結合テスト工程の成果物作成要領を記す。

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | 成果物名 | 作成要領 | ｻﾝﾌﾟﾙ |
| 1 | 結合テスト仕様書 | テスト仕様書には、テスト情報とテスト管理情報を記載する。  ・テスト情報 　テストの内容とテスト実行結果を記す。 　・テスト内容 　　・テストケース 　　　・テストケースを洗い出す 　　　　・画面の操作や実行手順を具体的な業務の順序に沿って洗い出す。 　　　   　・ユーザ要望の操作パターンだけでなく、システム的に実行可能な 　　　　   　全てのパターンを洗い出す。 　　　   　・ただし、時間（コスト）との兼ね合いで、業務上発生することがない 　　　　   　例外パターンについてはモンキーテストで検証するが、テストケース 　　　　   　の細分化に多くの時間は費やさない等の工夫も必要。 　　   　・論理的にパターンを洗い出す（デシジョンテーブル等で整理する） 　　　　   ・データパターンは条件と動作の組み合わせで整理することができる。 　　　　   　例）テーマパークのフリーパスを発券する。 　　　　　　・条件①子供／大人区分　→金額のパターンに影響する 　　　　　　　　　　・未就学時 　　　　　　　　　　・小学生～大学生 　　　　　　　　　　・社会人 　　　　　　　　　　・60歳以上　　　　　　　　 　　　　　　・条件②パーク滞在時間　→金額のパターンに影響する 　　　　　　　　　　・1日　　　 　　　　　　　　　　・17時以降　　　　　 　　　　　　・動作　・発券する　 　　　　　　　　　　・発券しない　　　 　　  　　　　　＜パターンの洗い出し表例＞ [F_テストパターンの洗い出し](http://open-process.jp/home/wp-content/uploads/2015/09/c647146459e1327c23638eb1255764b4.png)　　・テスト手順：具体的な手順を記載する 　　・予想結果：システムの出力結果の記載は具体的な数字や文字列にする 　　 　・テスト実行結果 　　・テスト結果（OK/NGとNGの場合の詳細内容） 　　・確認日 　　・確認者 　　・不具合対応状況（不具合一覧の管理番号とステータス） 　　　不具合の詳細情報は不具合一覧に記載するが、テストケースを 　　　完了して良いかを判断するための情報として不具合の対応状況を 　　　記載する。  ・テスト管理情報 　テスト管理情報はテスト品質の妥当性確認と、テスト進捗状況の把握の 　ために使用する。 　　 　・テスト品質の妥当性確認 　　テストの品質が妥当か否かを判断するために、1機能あたりの 　　テストケース設定数を検証する。 　　ただし、1機能あたりのテストケースを設定すると言っても、 　　それぞれの機能は同じソースコード行数ではないので、1機能あたり 　　のテストケース数を単純に比較することは誤った示唆を導き出して 　　しまう恐れがある。複雑な機能と単純な機能では、1機能あたりの 　　テストケース数に違いがあるのが当然で、複雑な機能の方が 　　テストケース数は多くなる傾向がある。　　 　　そのため、それぞれの機能の規模を考慮する必要がある。 　　 　　・1機能の規模の把握 　　　1機能あたりのテストケース設定数（＝ケース密度）を検証するためには 　　　まずそれぞれの機能の大きさを定量的に把握する必要がある。 　　　1機能の規模を把握する方法として、以下のような考え方がある。 　　　・SLOC（Source Lines Of Code:ソースコード行数）　 　　　　SLOCが多ければ、検証ポイントは増えるという仮説に基づいている。　 　　　・機能の要素分解（データ項目数ｘ機能の複雑度） 　　　　＜要素分解　機能の複雑度の例＞ 　　　　　複雑度：１　元のデータをそのまま登録する 　　　　　複雑度：２　データを単純変換して登録する 　　　　　　　　　　　　（A→１　B→２　のような単純な変換） 　　　　　複雑度：３　数式によってデータ計算して登録する 　　　　　 　　　　　IPAのソフトウェア開発データ白書では、SLOCのほかに 　　　　　FP（ファンクションポイント）を利用している。 　　　　　 　　　・設計に費やした実績工数 　　　　多くの時間を費やして設計した機能は、ソースコード量が 　　　　多くなると見なす考え方。 　　　 　　　SLOCが最も一般的な考え方だが、パッケージ製品の場合は 　　　システム開発時にSLOCを把握できないことがある。 　　　そのような場合に、機能の要素分解や設計の実績工数を 　　　用いることを検討する。 　　　 　　・ケース密度の妥当性（テスト結果報告書に記載する） 　　　テスト仕様書にテストケース密度の情報は記載しなくてもよいが、 　　　テスト完了までに密度の分析は実施する。 　　　1機能あたりのケース数の妥当性は過去に同様または類似する機能の 　　　テストを実施していればその時の数値をひとつの参考情報として用 　　　いることを検討する。 　　　過去の数値を利用できない場合は、当該プロジェクトの結合テスト 　　　の中で、密度が高い機能と低い機能を比較し、 　　　密度の差に客観的に説明しうる理由があるのかを確認する。 　　　 　・テスト進捗状況の把握 　　　テスト期間中にテストが終わるのか否かを把握するために、 　　　テストケースの完了状況と不具合の解消状況を把握する。 　　　＜進捗状況を把握するために使用する情報＞ 　　　　・テストケースの合計件数 　　　　・テスト結果の確認が完了した件数 　　　　・テスト結果がNGの件数 　　　　・未解消の不具合の件数 | [Excel](http://open-process.jp/home/wp-content/uploads/2015/08/ff59148028812ae92d2089df15d909f2.xls)ﾃｽﾄ 仕様書 |
| 2 | テストデータ | テストデータの作成は効率性と実用性を備えなければいけない。 　・効率性：データ作成にかかる時間をできるだけ少なくする 　・実用性：テストデータを見ただけで何のデータかわかること 　　これを満たすために、例えば下記のようなデータにする。 　　・会社名なら「取引先会社０００１」「グループ会社０００１」 　　・顧客名なら「顧客０００１」「顧客０００２」 　 　上記のデータは、Excelのオートフィル機能を使って簡単に連番の 　バリエーションを増やすことができる。Excelやテキストエディタの 　マクロを利用して極力手間をかけずにデータを作成することを推奨する。 | – |
| 3 | テストツール | Selenium、Appium、QTP等の自動化ツールを利用するか、カスタムツール 　を作成する。ツール利用に関する検討の詳細は基本設計工程の全体テスト 　計画を参照。 　　＜自動化する作業の範囲＞ 　　　・テスト実行（操作、駆動） 　　　・テスト結果検証（判定） 　　　・テスト結果報告 | – |
| 4 | 結合テスト  エビデンス | 結合テストのエビデンスも、単体テスト同様にテスター以外の人間が  見てもわかりやすいものである必要がある。   ＜テストエビデンスを取得する目的＞ 　・テスト担当者が結合テストを行ったことを証明する 　・レビュー担当者が結合テストの結果をチェックする際に使用する 　・総合テスト等の後続工程で不具合が発生した際の分析に使用する  　結合テストのエビデンスの注意点として、  　画面から実行した結果を確認する場合は、画面の結果とデータベースの  　データの両方がそろってはじめて十分なエビデンスと言える。  　データベースのデータをExcelシートに張り付けて保存する場合には、  　セルの形式が数字型だと数字の頭のゼロは消えてしまうため注意が必要。  　この事象はセルの形式を文字型にすれば回避できる。 | – |
| 5 | 不具合一覧 | **・不具合を管理する** 　不具合対応のフローは、大まかには以下のようになる。 　　①不具合の検知 　　　→②原因調査 　　　　→③修正（または対応不要のため完了） 　　　　　→再テスト 　　　　　　→対応完了 　　　　　　　（不具合が解消しない場合は再度原因調査に戻る）  　上記の①から③に関する情報を不具合一覧に記録する。　 　再テストの結果はテスト仕様書に記載するが、不具合のチケットは、 　再テストの結果、不具合が解消されたことを確認してからクローズする。 　以下に、不具合の管理に必要な情報を記載する。 　①不具合の検知時に記載する情報 　　・不具合の発生日 　　・不具合の事象（簡潔に記載） 　　・不具合の詳細内容 　　・再現方法 　　・検知者 　 　②不具合の原因調査に関する情報 　　・調査担当者 　　・調査結果 　　・品質分析に関する情報 　　　・不具合原因 　　　（設計不備、実装不備、環境構築不備、ﾃｽﾄ手順の不備、ﾃﾞｰﾀ不備） 　　　・不具合を埋め込んだ作業工程 　　　　（要件定義・基本設計・詳細設計・実装・テスト） 　　 　③不具合の修正に関する情報 　　・修正・対応担当者 　　・対応内容 　　・対応見積もり工数 　　・対応開始予定日 　　・対応完了予定日 　　・修正内容（対象ソースコード、その内容） 　　・修正理由 　　 　・作業のステータス管理情報 　　・進捗状況 　　　（調査中、修正中、結合テスト環境へﾘﾘｰｽ済み、再ﾃｽﾄ完了、対応不要） 　　・作業進捗率（0%~100%） 　　　（例）未着手：0% 　　　　　　調査中：10% 　　　　　　修正中：20%~70%（作業状況に応じた割合） 　　　　　　結合テスト環境へリリース済み：80% 　　　　　　再テスト完了／対応不要：100%　 　 **・BTS（Bug Tracking System／バグ管理システム）を利用する** 　不具合を管理するためには、情報を記録するためのツールが必要になる。 　以前はExcelの管理表を使うことが一般的であったが、 　現在はBTS（Bug Tracking System／バグ管理システム）が普及している。 　代表的なツールとしては、JiraやRedmineがある。 　従来のExcelの管理表よりもBTSが優れている点の例をあげる。 　＜BTSの優れている点＞ 　・大人数／異なる拠点間での情報共有が容易になる 　　・BTSはWEBブラウザから情報にアクセスできるため、 　　　専用のアプリケーションを個々人の端末に用意する必要がない。 　　　さらに、個々人が自分専用のViewを作成できるため、自分に 　　　必要な情報を抽出することが容易である。 　　　Excelもファイルを共有設定をすることで情報の共有は可能だが、 　　　誰か一人がExcelシートの表示設定を変更してファイルを保存すると、 　　　他の人もその影響を受けるため、都度表示設定を変更する手間がかかる。 　　・拠点が異なるとFileServerのアクセス権がそれぞれ異なる 　　　などの理由で共通のExcelファイルにアクセスできない場合が 　　　あるが、BTSを利用すればプロジェクト内に限定したアクセス権限 　　　を設定できる。 　　　FileServerの権限設定は全社レベルで設定していることが多く、 　　　これに比べるとBTSで権限設定をする方が影響範囲が狭く容易。 　・チケット駆動開発に有効 　　BTSのチケット（＝不具合）をガントチャートで一覧表示するなど、 　　用途に応じた形式のViewを用いることでチケット駆動開発が用意になる。  **・BTSのデメリットを克服する** 　BTSには優れている点がある一方で、BTSにはデメリットもある。 　BTSをデフォルト機能のままプロジェクトを運営を開始すると、 　開発メンバーが不便な経験をすることになる。 　従来のExcel管理に比べて、BTSは情報入力の手間が増えるという 　負の面がある。 　Excelはデータの編集に関しては非常によく出来たツールであり、 　WEBブラウザからデータを入力するBTSはExcelに比べると 　ユーザビリティは低い。 　この問題を克服するために、入力機能のカスタマイズに関する自由度が 　高いBTSを選択する。 　（すでに利便性の高いプラグインが存在しているとなお良い） 　例えば、JiraやRedmineはREST-APIが公開されており、これを利用 　することで、ExcelファイルやCSVファイルでチケットを一括更新する 　ことができる。 | – |
| 6 | 結合テスト結果  報告書 | テスト報告書に記載する情報の観点を以下に記す。 **・テスト結果の概況報告** 　テスト結果を総括して、  　テスト終了基準の中で未達成の事項がないか記載する。 　未達成の事項がある場合は残課題対応の予定を記載する。 　問題点は端的に記載し、詳細は後述のパートに記載する。  **・詳細の報告** 　・テスト進捗に関する報告の観点 　　・すべてのテストケースの検証が完了しているか。 　　・完了していない場合はその理由は何か。 　　・完了していないものが原因で総合テストの開始に影響が出ないか。  　・不具合解消状況に関する報告の観点 　　・すべての不具合が解消しているか。 　　・解消していない場合は、個別にその内容を報告する。 　　・未解消の不具合が原因で総合テストの開始に影響が出ないか。  　・品質分析に関する報告の観点 　　・テストケースの設定漏れはないか。 　　（全機能・非機能の要素を対象にしているか） 　　・設定したテストケースの密度は適切か。 　　（テストケース密度が荒すぎないか） 　　・テストケース密度に異常値（密度が低すぎる）がある場合、 　　　その理由は何か。 　　・テストケース密度が低い機能に対して追加のテストが必要ないか。  　・障害の分析に関する報告の観点 　　・障害発生件数が多すぎる機能、少なすぎる機能がないか。 　　・件数の過多・過小が起きている機能について、追加のテストの 　　　必要はないか。 | – |
| 7 | リリース手順書 （更新） | リリースリハーサルを通してタイムチャートや手順の精度を向上させる。 　[詳細設計工程の作成手順](http://open-process.jp/home/d%e8%a9%b3%e7%b4%b0%e8%a8%ad%e8%a8%88%e3%80%80%e6%88%90%e6%9e%9c%e7%89%a9%e4%bd%9c%e6%88%90%e8%a6%81%e9%a0%98#releaseDetail)を参照。 | – |
| 8 | 稼動確認手順書 （更新） |
| 9 | 移行データ （更新） |
| 10 | ﾘﾘｰｽﾀｲﾑﾁｬｰﾄ  （更新） |