視性を高め、より適切にライフサイクルを管理できます。

－　柔軟性の高いカスタマイズ可能な自動ワークフローにより、反復可能、実施可能、予測可能なプロセスが得られます。リアルタイムのレポート機能と測定基準により、プロジェクトの可視性が向上し、より的確な意思決定が可能になります。

－　すぐに使えるアプリケーション・ライフサイクル管理 (ALM) テンプレートにより、アプリケーション・ライフサイクル管理のベスト・プラクティスの強固な基盤が提供されます。アクセス制御、電子署名、反復可能なプロセス、監査証跡により、コンプライアンス管理が簡素化されます。

－　多様化するニーズに応えて、小規模のワークグループからエンタープライズ規模の分散型チームまで、あらゆる規模のチームをサポートします。要件管理ツール、開発ツール、ビルド・ツール、デプロイ・ツールと統合できるため、変更に迅速に対処できます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
																				○

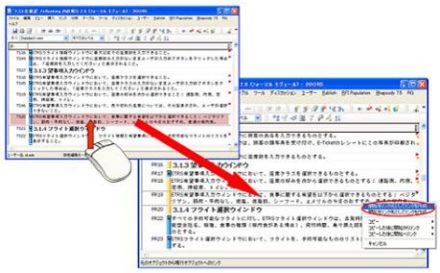
IBM Rational® DOORS®

対象ＯＳ	対象言語	動作形態	有償／無償
(Client) Win2003/2008/XP/Vista/7 (Server) Win2003/2008/RHEL/ SLES/Solaris	（管理ツールのため開発言語に依存しない）	クライアントツール	

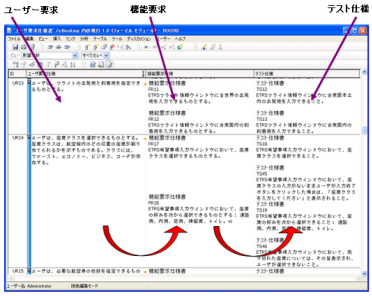
提供元	URL
日本アイ･ビー･エム	http://www-06.ibm.com/software/jp/rational/products/doors/productline/

概要

開発プロジェクトを効率的に進め、かつ、品質を確保するには、要求を正しく把握し、要求に対して合意をし、プロジェクトの活動が要求に適合するように管理 する必要があります。また、頻繁に発生する要求の変更を記録すると共に、その要求変更により影響をうける部分をもれなく確認し、適切なアクションを取って いく必要があります。Rational® DOORS® は、このような要求管理を行っていく上での基礎である、情報の一元管理、変更管理、トレーサビリティ管理の機能を有し、より効率的･効果的な要求管理を 実現します。



図：ドラッグ＆ドロップでのトレーサビリティ記録



図：複数階層にわたる関連項目の可視化

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
○																				

IBM Rational® Functional Tester

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/2003/Vista/2008/7 その他多数(Linux)	Java(J2EE), Web, .NET, SAP, Siebel, Adobe Flex, Dojo, GEF, 端末ベース	クライアントツール	有償

提供元	URL
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/test/rft/

概要

Rational® Functional Tester (RFT) は、テスト対象アプリケーションへの操作を記録し、回帰テストおよび機能テストの自動化を実現します。ユーザー・インターフェース (UI) に対するマウスやキーボード操作を記録し、その内容を再生することが可能です。再生時に検査する項目を追加登録し、テスト・データのバリエーション・テストも容易に設定できることにより、テストの正確性および効率化に貢献します。UI に軽微な修正を行った場合でも、RFT が自動判断するため、スクリプト修正を最小限に抑え、保守性も高めます。

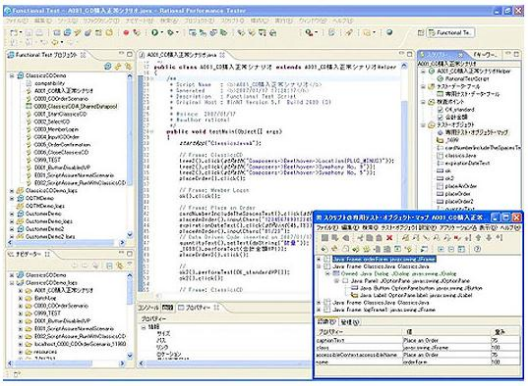


図　スクリプト編集画面

ツール体系とのマッピング（どのツールの種類に該当するツールか）

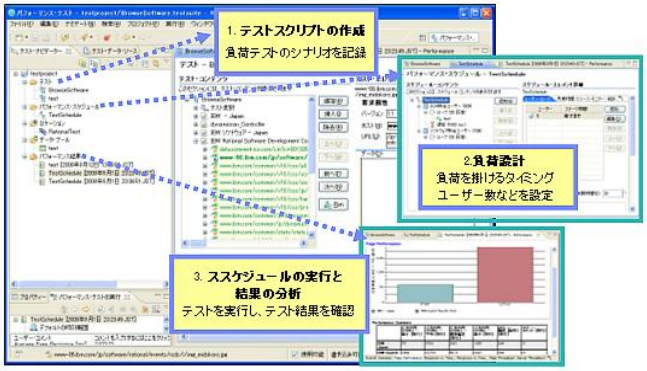
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○										○			○			

IBM Rational® Performance Tester

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/2003/Vista/2008/7 その他多数(Linux)	HTTP/HTTPS、SAP、 Citrix、Siebel、WebService	クライアントツール	有償
提供元	URL		
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/test/rpt/		

概要

Rational® Performance Tester (RPT) は、負荷テスト対象アプリケーションに対して、仮想的に複数ユーザーからのトランザクションを再現することが可能です。テスト・シナリオの組み合わせを自由に選択でき、実際のトランザクションに類似した状況を生成するため、柔軟な負荷テストの設定が容易にできます。負荷テスト実行状況はリアルタイムにツール画面から確認できるため、テストに対する迅速な一次判断に貢献します。



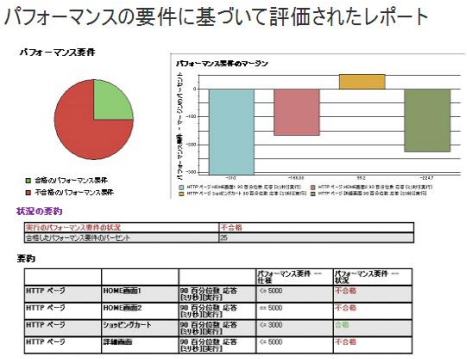


図　スクリプト編集画面

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○											○					

IBM Rational® Policy Tester®

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/2003/2008/7	（管理ツールのため開発言語に依存しない）	クライアントツール	有償

提供元	URL
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/test/policytester/

概要

企業の Web ページは日々更新され、情報量は増加の一途をたどっています。一方でブランドイメージの維持やコンプライアンス遵守のために全社的に管理する必要も増し、大規模化に伴い人手での管理が困難となっています。
IBM Rational® Policy Tester® は企業の Web サイトを自動探索し、品質、プライバシー、アクセシビリティの観点で課題を可視化・一覧化するソリューションです。修正するためのアドバイザリーや結果を共有するためのレポートにより Web サイト運用の効率化とガバナンスの確保を支援します。

図　レポート画面

図 ジョブとレポートのビュー

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計							テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
													○			○				

IBM Rational® PurifyPlus™

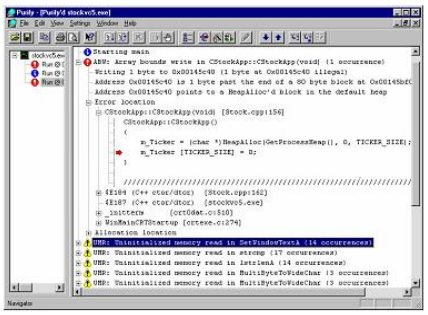
対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/Vista/200/2003/2008/7 Solaris/RHEL/SLES/AIX	C/C++	クライアントアプリケーション	有償
提供元	URL		
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/purifyp/		

概要

Rational® PurifyPlus™は、アプリケーション実行時のランタイム分析をするツールになります。それにより通常のテストでは発見できないメモリリークなどのメモリの問題を発見、対応が可能になります。また、メモリエラーの発見だけでなくテストカバレッジ、パフォーマンスの分析ができます。

PurifyPlus™ を利用することにより、アプリケーションの信頼性とパフォーマンスを向上することができます。

C/C++メモリー解析レポート



カバレッジ統計レポート

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○	○								○							

IBM Rational® Rhapsody®

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/Vista/7	C/C++、Java	Web システム	有償
提供元	URL		
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/rhapsody/productline/		

概要

近年のメカ・エレキ・ソフトの領域をまたがった複雑な開発には、上流工程でのシステム分析が不可欠です。IBM Rational® Rhapsody® により、システムの視点で、要求分析から設計・実装・テストへの追跡可能性の保持、UML/SysML を用いたイテレーティブなモデル駆動システムズ・エンジニアリングを行い、システムとしての品質を確保、開発期間の短縮に寄与することができます。
一方で、製品の差別化要因としてソフトウェアが重視された結果、複雑で高品質のソフトを短期間で開発する要求が高まっています。これらの課題を解くカギは、要求の段階から機能および品質の作りこみを行うこと、開発対象を見通しよく表現すること、継続的なテストおよび統合を行うことにあります。Rhapsody® は、これらの活動をスムーズに行う各種機能を提供し、競争力のある製品の開発をご支援します。

図 ビジュアルモデリング

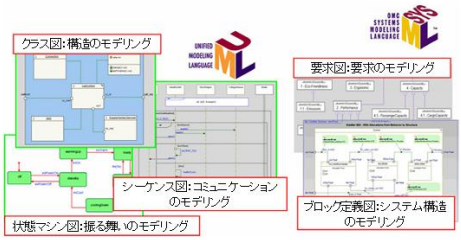
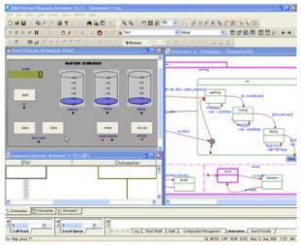


図 モデルのシミュレーション



ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○	○			○			○		○							

IBM Rational® Service Tester for SOA Quality

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/2003/Vista/2008/7 その他多数(Linux)	SOA（Web サービス）	クライアントアプリケーション	有償
提供元	URL		
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/test/tester/soa/		

概要

Rational® Service Tester for SOA Quality は、Web サービス・ベースの SOA アプリケーションの機能/回帰テスト用ツールです。GUI を持たない Web サービスの機能テスト、回帰テストを、スクリプトの記述なしで自動的に実施できます。

- 対象 Web サービス用に生成されたカスタム GUI インターフェースでテストを作成
- シンプルなテスト・オーサリング機能により、スクリプトの記述なしで迅速にテストを作成
- WS-BPEL リソースからの自動テスト生成により、サービス統合テストを簡素化
- WSDL、SOAP、XML/XSD、HTTP(S)、JMS、JSON、MTOM、WebSphere MQ、WS-Security などの一般的な Web サービス標準規格をサポート

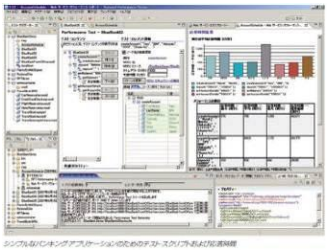


図　スクリプト編集画面

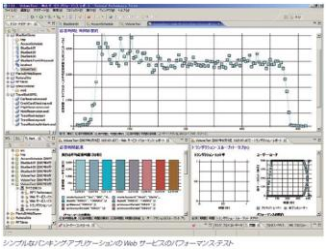


図　テスト結果レポート

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○				○					○							

IBM Rational® Software Architect

対象ＯＳ	対象言語	動作形態	有償／無償
Win2003/XP/Vista/2008/7 SLES9.0/10.0/11.0 RHEL5/6	Java	クライアントツール	有償

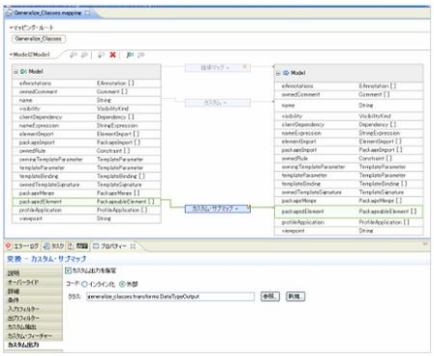
提供元	URL
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/design/rsa/

概要

UML モデルを使った分析・設計から開発までをサポートするモデル駆動型開発のための製品です。 アプリケーションの複雑性を緩和し、リスクを管理しながら、高品質アプリケーションを構築できるだけでなく、新技術の学習曲線の短縮も可能にするソリューションです。

- UML モデリングから Java/JavaEE コーディング・テストまでこれ 1 本！
- UML モデルのメトリックと品質レビュー
- UML モデル変換、レポーティング機能
- UML を用いた開発のベストプラクティスを同梱

モデルからモデルへのトランフォーメーションの定義例



ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○	○						○		○							

IBM Rational® Test Virtualization Server

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/2003/2008/7 AIX5.2/5.3/6.1/7.1 SLES9/10/11 RHEL4/5/6	SOAP, WebSphere MQ, SAP, SWIFT, TIBCO など 70 種類以上のテクノロジー	クライアント サーバー	有償
提供元	http://www-06.ibm.com/software/jp/rational/products/rtw/		
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/rtw/		

概要

システムの複雑な依存関係を低減し、テストを並行的に実施しながら、テスト期間を短縮します。複数のテスト環境をすばやく提供できるため、構築および保守コストを削減できます。アプリケーション間のインターフェース定義からテストスクリプトを自動生成可能であり、テストスクリプトのスケジュール実行も可能です。平行開発およびテストを実現するため、アプリケーションの振る舞いをシミュレーションするスタブをインターフェース定義から自動生成し、サーバーにデプロイするのみで開発者およびテスト担当者が該当のスタブを利用可能です。70 を超えるテクノロジーおよびプロトコルをサポート！

図　Rational Test Virtualization Server 画面

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○				○					○	○	○		○			

IBM Rational® Test Workbench

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/2003/2008/7 AIX5.2/5.3/6.1/7.1 SLES9/10/11 RHEL4/5/6	SOAP, WebSphere MQ, SAP, SWIFT, TIBCO など 70 種類以上のテクノロジー	クライアントツール	有償
提供元	URL		
日本アイ・ビー・エム	http://www-06.ibm.com/software/jp/rational/products/rtw/		

概要

従来の GUI からの機能・回帰テストおよび負荷テストに加え、インターフェース・レベルのテストをサポートします。テスト専門家向けに、機能・回帰テストツール(Rational Functional Tester)および負荷テストツール(Rational Performance Tester)とインターフェース・テストツール(Rational Integration Tester)を組み合わせたソリューションです。アプリケーション間のインターフェース定義からテストスクリプトを自動生成し、平行開発およびテストを実現するため、アプリケーションの振る舞いをシミュレーションするスタブをインターフェース定義から自動生成する機能を提供します。さらに、70 を超えるテクノロジーおよびプロトコルをサポート！

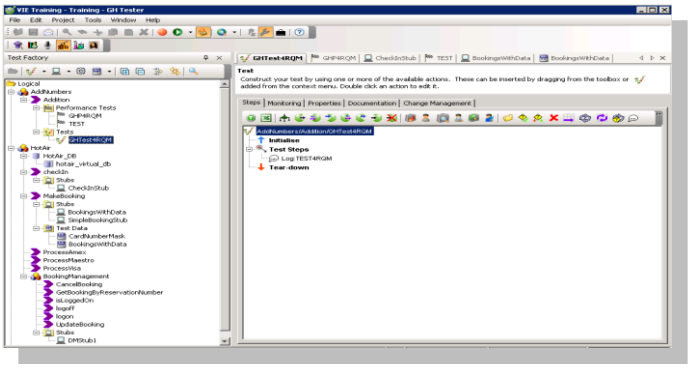


図　Rational Test Workbench 画面

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○				○					○	○	○		○			

IBM Security AppScan®

対象ＯＳ	対象言語	動作形態	有償／無償
WinXP/2003/Vista/2008/7	（管理ツールのため開発言語に依存しない）	クライアント	有償

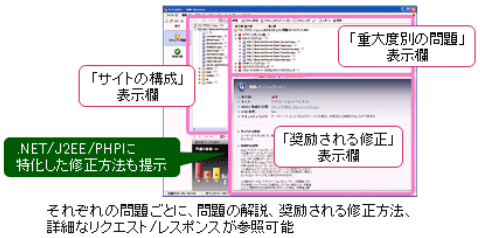
提供元	URL
日本アイ･ビー･エム株式会社	http://www-06.ibm.com/software/jp/rational/products/test/appscan/

概要

システムへの外部からの不正なアクセスによるトラブルは近年ますます増加しております。また、セキュリティーホール、不正アクセス方法も日々発見されており、人手の検査には限界があります。

Security AppScan® は、セキュリティー脆弱性を検査するツールです。Security AppScan® には、ソースコードを静的に検査する Security AppScan® Source Edition と、システムに対して動的に擬似ハッキングをする Security AppScan® Standard Edition/Enterprise Edition があります。Security AppScan® を利用することにより、システムのセキュリティーの問題を開発時に発見し、対策することができます。それにより、運用後のセキュリティートラブルを未然に防ぐことが可能になります。

図 問題を重大度に従い色分け表示



ツール体系とのマッピング（どのツール体系に該当するツールか）

テスト分析	テスト設計			テスト実装				コード解析	テスト実行							テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	原因結果グラフ作成ツール	組合せテスト支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	セキュリティテストツール	パフォーマンステストツール	キャプチャリプレイツール	テストハーネス／ユニットテストツール	プロファイラ	カバレッジ計測ツール	動的解析ツール	構成管理ツール	テスト結果管理／レポートツ	欠陥追跡ツール
								○	○						○			

Jenkins

対象ＯＳ	対象言語	動作形態	有償／無償
Windows、UNIX、Linux、MacOS 等	ー	Web アプリケーション	無償
提供元	URL		
	http://jenkins-ci.org/		

概要

継続的インテグレーションを行うツールです。Ant や Maven などのビルドツールを内包しており、GUI 上から手動で、あるいはスケジューラにより定期的にビルドを実行できます。また、プラグインを追加することで、静的解析ツール、ユニットテストツール、カバレッジ計測ツール、キャプチャ／リプレイツールなどと連携でき、テストに関する様々な作業を自動化できます。さらに、それらの結果をブラウザ上で見ることができたり、プラグインを追加することでメールや Twitter などに結果を自動通知することもでき、品質や進捗をタイムリーに把握するのに役立ちます。スケジューラの設定やプラグインの追加などの様々な作業をブラウザ上で容易にできるのも特徴です。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
																	○		○	

JUnit

対象ＯＳ	対象言語	動作形態	有償／無償
Windows、Linux 等	Java	コマンドライン、各種 IDE のプラグイン（Eclipse、NetBeans 等）、Ant	無償

提供元	URL
JUnit.org	http://www.junit.org/

概要

Java 言語で記述されたソースコードに対するユニットテストフレームワークです。JUnit の API を利用してテスト用のソースコード（テストコード）を作成し、それを実行することによりテスト実行および結果検証が容易にできます。テストコードは、テスト対象となる Java クラスのメソッドを呼び出すドライバに相当し、メソッドへの入力データや、メソッドの戻り値の期待結果を記述します。テストの実行は Eclipse 等の IDE 上で行うことができ、すべてのテストに成功すると緑色のバーが、ひとつでもテストに失敗すると赤色のバーが表示されます。また、JUnit に対応したさまざまなツールと連携することで機能を拡張し、より高度なテストを実施できます。

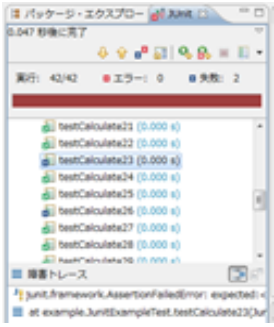


図　JUnit の実行結果画面：どのテストに失敗したかが分かる

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
													○							

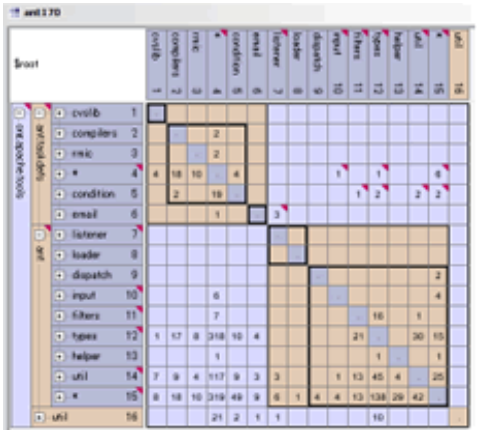
Lattix 7.3.1

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP/2003/Vista/2008/7 (32bit/64bit)、Linux x86/x86-64(x64)	Java、C、C++、.NET、Oracle、SQL、UML/SysML、ActionScript、Ada、Delphi Pascal、Fortran、LDI	クライアントアプリケーション、コマンドライン インターフェイス、Web アプリケーション	有償

提供元	URL
テクマトリックス	http://www.techmatrix.co.jp/quality/lattix/index.html

概要

システムの構成要素間の依存関係を、DSM(Dependency Structure Matrix) と呼ばれる手法により、直感的でわかりやすい表形式(マトリクス)で可視化するアーキテクチャ分析ツールです。リファクタリングのシミュレーション機能や、設計意図に違反する実装を違反としてレポートする機能によって、ソフトウェアの品質を向上させます。また、ソフトウェア変更の影響分析機能は、テストを効率的に計画・実行するのに役立ちます。



システムの構成要素間の依存関係を可視化する DSM(Dependency Structure Matrix)

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
												○								

LDRA tool suite

対象ＯＳ	対象言語	動作形態	有償／無償
9x/NT/2000/XP/Vista/Windows 7、Linux、HP-UX、Solaris、	C、C++、Ada83、Ada95、Java	単独起動、 各 IDE と統合	有償

提供元	URL
LDRA 社（英） 国内代理店：富士設備工業(株)	http://www.fuji-setsu.co.jp/products/LDRA/index.html

概要

- 静的解析、動的カバレッジ解析、単体テスト自動化機能を融合： システムワイドな解析により得られる情報から、コード変更に伴うテストの変更を示唆できるので、規模の拡大や繰り返される派生開発などから敬遠されてきた単体テストやリグレッションテストの実施を支援できる
- あらゆるコンパイラ・実行環境に対応： 組込みシステムの検証に活用
- テストケースの自動生成： ソースコードの解析から自動生成できるテストケースを用いれば、レガシーコードのテストのテンプレートとして、あるいは既存コードの変更後の比較試験に活用できる
- システムワイドにソースコードを可視化： 既存コードを解析してグラフやテーブルを用いて可視化（派生開発時に既存ソースコードを分析して変更によるインパクトを理解するなど）
- 動的解析結果をグラフィカルに可視化： フローグラフ・コールグラフ上に動的テストのカバレッジ結果を反映し、グラフィカルに解析できる。DC、MCDC カバレッジの各実行条件を表示してテストケースプランに活用できる
- 要件トレーサビリティ： 要件管理ツールや Word・Excel・PDF などのドキュメントをインポートし、各要件に領域を割り当て、ソースコードをマッピング。その上でさまざまなテストを実行することで、要件～テストのトレーサビリティを自動取得して管理し、エビデンスを残すことが容易になる。また要件カバレッジや影響解析を行い包括的なレポートを出力できる
- あらゆる国際スタンダードに対応（IEC61508、IEC62304、ISO26262、DO-178B/C、MISRA、CERT など）：開発プロセスのあらゆるフェーズのテスト・検証作業の自動化を支援。要件トレーサビリティを含め、それらの成果物はエビデンスとして認証機関に提出できるフォーマットにまとめることができる

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○				○	○		○	○		○	○		○						○	

Mantis

対象ＯＳ	対象言語	動作形態	有償／無償
LAMP	―	Web アプリケーション	無償
提供元	URL		
MantisBT team	http://www.mantisbt.org/		

概要

Web ベースのインシデント管理ツールです。インシデント情報を処理するフローや権限を自由にカスタマイズができるため、さまざまな開発プロセスに適用させることができます。各インシデントには優先度や重要度を定義でき、対応を優先させるべきインシデントを一覧で確認できます。また、外部システムとの連携ができるよう API が用意されているため、例えば Subversion 等の構成管理ツールから自動的にインシデント情報を更新する使い方もできます。

図　:「Mantis」統計画面

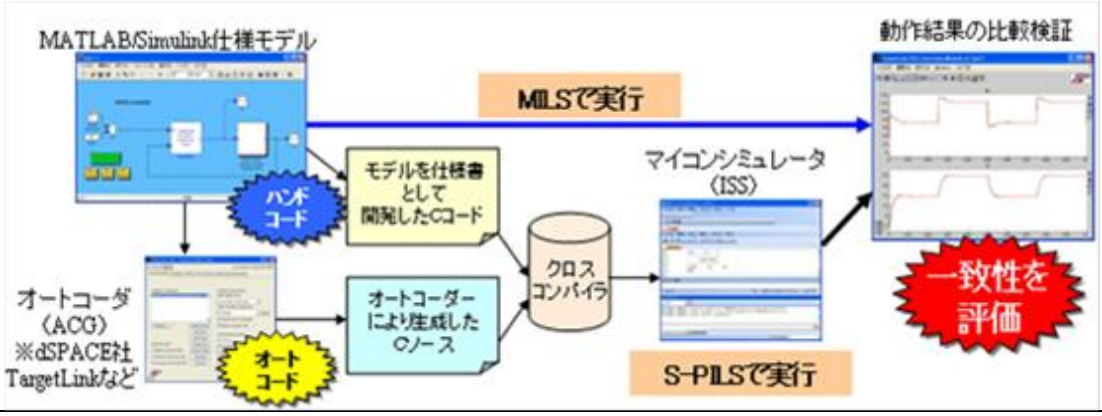
ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
																			○	○

MC-Checker

対象ＯＳ	対象言語	動作形態	有償／無償
WindowsXP Profesional SP3	C, 対応 MATLAB/Simulink: R2006b, R2007b, R14SP2	クライアント（ノードロック）/サーバ（フローティング）	有償
提供元	URL		
ガイオ・テクノロジー株式会社	http://www.gaio.co.jp/product/dev_tools/pdt_mcc.html		

概要

「MC-Checker」は、モデルベース開発における仕様モデル/実装コードの一致性確認ツールです。MATLAB/Simulink による仕様モデルの MIL シミュレーション結果とガイオのマイコンシミュレータによる組込み「実コード」の PIL シミュレーション結果を効率よく比較検証します。ハンドコードや各社 ACG によるコートコードにも対応。自動車機能安全規格 ISO26262 で規定されたモデル/コードの Back-To-Back テストが可能です。



ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○				○		○			○				○		○	

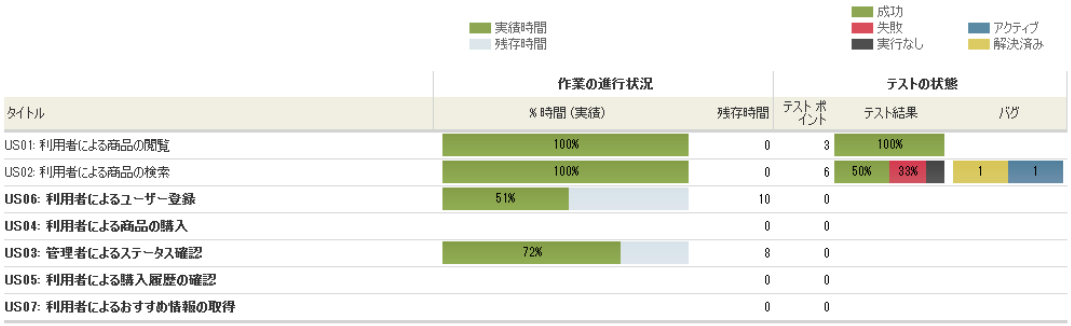
Microsoft® Visual Studio® Team Foundation Server (TFS)

対象ＯＳ	対象言語	動作形態	有償／無償
サーバー: Windows Server 2003 SP2, 2003 R2, 2008 SP2, 2008 R2, Windows Vista, 7	（管理ツールのため開発言語に依存しない）	クライアント: Visual Studio, Eclipse, コマンドライン, Web *対応 OS は、各クライアントの対応 OS（例: Visual Studio: Windows、Eclipse: Linux, Mac OS X や Windows）	有償/無償

提供元	URL
日本マイクロソフト株式会社	http://www.microsoft.com/japan/visualstudio/products/2010-editions/Microsoft® Visual Studio® Team Foundation Server

概要

Microsoft® Visual Studio® Team Foundation Server は、Application Lifecycle Management (ALM) の共通基盤として開発プロジェクトのライフサイクルで作成、利用される開発リソース (要件、タスク、テストケース、バグ、ソースコード、ビルド、テスト環境、テスト結果など)を一元管理する製品です。普段利用している開発環境、テスト環境、マネジメント環境からアクセスすると、開発リソースを目的に応じて最適な形で提供することできます。
要件から開発、自動ビルド、テスト管理と実行、バグと一貫性のある自動化と省力化が実現され、どこからでもトレーサビリティがとれ、包括的かつ透明性の高いレポーティングが行えます。



ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
○						○			○	○							○	○	○	○

Microsoft® Visual Studio® Premium with MSDN®

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP SP3, Vista SP2 Windows 7 Windows Server 2003 SP2, 2003 R2 Windows Server 2008 SP2, 2008 R2	C#, VB, C++/CLI, C++ (Native), F#,	Visual Studio IDE 内テスト機能として動作	有償

提供元	URL
日本マイクロソフト株式会社	http://www.microsoft.com/japan/visualstudio/products/2010-editions/premium

概要

Microsoft® Visual Studio® Premium with MSDN® は Application Lifecycle Management (ALM) を包括的に実現できる Visual Studio です。Visual Studio IDE に開発者テストに必要な機能が搭載されており、実装時でもテストを前倒しで実行できます。コード化された UI テストを製品コードと同じ言語にとより単体テストフレームワーク上で自動実行が可能です。Microsoft® Visual Studio® Team Foundation Server を利用でき、開発者テストを自動ビルドで実施できるため、継続的インテグレーションが行え、品質の劣化を未然に防ぐ仕組みを実現できます。製品コードまたは、テストコードを変更した際に、影響範囲を分析するテスト影響分析などの先進的な機能も含まれます。また、テストに必要な OS やサーバー製品が提供されます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○				○	○	○	○			○	○	○	○	○		○	○	○	○	○

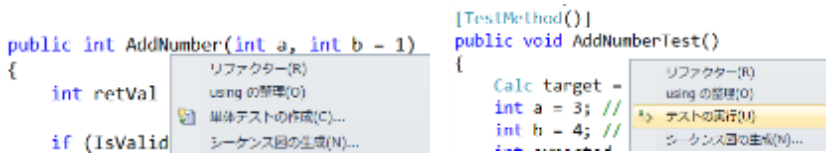
Microsoft® Visual Studio® Professional

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP SP3, Vista SP2 Windows 7 Windows Server 2003 SP2, 2003 R2 Windows Server 2008 SP2, 2008 R2	C#, VB, C++/CLI, C++ (Native), F#,	Visual Studio IDE 内テスト機能として動作	有償

提供元	URL
日本マイクロソフト株式会社	http://www.microsoft.com/japan/visualstudio/products/2010-editions/professional

概要

Microsoft® Visual Studio® Professional は単体テスト機能を含む、統合開発環境を提供します。あらかじめ統合開発環境に、単体テストの実行フレームワークが搭載されており、製品コードから単体テストコードのひな形を生成したり、リファクタリングまでのテスト駆動開発のサイクルに必要な機能を提供しています。



ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
													○							

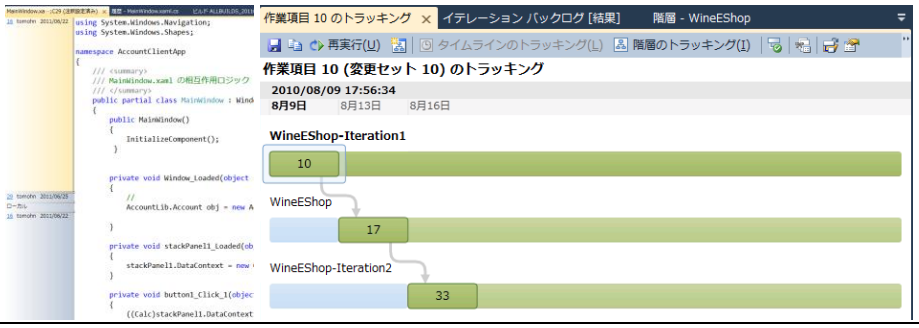
Microsoft® Visual Studio® Professional with MSDN®

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP SP3, Vista SP2 Windows 7 Windows Server 2003 SP2, 2003 R2 Windows Server 2008 SP2, 2008 R2	C#, VB, C++/CLI, C++ (Native), F#,	Visual Studio IDE 内テスト機能として動作	有償

提供元	URL
日本マイクロソフト株式会社	http://www.microsoft.com/japan/visualstudio/products/2010-editions/professional

概要

Microsoft® Visual Studio® Professional with MSDN® は単体テスト機能を含む、統合開発環境と、ALM 基盤である Microsoft® Visual Studio® Team Foundation Server で構成されます。作成した単体テストは、IDE 内から実行できるほか、TFS の自動ビルド機能で実行できるため、継続的インテグレーションを実現することができます。



ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
○										○			○				○	○	○	○

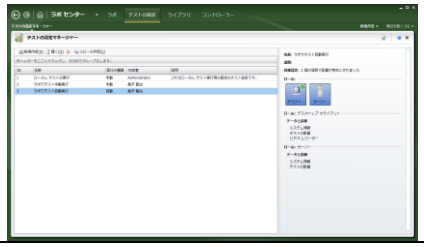
Microsoft® Visual Studio® Test Professional with MSDN®

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP SP3, Vista SP2 Windows 7 Windows Server 2003 SP2, 2003 R2 Windows Server 2008 SP2, 2008 R2	（開発言語に依存しない）	テスト専用ツールとラボ管理	有償

提供元	URL
日本マイクロソフト株式会社	http://www.microsoft.com/japan/visualstudio/products/2010-editions/test-professional

概要

Microsoft® Visual Studio® Test Professional with MSDN® は Application Lifecycle Management (ALM) のテスト領域をカバーするテスト専用ツールと ALM 基盤である Microsoft® Visual Studio® Team Foundation Server で構成されます。テスト計画から実行、結果の追跡やバグ追跡が行える Microsoft Test Manager を中心に、テスト環境の仮想化が可能な Lab Management を利用できます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○						○			○	○				○			○	○	○	○

Microsoft®　Visual Studio® Ultimate with MSDN®

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP SP3, Vista SP2 Windows 7 Windows Server 2003 SP2, 2003 R2 Windows Server 2008 SP2, 2008 R2	C#, VB, C++/CLI, C++ (Native), F#,	Visual Studio IDE 内テスト機能として動作および、テスト計画・実行ツール Test Manager として動作、Eclipse から TFS に接続し、包括的な管理を行うクライアントを提供	有償

提供元	URL
日本マイクロソフト株式会社	http://www.microsoft.com/japan/visualstudio/products/2010-editions/ultimate

概要

Microsoft(r) Visual Studio(r) Ultimate with MSDN® は Application Lifecycle Management (ALM) を包括的に実現できる最上位の Visual Studio です。Visual Studio IDE に設計からテストまでの包括的な機能を提供し、実装時でもテストを前倒しで実行できます。また、テスト管理と実行のための Microsoft Test Manager, ALM 基盤の Microsoft® Visual Studio® Team Foundation Server を利用できるほか、テストに必要な OS やサーバー製品が提供されます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

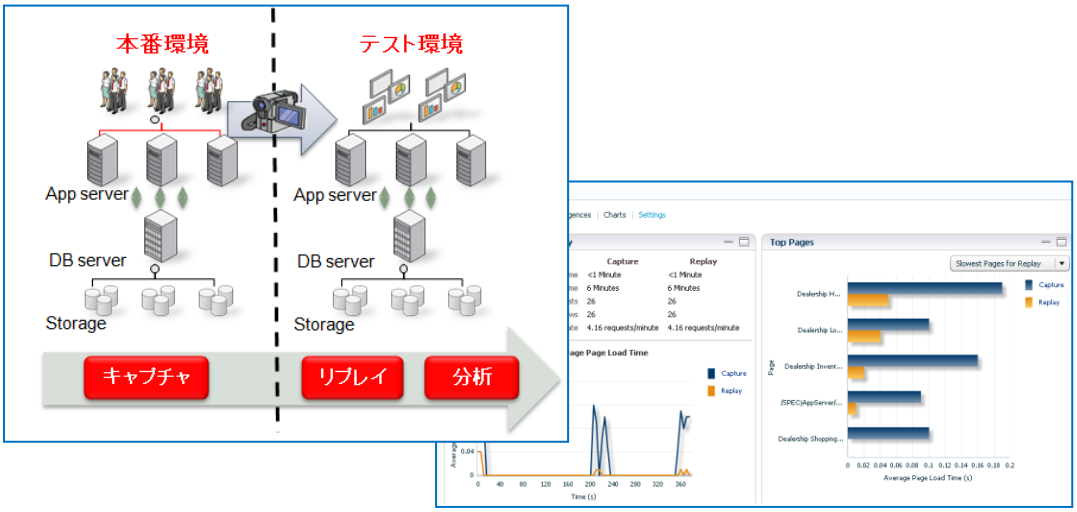
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○				○	○	○	○		○	○	○	○	○	○	○	○	○	○	○	○

Oracle Application Replay Pack

対象ＯＳ	対象言語	動作形態	有償／無償
Linux	開発言語に依存しない	Web システム	有償
提供元	URL		
日本オラクル	http://www.oracle.com/technetwork/jp/oem/app-quality-mgmt/index.html		

概要

本番環境における Web アプリケーションのすべてのワークロード(ユーザのアクセス情報やパフォーマンスデータなど)をキャプチャし、リプレイテストを行います。アプリケーションのインフラの変更、パッチ適用、チューニングなどによる影響調査を容易に行うことができます。
ワークロードは Web アプリケーションの性能をユーザー視点で監視する関連製品の「Oracle Real User Experience Insight」がキャプチャし、本番環境のパフォーマンスへは影響を及ぼしません。



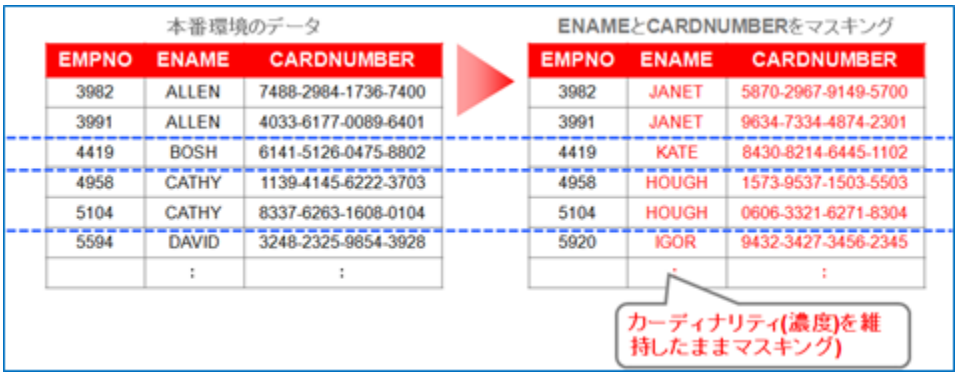
ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
														○	○					

Oracle Data Masking Pack

対象ＯＳ	対象言語	動作形態	有償／無償
Windows、Linux、Solaris、他（Oracle Database の稼動する環境）	Oracle Database	Web システム	有償
提供元	URL		
日本オラクル	http://www.oracle.com/technetwork/jp/oem/app-quality-mgmt/index.html		

概要

本番環境のデータを、機密性を維持しながらテストに活用します。データの形や関係、制約、カーディナリティなどの特性を保ったままデータにマスクをかけて不可視化します。特に個人情報等の機微な情報をテストデータとして利用する際、機密性を維持しながらテストに活用できます。また、マスキングの定義は保存することができ、再利用もできるため、テストデータ作成の効率アップにつながります。



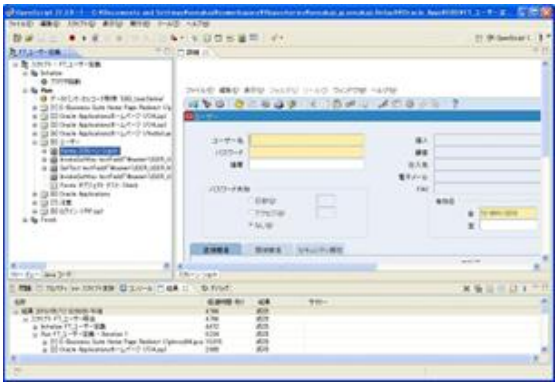
ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
							○													

Oracle Functional Testing

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP、Windows Vista、Windows 2003、Windows 7、Windows 2008	開発言語に依存しない	クライアントアプリケーション	有償
提供元	URL		
日本オラクル	http://www.oracle.com/technetwork/jp/ats-tech/		

概要

Web アプリケーションや Web サービス、Oracle Applications の機能テストや回帰テストを自動化できるテストツールです。コンテンツや応答時間のチェック、エビデンスとしてのスクリーンショット取得など、手動テストと同等の自動テストが実現できます。テストスクリプトは GUI により、直感的かつ簡単に作成することができます。必要に応じて、Java によるカスタマイズや拡張も行え、より柔軟なテストを実現します。
Oracle Functional Testing では Oracle Load Testing で使用する負荷テスト用のテストスクリプトも作成できます。



ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
														○			○			

Oracle JRockit Mission Control

対象ＯＳ	対象言語	動作形態	有償／無償
Windows、Linux、Solaris	Java	JVM のリモートクライアント	有償
提供元	URL		
日本オラクル	http://www.oracle.com/technetwork/jp/middleware/jrockit/mission-control/index.html		

概要

稼動中の Java VM（JRockit JVM）にリモートで接続し、稼動状況を可視化します。リアルタイムの CPU 利用率やメモリ使用状況、メソッド呼び出し状況を確認できます。また、稼動状況をオンメモリで継続記録する機能（JRockit Flight Recorder）を実行 JVM に仕掛けることによって、テスト時および本番稼動時の実際の JVM の挙動（内部スレッド毎の JVM イベント、GC 状況、JBDC 処理状況など）を時系列で可視化できます。

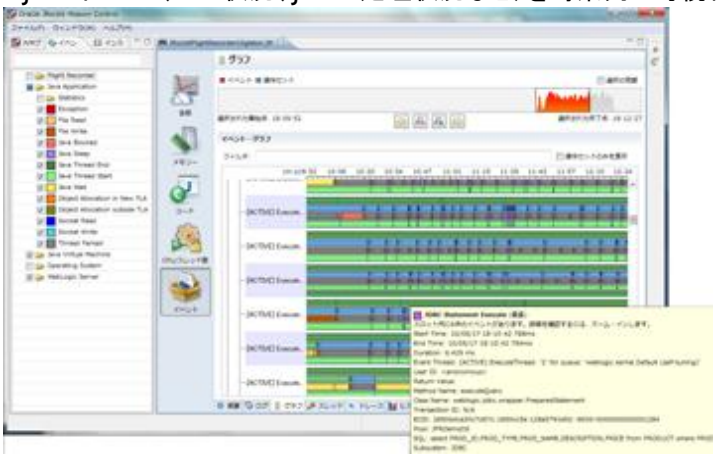


図　キャプチャした JVM 挙動の分析：スレッド毎の時系列処理を可視化し、JDBC 処理状況も分析可能

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○											○					

Oracle Load Testing

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP、Windows Vista、Windows 2003、Windows 7、Windows 2008	開発言語に依存しない	Web システム	有償
提供元	URL		
日本オラクル	http://www.oracle.com/technetwork/jp/ats-tech/		

概要

Web アプリケーションやデータベースのパフォーマンスとスケーラビリティを正確かつ容易にテストできるテストツールです。Web ベースの直感的なユーザー・インタフェースにより、テストシナリオの設定、テストの実行、レポートの作成が簡単に実行できます。また、テスト対象システムの CPU やメモリなどさまざまなリソース情報をエージェントレスで取得することができ、ボトルネック分析を最適化します。
Oracle Real Application Testing や Oracle Real User Experience Insight などから本番環境のトランザクションをインポートでき、よりリアルなテストを実施することもできます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計							テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
															○					

Oracle Real Application Testing

対象ＯＳ	対象言語	動作形態	有償／無償
Windows、Linux、Solaris、他（Oracle Database の稼動する環境）	Oracle Database	Web システム	有償

提供元	URL
日本オラクル	http://www.oracle.com/technetwork/jp/oem/app-quality-mgmt/index.html

概要

システム変更に伴うデータベースの影響を分析し、リスクを削減します。データベースのチューニング、インフラ変更、パッチ適用、アップグレードなど、あらゆる「変更作業」の品質を高めるリアルなテストを行います。アプリケーションがなくても、本番データベースのトランザクションをキャプチャしてテストデータベースに再現。複数の擬似クライアントで高負荷状態のテストを行うことも可能です。数多い SQL のリグレッションテストや、データベース全体のリアルな負荷テストなど、データベースのテスト効率を大幅に向上させます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
														○	○					

Oracle Test Data Management Pack

対象ＯＳ	対象言語	動作形態	有償／無償
Windows、Linux、Solaris、他（Oracle Database の稼動する環境）	Oracle Database	Web システム	有償
提供元	URL		
日本オラクル	http://www.oracle.com/technetwork/jp/oem/app-quality-mgmt/index.html		

概要

複雑かつ大量なデータのサブセットを自動的に生成します。大容量かつ複雑化したシステム環境において、本番環境により近いテストデータを生成するには時間やストレージなどのコストがかかります。Oracle Test Data Management Pack では、データ間連携の維持に必要な最小限の情報を自動的に収集し、データのサブセットを容易に作成します。

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計							テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
							○													

Oracle Test Manager

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP、Windows Vista、Windows 2003、Windows 7、Windows 2008	開発言語に依存しない	Web システム	有償

提供元	URL
日本オラクル	http://www.oracle.com/technetwork/jp/ats-tech/

概要

アプリケーション開発上のテスト工程全体を構築、体系化する、柔軟で操作が容易なテスト工程管理ツールです。「テスト計画」「要件」「テスト/テストステップ」「不具合」を関連付けて管理でき、要件からテスト、不具合からテストなど、相互の影響を容易に把握することができます。「テスト/テストステップ」ではスプレッドシートなどで管理されていた手動テストの他に、Oracle Functional Testing による自動テストスクリプトなどを含めることも可能です。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

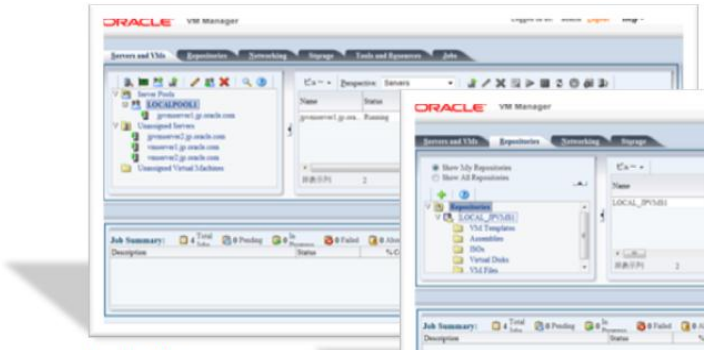
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○										○									○	

Oracle VM

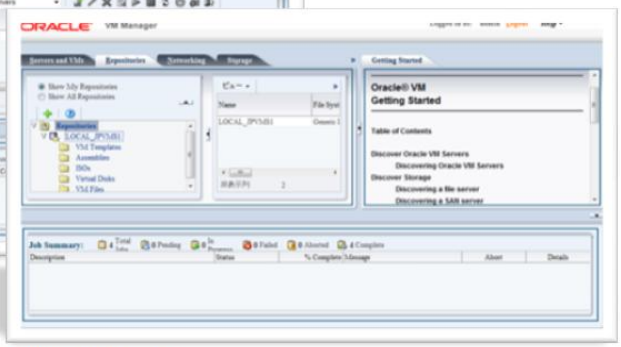
対象ＯＳ	対象言語	動作形態	有償／無償
Linux	開発言語に依存しない	システム	無償
提供元	URL		
日本オラクル	http://www.oracle.com/jp/technologies/virtualization/oraclevm/index.html		

概要

ミッションクリティカルなシステムの稼動にも耐えうる高い性能と信頼性を備えた仮想化ソフトウェア。テストに必要な OS やアプリケーションをインストールおよび構成した仮想マシンイメージを作成できます(Oracle VM Template)。Oracle VM Template をダウンロードして展開するだけでテスト環境を簡単かつ瞬時に構築できます。Oracle Linux や Oracle Database などをインストール/構成した Oracle VM Template も提供しています。



Home View



Quick Start Guide

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計							テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
									○											

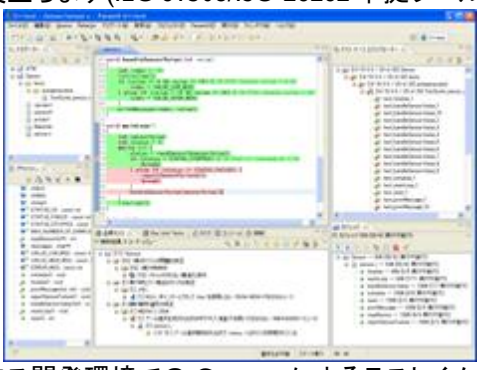
Parasoft C++test 9.2

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2000/ Server 2003/XP/Vista/7、Linux、Solaris 7/8/9/10	C、C++	クライアントアプリケーション、Eclipse プラグイン、Visual Studio プラグイン、コマンドライン インターフェイス	有償

提供元	URL
テクマトリックス	http://www.techmatrix.co.jp/quality/ctest/index.html

概要

C/C++プログラムの総合的なテストツールです。ホスト OS だけでなく、組込みソフトウェアのシミュレータやターゲット機でもテスト実行可能です。MISRA などのコーディング規約検証のほか、MC/DC を含む 6 種類のカバレッジ分析、フロー解析など、安全で高品質なソフトウェアの開発に推奨されるプラクティスを自動化でき、各種安全規格の認証取得にも役立ちます(IEC 61508/ISO 26262 準拠ツール認証取得済み)。



クロス開発環境での C++test によるテストイメージ

ツール体系とのマッピング（どのツールの種類に該当するツールか）

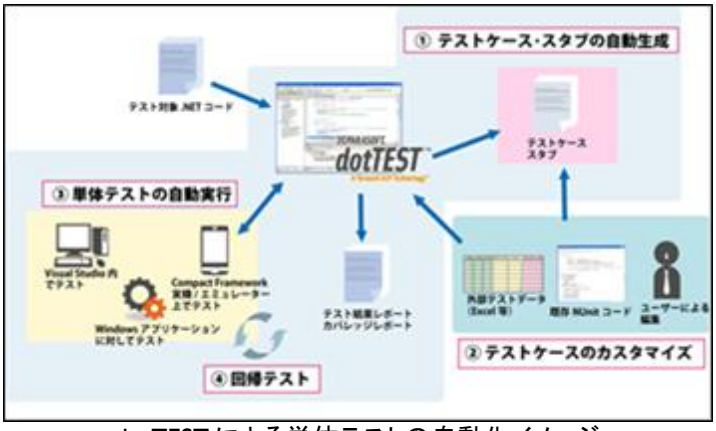
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○	○		○	○		○	○		○			○			○	

Parasoft dotTEST 9.0

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP SP2 以上/2003 Server/Vista/7	C#、VB.NET、Managed C++（一部機能のみ）	Visual Studio プラグイン、コマンドライン インターフェイス	有償
提供元	URL		
テクマトリックス	http://www.techmatrix.co.jp/quality/dottest/index.html		

概要

.NET Framework プログラムの総合的なテストツールです。静的解析機能では、オブジェクト指向開発に役立つルールや C#言語特有のルールなど、約 440 個のコーディングルールによってコードのエラーを指摘します。ユーザー独自のルールをグラフィカルなツールで作成することも可能です。単体テスト機能では、NUnit テストケースの自動生成から、テスト実行、カバレッジ計測、結果レポートまで一連の処理を自動化します。



dotTEST による単体テストの自動化イメージ

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○	○		○	○		○	○		○			○			○	

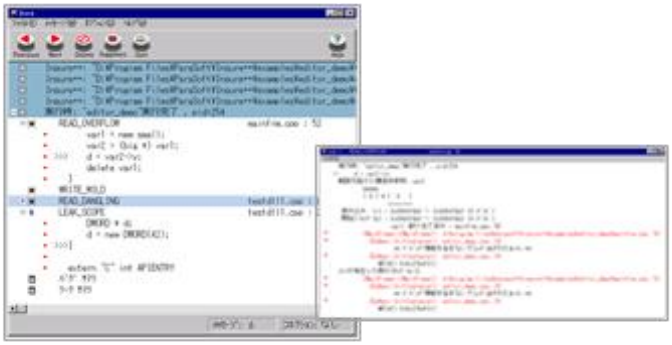
Parasoft Insure++ 7.1

対象ＯＳ	対象言語	動作形態	有償／無償
Windows(32bit): Windows2000 Windows(32/64bit):Windows Server 2003/XP/Vista/7、 Linux(32/64bit)、Solaris 8/9/10(32/64bit)	C、C++	クライアントアプリケーション	有償

提供元	URL
テクマトリックス	http://www.techmatrix.co.jp/quality/insure/index.html

概要

C/C++アプリケーションのランタイムエラーを自動的に検出する開発支援ツールです。メモリ破壊 、メモリリーク、ポインターエラー 、I/O エラーといった C/C++特有の検出困難なエラーをプログラムの実行時に自動的に検出し、ファイル名、行番号、ソースコードなど、プログラムの修正に役立つ情報を的確にレポートします。



Insure++のエラー検出結果

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
				○	○															

Parasoft Jtest 9.1

対象ＯＳ	対象言語	動作形態	有償／無償
Windows 2000/ XP/2003/ Vista/7(32/64bit) Red Hat Enterprise Linux 3/4/5(32/64bit) Solaris 10	Java	クライアントアプリケーション、Eclipse プラグイン、コマンドライン インターフェイス	有償

提供元	URL
テクマトリックス	http://www.techmatrix.co.jp/quality/jtest/index.html

概要

Java プログラムの総合的なテストツールです。静的解析機能では、業界のベストプラクティスに基づく約 1,100 個のコーディングルール、70 種類のメトリクスルールによる解析のほか、処理フローに依存するバグを静的に検出するフロー解析機能を備えています。単体テスト機能では、Struts、Spring などのフレームワークを使用したアプリケーションのテストケース生成にも対応し、サーバサイドコードのテストの自動化に利用できます。

処理フローに依存するバグを静的に検出するフロー解析（バグ探偵）

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○	○		○	○		○	○		○			○			○	

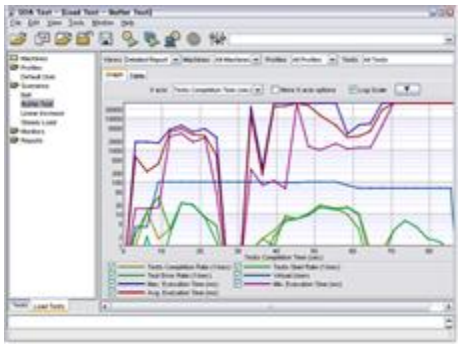
Parasoft SOAtest 9.0

対象ＯＳ	対象言語	動作形態	有償／無償
Windows 2000/2003/XP/ Vista/2008/7(32/64bit)、 Linux (32/64bit)、 Solaris(32bit)	XML、WSDL、SOAP、 REST、JSON、JavaScript、 VBScript/ASP、JSP、 HTML、CSS など	クライアントアプリケーション、 Eclipse プラグイン、コマンドライン インターフェイス	有償

提供元	URL
テクマトリックス	http://www.techmatrix.co.jp/quality/soatest/index.html

概要

SOA/Web サービスおよび Web アプリケーションの機能・セキュリティ・パフォーマンスを総合的に検証するテストツールです。アプリケーションの操作をキャプチャしてテストシナリオを作成し、機能テストを自動化できるほか、Excel などの外部データを利用したデータ駆動型テスト、ペネトレーションテストによるセキュリティの検証にも対応。また、仮想クライアントによる負荷テスト機能も備えています。



負荷テスト実行画面

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
				○			○	○		○	○			○	○	○			○	

Parasoft Virtualize 9.3

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2000/2003/XP/Vista/2008/7(32/64bit)、Linux(32/64bit)	HTTP/HTTPS、JMS、MQ、SOAP、REST、JSON など多くのプロトコルや技術をサポート	クライアントアプリケーション、Eclipse プラグイン、コマンドライン インターフェイス、Web アプリケーション（環境マネージャー）	有償

提供元	URL
テクマトリックス	http://www.techmatrix.co.jp/quality/virtualize/index.html

概要

システムを構成するコンポーネントを仮想化し、振る舞いをシミュレートするサービス仮想化ツールです。テストに必要な実環境のシステム、コンポーネント、サービスの振る舞いを仮想化環境で再現することで、各チームがあたかも実環境を利用しているかのように、実施したいタイミングでアプリケーションをテストできるようにします。これによって、並行開発や機能テスト時のアクセスやテスト環境の利用に関する問題を解決し、開発効率の向上とコスト削減を実現します。

グラフィカルな UI 上での数クリックの操作でテスト環境をセットアップ

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
							○	○	○	○										

PICT

対象ＯＳ	対象言語	動作形態	有償／無償
Windows	（開発言語に依存しない）	コマンドライン	無償
提供元	URL		
Microsoft	http://download.microsoft.com/download/f/5/5/f55484df-8494-48fa-8dbd-8c6f76cc014b/pict33.msi		

概要

All-pair 法を適用した組合せを自動生成するツールです。パラメータ名（因子）とその取りうる値（水準）を列挙したテキストファイルを入力とし、それらに対して All-pair 法に基づいた組合せ、すなわちふたつのパラメータ間の全組合せを網羅したテストケースを自動生成します。パラメータ間で同時に成立しえない組合せ（禁則）を除外したり、3 つ以上のパラメータによる組合せを網羅したテストケースを生成したりといった機能もあります。

図　PICT の入力（左）と出力された組合せ（右）

ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
		○																		

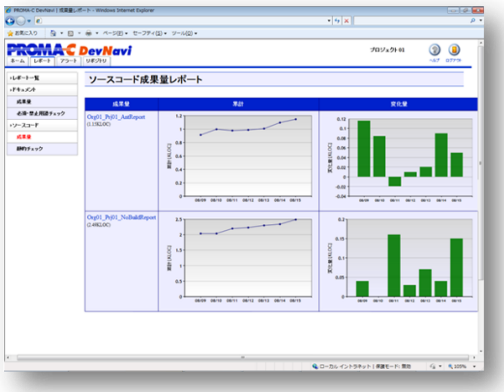
PROMA-C DevNavi Ver2.0

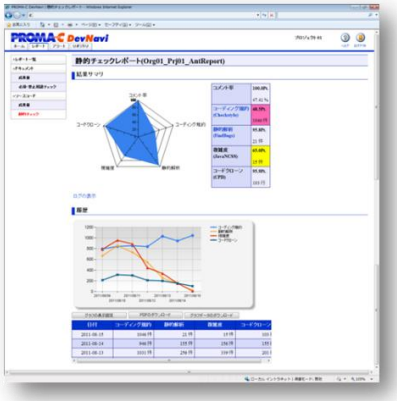
対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP/Vista/7	Java .NET、C/C++	Web アプリケーション/クライアントソフトウェア	有償

提供元	URL
Acroquest Technology	http://endosnipe.acroquest.co.jp/index.html

概要

PROMA-C DevNavi(デブナビ)とは、構成管理された成果物に基づき、成果量や静的解析結果及び品質を自動的に「見える化」する、今までにない構成管理 SaaS です。
Subversion と Trac をベースとした構成管理・課題管理機能のほか、プロジェクトの進捗状況や品質状況を自動的に出力するレポート機能で開発状況を「リアルタイムに見える化」し、プロジェクトを成功に導きます。





ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
											○							○		○

Quality Commander

対象ＯＳ	対象言語	動作形態	有償／無償
組込み機器、MS Windows、Android	ー	Windows ネイティブアプリ	有償

提供元	URL
日本ノーベル株式会社	http://www.jnovel.co.jp/qc/index.html

概要

組込みシステムのテスト自動化システムです。スマートフォンからカーナビやタッチパネル製品など、組込みシステムの操作を様々なロボットを駆使して自動操作し、強力な画像処理技術で画面表示が正しいか OK・NG 判定します。ロボットを使わない構成としては、Windows アプリケーションや Android デバイスを USB 接続で制御できます。対象機器のみならず、周辺の機材も同時に丸ごと自動化でき、拡張性が非常に高いです。

図　:XY ロボットとソフトウェア

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
														○	○		○			

Selenium

対象ＯＳ	対象言語	動作形態	有償／無償
ー	ー	ブラウザ拡張 C/S システム	無償
提供元	URL		
seleniumhq	http://seleniumhq.org/		

概要

Web アプリケーションのテスト実装、実行を省力化するためのツールです。キャプチャリプレイツールにおけるをキャプチャをブラウザ(Firefox)上で行うためのブラウザ拡張「Selenium IDE」を始めとして、プロキシサーバを介して任意のプログラミング言語にとテスティングフレームワークよるテストを実行できる「Selenium Remote Control(RC)」を中心にさまざまなコンポーネント、製品が存在します

図　:Selenium Remote Control(RC)の動作モデル

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
														○			○			

SilkCentral® Test Manager

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008 WindowsXP/Vista/7	（管理ツールのため開発言語に依存しない）	Web システム	有償

提供元	URL
マイクロフォーカス	http://www.microfocus.co.jp/products/silk/silkcentraltestmanager/

概要

テスト要件、テスト項目、テストシナリオ、テストデータ、テスト実行環境、テストで発生した問題などのテスト作業に関連するさまざまな情報を一元管理します。
マイクロフォーカスのテスト自動化ツール SilkTest、負荷テストツール SilkPerformer を始め Junit、Nunit、Fitnesse、VMware Lab Manager などのソフトウェアのテストフレームワークを利用した自動テスト実行ができます。自動テスト実行の際にはスケジュール、実行サーバ、ビデオ録画の有無、実行結果のメール通知などの設定ができます。
標準で提供されている手動テスト機能を利用してテスト前にテストの手順、入力データの情報と期待される結果の定義を行い、テスト手順の進行とともに結果を順次入力していくことができます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
					○					○									○	○

SilkPerformer®

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008 WindowsXP/Vista/7	BDL(Pascal ライクな言語)	専用クライアント	有償
提供元	URL		
マイクロフォーカス	http://www.microfocus.co.jp/products/silk/silkperformer/		

概要

Web システム、クライアント/サーバ、端末エミュレータ等の負荷テストが行えます。
スマートフォンへの負荷テストにいち早く対応し、Android、iPhone、iPad 等の実機を利用したスクリプト作成およびネットワーク帯域を指定した負荷テストが可能です。
また、クラウド環境に負荷テスト用エージェントを配置して日本、シンガポール、ブラジル、アメリカ3か所、ヨーロッパの 7ヶ所にある AmazonEC2 サイトから負荷テストを実行できます。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果[管理／レポート]ツール	インシデント管理ツール
															○					

SilkTest®

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008 WindowsXP/Vista/7	非手続き型言語、VB.NET、 C#.NET、Java	専用クライアント、 VisualStudio、Eclipse	有償
提供元	URL		
マイクロフォーカス	http://www.microfocus.co.jp/products/silk/silktest/		

概要

スクリプト作成用の専用統合環境から視覚的に優れたビジュアルテストと呼ばれる非手続き型言語を用いてGUI テストの記録/再生が行えます。エキスパートユーザ向けには専用統合環境から VB.NET を用いたるスクリプト環境も利用できます。
VisualStudio や Eclipse を利用して開発しているエンジニア向けには C#や VB、Java を利用したスクリプト記録/再生機能を提供しています。
IE で記録したスクリプトを IE の複数バージョン、Firefox、Chrome で変更無しに実行できるクロスブラウザ機能は、Web2.0 におけるトレンドリーダの機能として有名です。
ブラウザの最新版への対応、新 OS、新しい技術への対応は早いと言われています。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
														○			○			

StarTeam

対象ＯＳ	対象言語	動作形態	有償／無償
サーバ：Windows Server 2003、2008 クライアント：Windows XP、Vista、7	（管理ツールのため開発言語に依存しない）	Client/Server クライアントは、Web、Windows、Eclipse/VisualStudio	有償

提供元	URL
マイクロフォーカス	http://www.microfocus.co.jp/products/starteam/starteam/

概要

StarTeam には、さまざまな規模、地理的状況、作業スタイルを持つ、あらゆる開発チームのニーズを満足させる、ソフトウェアの構成および変更を管理するための機能が完備されています。ソフトウェア開発プロセス全体をまとめて管理する堅牢なプラットフォームである StarTeam は、プロジェクト関連のすべてのデジタル資産、およびこれらの資産に変更を加える作業内容を一元管理することにより、チーム内のコミュニケーションとコラボレーションを促進します。柔軟で、しかもセキュリティ保護されたクライアントアクセスが可能なため、チームメンバーはいつでもどこでも、デスクトップ、Web ブラウザ、IDE、コマンドラインクライアントを通じて作業を行うことができます。Micro Focus ALM 製品ファミリのひとつである StarTeam は、プロジェクト・タスク管理のほかに、統合された変更管理、欠陥追跡、ファイルのバージョン化、要件管理、スレッド形式のディスカッションなどを含む、他に類のない包括的ソリューションを提供します。

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
																		○		○

StateMatrix

対象ＯＳ	対象言語	動作形態	有償／無償
Windows	（開発言語に依存しない）	Excel マクロ	無償
提供元	URL		
	http://sourceforge.jp/projects/testools/		

概要

状態遷移表からｎスイッチの状態遷移パスを自動抽出するツールです。遷移前の状態、実行イベント、遷移後の状態を表した状態遷移表を記述し、スイッチ数を指定すると、遷移前の状態から、どのようなイベントの組合せ（順序）によって遷移後の状態に移るかを自動的に生成します。イベントの組合せが複数存在する場合は、それらが「+」で結合された形で羅列されます。

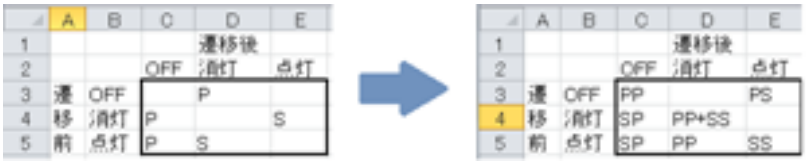


図　状態遷移表（左）から１スイッチの状態遷移パスを自動生成（右）

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
	○																			

T-VEC

対象ＯＳ	対象言語	動作形態	有償／無償
Windows 各種	プログラミング言語に依存しない	単独起動、および Simulink と統合	有償

提供元	URL
T-VEC 社（米） 国内代理店：富士設備工業(株)	http://www.fuji-setsu.co.jp/products/T-VEC/index.html

概要

形式手法（フォーマルメソッド）を活用する MBT（モデルベースドテスト）ツール。要求仕様や設計モデルを、独自の形式手法を活用して解析し欠陥を抽出します（定理証明）。この解析結果として得られる入力値と期待値をテストベクタとして、ソースやオブジェクトコードの一致性の検証に活用できることが特徴。
＊検査、テストベクタ生成時のデータ型はフローティング、ダブルにも対応（インテジャー、ブーリアンなどのみでは無い）

- リニア、ノンリニア式 に対応
- サブシステム毎に上流のコンストレインツを加味した入力で検証（状態爆発しない）
- 生成されるテストベクタを活用して一致性検証ができる・要件トレーサビリティを支援できる
- 要求モデル、設計モデルの両方に対応する

ツール体系とのマッピング（どのツールの種類に該当するツールか）

テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○	○	○				○	○			○									○	

TestLink

対象ＯＳ	対象言語	動作形態	有償／無償
LAMP	ー	Web アプリケーション	無償
提供元	URL		
TestLink Dev Team	http://testlink.sourceforge.net/docs/testLink.php		

概要

Web ベースのテストマネジメントツールです。テスト項目の管理はもとより、テスト実施状況やテスト実施結果を管理できます。テスト計画を立てて各テスト項目にメンバーをアサインするなど、テストプロセスのマネジメントをサポートするツールです。要件からテスト項目を紐付けができ、各テスト項目の版管理もできるため、要件に変更があった場合でも変更が必要なテスト項目に漏れが無いようトレーサビリティを確保できます。

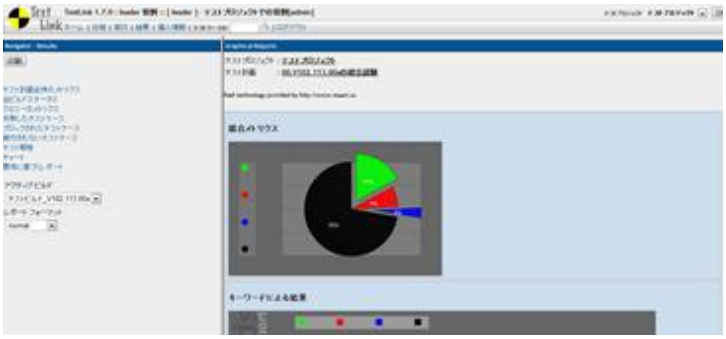


図　:「TestLink」メトリクス表示画面

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○										○								○	○	

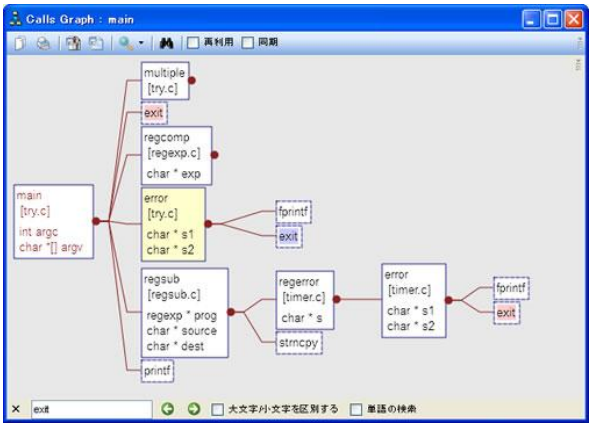
Understand 2.6

対象ＯＳ	対象言語	動作形態	有償／無償
Windows 2000/XP/2003/Vista/2008/7 (32bit/64bit)、Linux x86/x86-64(x64)	C、C++、C#、Java、Ada、Fortran、Jovial、Pascal、PL/M、VHDL、Web (HTML、CSS、JavaScript、PHP)	クライアントアプリケーション、コマンドライン インターフェイス	有償

提供元	URL
テクマトリックス	http://www.techmatrix.co.jp/quality/understand/index.html

概要

大規模で複雑なプログラムを素早く解析するソースコード解析ツールです。プログラムの制御フローや構造、クラス継承、関数や変数の関係などを、グラフィカルなビューやリファレンス情報としてわかりやすく表示し、ソースコードの理解を容易にします。また、サイクロマチック複雑度など 70 種類のコードメトリクスを分析することで、コードの品質を確認したり、レビューやリファクタリングが必要な複雑すぎるコードを検出できます。



関数呼び出しの階層関係を可視化するビュー

ツール体系とのマッピング（どのツールの種類に該当するツールか）

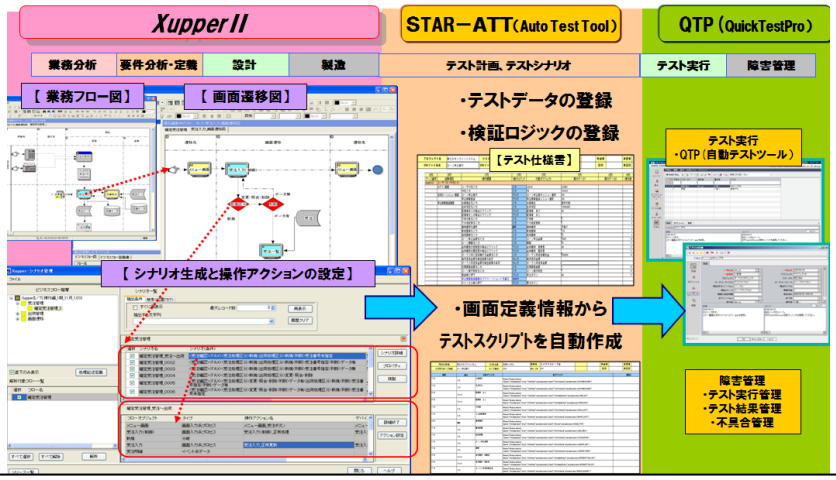
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ／スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
												○								

Xupper Ⅱ /TI Ver9.0

対象ＯＳ	対象言語	動作形態	有償／無償
Windows2003/2008(Server) WindowsXP/7(Client)	設計ツールのため開発言語に依存しない	C/S システム。TSE で Web 可	有償
提供元	URL		
ケン・システムコンサルティング	http://www.kensc.co.jp		

概要

Xupper Ⅱ リポジトリに登録されているビジネスフロー図と画面遷移図を解析し、テストシナリオを自動生成します。テストシナリオを「STAR－ATT※1」へ連携し、テストデータと検証ロジックを登録し、テストスクリプトを自動生成。さらに 「HP QuickTest Professional software（QTP）※2」を駆動し、テストを自動実行します。



※1 STAR-ATT は 株式会社フロンテス の製品です。　※2 HP QuickTest Professional は 日本ヒューレット・パッカード株式会社の製品です。

ツール体系とのマッピング（どのツールの種類に該当するツールか）																			
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
○						○				○								○	○	

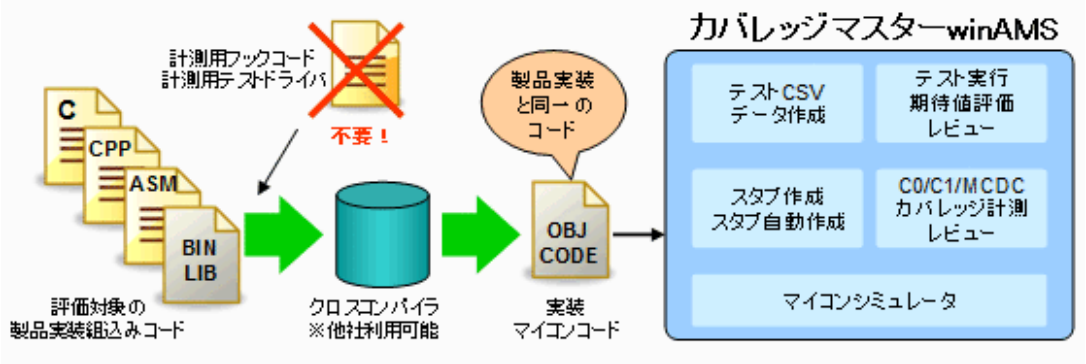
カバレッジマスターwinAMS

対象ＯＳ	対象言語	動作形態	有償／無償
Windows XP SP2 以降 Windows Vista Windows 7	C/C++	クライアント（ノードロック）/サーバ（フローティング）	有償
提供元	URL		
ガイオ・テクノロジー株式会社	http://www.gaio.co.jp/product/dev_tools/pdt07_winams.html		

概要

C/C++ 組込みソフト検証用 モジュール単体テスト自動化ツールです。
マイコンシミュレータ上でテスト実行するため、フックコードを使わず実装マイコンコードをそのまま動作させて単体テストが実行できます。（ステートメントカバレッジ、ブランチカバレッジ、MC/DC カバレッジの計測が可能）
静的解析ツール「CasePlayer2」と連携してブランチカバレッジ、MC/DC カバレッジの最適化テストケースの自動作成をサポートします。

日本語／英語バージョンをリリースしています。またテスト結果レポート出力は多言語対応が可能です。



ツール体系とのマッピング（どのツールの種類に該当するツールか）																				
テスト分析	テスト設計						テスト実装				コード解析		テスト実行					テストウェア管理	テスト結果管理	インシデント管理
要件管理ツール	状態遷移テストツール	組合せテスト支援ツール	原因結果グラフ作成ツール	動的解析ツール	カバレッジ計測ツール	その他のテスト設計支援ツール	テストデータジェネレータ	シミュレータ/スタブ	ラボイメージ	テストケース管理ツール	静的解析ツール	構造解析ツール	ユニットテストツール	キャプチャ／リプレイツール	性能テストツール	セキュリティテストツール	テスト自動実行支援ツール	構成管理ツール	テスト結果［管理／レポート］ツール	インシデント管理ツール
		○			○		○	○		○			○				○		○	

参考文献

- 日経システムズの調査結果　http://itpro.nikkeibp.co.jp/article/COLUMN/20110512/360288/
- 「ハイ・コンセプト「新しいこと」を考え出す人の時代」ダニエル・ピンク著 大前研一訳　三笠書房 2006
- ソフトウェアテスト標準用語集（日本語版） Version 2.1.J01 ISTQB 編　JSTQB 技術委員会訳 JSTQB 2011
- テスト技術者資格制度 Foundation Level シラバス 日本語版　Version 2011.J02 ISTQB 編　JSTQB 技術委員会訳 JSTQB 2011
- ソフトウェアテスト技法ドリル―テスト設計の考え方と実際 秋山浩一 日科技連出版社 2010

執筆者一覧

ASTER テストツール WG「テストツールまるわかりガイド（入門編）」 作成メンバー

浅野義雄	富士設備工業
東大輔	日本ノーベル / ASTER
大西建児	ガイオ・テクノロジー / ASTER
金元隆志	日本アイ・ビー・エム
酒井利治	
高橋俊夫	ケン・システムコンサルティング
長沢智治	日本マイクロソフト
中島良樹	日本オラクル
西田啓一	
速川徹	Acroquest Technorogy
堀岡勝	コベリティ日本支社
槇信之	Acroquest Technorogy
町田欣史	NTT データ
松尾政仁	日本ヒューレット・パッカード
松木晋祐	ACCESS /ASTER
安竹由紀夫	コベリティ日本支社
山城裕一	マイクロフォーカス
湯本剛	日本ヒューレット・パッカード / ASTER

（50 音順）

テストツールまるわかりガイド（入門編）

2012 年07月02日Version 1.0.0 版発行
著者　ASTERテストツールWG
発行　NPO 法人ソフトウェアテスト技術振興協会
http://aster.or.jp/