

改訂版

組込みソフトウェア向け 開発プロセスガイド

独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター 編著

本書の内容に関するお問い合わせについて

このたびは翔泳社の書籍をお買い上げいただき、誠にありがとうございます。弊社では、読者の皆様からのお問い合わせに適切に対応させていただくため、以下のガイドラインへのご協力をお願い致しております。下記項目をお読みいただき、手順に従ってお問い合わせください。

● お問い合わせの前に

弊社Webサイトの「正誤表」や「出版物Q&A」をご確認ください。これまでに判明した正誤や追加情報、過去のお問い合わせへの回答(FAQ)、的確なお問い合わせ方法などが掲載されています。

| | |
|--------|---|
| 正誤表 | http://www.seshop.com/book/errata/ |
| 出版物Q&A | http://www.seshop.com/book/qa/ |

● ご質問方法

弊社Webサイトの書籍専用質問フォーム(<http://www.seshop.com/book/qa/>)をご利用ください(お電話や電子メールによるお問い合わせについては、原則としてお受けしておりません)。

※質問専用シートのお取り寄せについて

Webサイトにアクセスする手段をお持ちでない方は、ご氏名、ご送付先(ご住所/郵便番号/電話番号またはFAX番号/電子メールアドレス)および「質問専用シート送付希望」と明記のうえ、電子メール(qaform@shoeshisha.com)、FAX、郵便(80円切手をご同封願います)のいずれかにて「編集部読者サポート係」までお申し込みください。お申し込みの手段によって、折り返し質問シートをお送りいたします。シートに必要事項を漏れなく記入し、「編集部読者サポート係」までFAXまたは郵便にてご返送ください。

● 回答について

回答は、ご質問いただいた手段によってご返事申し上げます。ご質問の内容によっては、回答に数日ないしはそれ以上の期間を要する場合があります。

● ご質問に際してのご注意

本書の対象を越えるもの、記述箇所を特定されないもの、また読者固有の環境に起因するご質問等にはお答えできませんので、予めご了承ください。

● 郵便物送付先およびFAX番号

| | |
|-------|---------------------|
| 送付先住所 | 〒160-0006 東京都新宿区舟町5 |
| FAX番号 | 03-5362-3818 |
| 宛先 | (株)翔泳社 編集部読者サポート係 |

.....
※本書に記載されたURL等は予告なく変更される場合があります。

※本書の出版にあたっては正確な記述につとめました。著者や出版社などのいずれも、本書の内容に対してなんらかの保証をするものではなく、内容やサンプルに基づくいかなる運用結果に関してもいっさいの責任を負いません。

※本書に記載されている会社名、製品名は、各社の登録商標または商標です。

.....
※本書ではTM、®、©は割愛させていただいております。

はじめに

ESPR Ver.2.0 発行によせて

SECでは2006年度に組込みソフトウェア開発を対象として開発プロセスを整理しESPR Ver.1.0として発行しました。このESPR Ver.1.0は我々の予想をこえて、大変に多くの組込みソフトウェア技術者の方々にご利用いただけたと考えています。ESPR Ver.1.0は近年、大規模化や複雑化が進む組込みソフトウェア開発をより円滑に進めたいという声のもと、ソフトウェア・エンジニアリング・プロセス (SWP) とサポート・プロセス (SUP) の一部を先行して検討し公開しました。

しかしながら、製品として組込みシステムをとらえた場合には、ソフトウェア開発の前段階に相当するシステム設計などのシステム・エンジニアリング・プロセス (SYP) などを抜きに考えることは不可能です。このためSECではESPR Ver.1.0の検討に携わっていただいた方々に再度ご協力いただき、2006年度にシステム・エンジニアリング・プロセス (SYP) の整理を進め、ESPR Ver.2.0を策定しました。このESPR Ver.2.0は、上記のシステム・エンジニアリング・プロセス (SYP) の追加以外にもESPR Ver.1.0をベースに下記のような追加や修正を施しています。

- ・システム・エンジニアリング・プロセス (SYP) を追加
- ・システムの安全性や信頼性を実現するためのセーフティ・エンジニアリング・プロセス (SAP) を追加
- ・サポート・プロセス (SUP) の一部を追加
- ・ESPR Ver.1.0の誤りについて修正

このようにESPR Ver.2.0ではESPR Ver.1.0をベースに、組込みシステム開発を円滑に進めるためのノウハウを集めたものとなっています。ESPR Ver.1.0と同様、より多くの組込みシステム技術者の皆様に参考にしていただければと思います。

2007年11月

IPA SEC 組込みソフトウェア・エンジニアリング領域

上梓によせて

「品質の高い組込みソフトウェアを効率的に開発するにはどうしたらよいでしょうか？」

ソフトウェア・エンジニアリング・センター（SEC、所長：鶴保証城）の業務に携わってきたこの2年間で恐らく最も多くいただいた質問の1つのような気がする。この質問への答えとして、SECでは組込みソフトウェアの開発力向上を目指すソリューションとして、開発に携わる人材スキルの強化と組込みソフトウェア向けのソフトウェア・エンジニアリング手法整備を推進してきた。

本書はこうしたソリューションの1つとしてSECの組込みソフトウェア・エンジニアリング領域で検討を進めてきた「組込みソフトウェア向け開発プロセス」を整理したものである。わが国の組込みソフトウェアはこの数年で急激に規模・種類とも増加した結果、開発作業が煩雑化し現場で混乱をきたしているケースが散見されている。本書では組込みソフトウェアを開発する上で必要となる作業や基本動作を現場の技術者の方にも分かる言葉で体系的に整理したものである。本書を参考に、それぞれの組織、プロジェクト、個人の作業を見つめなおしていただき、それぞれに適したより効率的な開発スタイルを作り上げていただければ幸いである。

本書の発行に際しては、経済産業省とSECとで組織する組込みソフトウェア開発力強化推進タスクフォース（委員長：門田浩、副委員長：田丸喜一郎）の開発プロセス技術部会を中心に約2年を費やして原案の検討を進めた。部会メンバはそれぞれの企業や大学においてソフトウェアプロセスの実践や研究に長年携わってきたわが国のこの分野を代表する専門家であり、こうした方々の協力のもと本書を発行できることに感謝したい。

「ソフトウェア開発に銀の弾丸はない」と言われて久しいが、ソフトウェア開発に散在するさまざまな課題を一発の弾丸で解決することは難しい。まずは組込みソフトウェアの開発プロセスに潜む魔物退治のためのソリューションの1つとして本書を活用していただければ、関係者一同、望外の喜びである。

2006年10月

IPA SEC 組込みソフトウェア・エンジニアリング領域
平山雅之、山崎太郎、室修治

まえがき

我々の身の回りにはさまざまな情報機器が溢れています。これらの情報機器の多くはいわゆる組み込みシステムによってその機能を実現しており、その中核をなすものが組み込みソフトウェアです。これらの組み込みソフトウェアを効率的に開発し、かつその品質を確実なものとしていくためには、開発の段階で適切な作業を適切な順序で実施することが必須となります。本書は、組み込みソフトウェアの開発を円滑に進めるための標準的な作業やベストプラクティスを『**組み込みソフトウェア向け 開発プロセスガイド**』（英語名＝**ESPR：Embedded System development Process Reference**）として整理したものです。

本ガイドの位置づけおよび構成

本ガイドは、組み込みソフトウェア開発に従事する開発リーダーやマネージャの方々が、自部門や担当プロジェクトの開発プロセスを検討する際の参考にしていただくことを目的としています。

本ガイドは組み込みソフトウェア開発プロセスの普及のために、以下に示す3つのパートで構成されています。

Part 1 組み込みソフトウェア向け 開発プロセスガイド：解説編

Part 2 組み込みソフトウェア向け 開発プロセスガイド：技術編

Part 3 組み込みソフトウェア向け 開発プロセスガイド：活用編

備考

本ガイドは、経済産業省 組み込みソフトウェア開発力強化推進委員会 エンジニアリング領域 開発プロセス技術部会で検討を重ね整理したものです。

目次

| | |
|-----------|-----|
| はじめに..... | iii |
| まえがき..... | v |

| | | |
|---------------|--------------------------|----------|
| Part 1 | 解説編 | 1 |
| 1.1 | 開発プロセスとは..... | 2 |
| 1.2 | 開発プロセスガイドの目的と位置づけ..... | 3 |
| 1.3 | 想定する利用者・利用方法と得られる効果..... | 5 |
| 1.4 | 開発プロセスガイドの構造..... | 7 |
| 1.5 | 本ガイドの利用に関する注意事項など..... | 12 |
| 1.6 | 関連する規格など..... | 14 |

| | | |
|---------------|---------------------------------|-----------|
| Part 2 | 技術編 | 17 |
| 2.1 | 全体構成..... | 18 |
| 2.2 | プロセス定義書..... | 21 |
| | システム・エンジニアリング・プロセス (SYP)..... | 21 |
| | ソフトウェア・エンジニアリング・プロセス (SWP)..... | 62 |
| | セーフティ・エンジニアリング・プロセス (SAP)..... | 131 |
| | サポートプロセス (SUP)..... | 148 |
| 2.3 | ドキュメント・テンプレート例..... | 163 |

| | | |
|---------------|------------------------------|------------|
| Part 3 | 活用編 | 217 |
| 3.1 | 活用の手順..... | 218 |
| 3.2 | 組織 / 部門の開発プロセスの整備..... | 221 |
| 3.3 | 開発プロジェクトの工程設計..... | 222 |
| 3.4 | 開発プロジェクトの作業計画（工程設計の詳細化）..... | 227 |
| | 付録 | 233 |
| 付録 1 | 用語..... | 234 |
| 付録 2 | 規格対応表（本ガイド - X0160）..... | 238 |
| 付録 3 | アクティビティ / タスク / サブタスク一覧..... | 239 |
| 付録 4 | ESPR Ver.1.0 からの変更点..... | 242 |
| | あとがき..... | 245 |

Part **1**

解説編

| | |
|-------------------------|----|
| 1.1 開発プロセスとは | 2 |
| 1.2 開発プロセスガイドの目的と位置づけ | 3 |
| 1.3 想定する利用者・利用方法と得られる効果 | 5 |
| 1.4 開発プロセスガイドの構造 | 7 |
| 1.5 本ガイドの利用に関する注意事項など | 12 |
| 1.6 関連する規格など | 14 |

1.1 開発プロセスとは

■ プロセスとは

一般的にどのような業務でも、その業務の目的や目標を達成する上では、その途中でさまざまな作業を実施することが求められます。「どのような作業を実施していくか」について、その作業の入力や出力、作業内容などを整理したものをプロセスと呼びます。

■ ソフトウェア開発プロセスとは

ソフトウェアというモノをひとつの製品として完成させるためには、他の業務と同じようにさまざまな作業を積み重ねていくことが必要となります。このソフトウェアという製品を作りあげる上で、実施すべき作業を整理したものを**ソフトウェア開発プロセス**と呼びます。

一般的なソフトウェア開発プロセス

ソフトウェア開発プロセスについては、これまでも ISO/IEC12207 をはじめとするさまざまな考え方が提唱され実践されてきています。またこうしたソフトウェアから構成されるシステムの開発についても ISO/IEC15288 などの国際規格が提案され実施されています。

しかし、既存のソフトウェア開発プロセス標準は、対象とするソフトウェアを特に意識したものとなっていないため、これらを組込みソフトウェア開発に利用しようとした場合、使いづらいといった問題がありました。

組込みソフトウェア向け開発プロセスガイド

本ガイドは、組込みソフトウェア開発を対象に、その開発を進める上で必要な作業を開発プロセスの視点で整理したものです。

もちろん、組込みソフトウェアもソフトウェアのひとつに属するものですから、従来のソフトウェア開発プロセスの考え方を踏襲する部分があります。このため、本ガイドは、既存のさまざまなプロセス標準を参考に整理してあります。

■ ソフトウェア開発プロセスモデルとの違い

一般的にソフトウェア開発に関してはウォーターフォールモデル、スパイラルモデルといった開発プロセスモデルと呼ばれるものがあります。これらの開発プロセスモデルは、ソフトウェア開発で行うべき作業をどのような順序関係で実施するかといった、時間的な順序関係を考慮して、典型的な開発作業を抽象的に整理したものです。

これに対し、本ガイドやISO/IEC12207、15288などの規格は、個々のプロセスや作業の実施順序を規定することなく、開発に必要な作業を体系的に整理したものです。

1.2 開発プロセスガイドの目的と位置づけ

■ 組込みソフトウェア開発における開発プロセス

近年、さまざまなデバイス技術などの進化に伴い、ソフトウェアを内包したシステムでは、さまざまな機能実現が求められるようになってきています。これに伴い、これらの機能を実現するための組込みソフトウェアの規模は増加の一途を辿っています。従来の組込みソフトウェアは、その規模がそれほど大きくなかったため、開発プロセスをあえて意識しなくても何とか製品としての形を実現できていました。しかし、近年のソフトウェア規模の増大の中で、組込みソフトウェアあるいは組込みシステムに起因するさまざまなトラブルや問題が発生するようになってきており、その原因のひとつとして開発プロセスの問題がクローズアップされてきています。

開発プロセスの現状

実際の組込みソフトウェア開発の現場では、開発規模の急増や開発組織の複雑化に伴い、さまざまなゆがみが生ずるようになってきています。その原因として、

- ① 組織における開発プロセスが明確に定義できていない。
- ② 組織で採用している開発プロセスに漏れあるいは曖昧なところなどがある。
- ③ 開発プロセスの一部しか考えられておらず、全体としての作業が網羅できていない。

などがあります。このような状況の結果として、開発プロセスに起因する組込みシステムの品質問題が発生する傾向にあります。

組込みソフトウェア向け開発プロセスガイドの目的

本ガイドは、このような状況を解決し、高品質の組込みソフトウェアを効率的に開発することを目的として、開発プロセスを整備するための情報を整理したものです。

本ガイドは、ソフトウェアの開発という視点において、これまでの経験をベースに国際的な合意の下に策定された国際規格 (ISO/IEC12207、15288) を参考に、わが国の組込みソフトウェア開発の現場で実践されてきたソフトウェア開発作業 (ソフトウェア開発プロセス) に関する知見をブレンドして整理しました。

また、ESPR Ver.2.0では、次の2点についても従来のESPR Ver.1.0に対して新しい要素を組み入れてあります。

- ① 製品としての組込みシステムの視点から、一部、ソフトウェアからみたシステムという観点も組み入れ、組込みソフトウェア (システム) として開発の上流フェーズに相当するプロセスを組み入れました。
- ② 「安全・安心な組込みシステム」を実現するという視点から、システムの安全に関する国際規格 (IEC61508) なども参考に、安全なシステム開発のために求められる作業や基本動作も組み入れました。

本ガイドは、組込みソフトウェア開発に携わる皆さまの参考として利用していただくことを想定しています。

組込みソフトウェア向け開発プロセスガイドの特徴

この「組込みソフトウェア開発プロセスガイド」は、以下のような特徴を持っています。

- 特徴1:** ソフトウェア、システムの開発プロセスに関する国際規格を参考に、より具体的な作業のレベルで開発プロセスを整理している。
- 特徴2:** 組込みソフトウェア (システム) の開発において実施すべき作業項目と注意すべき事項を整理している。
- 特徴3:** 個々の作業の入出力を明示するとともに、作業内容についても具体的で理解容易な表現を用いている。

1.3 想定する利用者・利用方法と得られる効果

■ 想定する利用者・利用方法

本ガイドが想定する利用者、利用方法は下記のとおりです。

想定する利用者

本ガイドは、組込みソフトウェアの開発における下記の利用者を想定しています。

- ① 開発プロジェクトや開発組織などをマネジメントし、個々の開発案件において実際の開発プロセスや工程を検討・決定するマネージャやリーダー
- ② 組込みソフトウェアを開発する組織において、組織や部門の開発プロセスの標準や基本的な考え方を整備し、その運用を支援するメンバ
- ③ 組込みソフトウェアを開発する組織において、品質保証など、ソフトウェア開発を間接的に支える支援グループのメンバ

想定する利用方法

本ガイドは、組込みソフトウェア開発に必要な作業を中心に網羅的に整理したものです。本ガイドは個々の組織や部門、あるいは個々のプロジェクトに適した開発プロセスを整備する際に利用することを想定しています。

開発プロセスの整備に関しては、下記のような利用シーンが考えられます。

- ① ソフトウェアの開発プロセスなどが未整備で、新規に組織や部門の開発プロセス標準を決める場合
- ② 既に組織や部門の標準的な開発プロセスがあるが、実際の作業などとのずれが生じていて見直しが必要な場合
- ③ 従来の開発プロセスでは対応できないため、新たな開発プロセス標準が必要な場合

■ 得られる効果

本ガイドを利用して、組織や部門の開発プロセスを整備することで、下記のような効果を期待することができます。

開発プロセスを整備することによる効果

組織や部門の開発プロセスを整備すると、下記のような効果が期待できます。

- ① 組織や部門での組込みソフトウェア開発において、本来、実施すべき作業の漏れがないか確認したり、不要な作業を見直したりすることができます。
- ② 市場や顧客が求める機能や品質 / 信頼性を実現する上での品質作りこみなどの仕組みを確立することができます。
- ③ さまざまな開発メンバーが関わる大規模な開発プロジェクトにおいて、作業の分業化を進める場合、作業の分担や作業間の連携などを確実にすることができます。
- ④ 開発作業の一部を外部に委託する場合などに、委託先に提示する情報を明確にできます。たとえば、作業成果物、納品物、個々の作業内容や作業名称などがあります。

本ガイドを利用することによる効果

開発プロセスの整備において、本ガイドを参考として利用していただくことにより、下記の効果が期待されます。

- ① 国際規格やソフトウェア開発プロセスに関するこれまでの知見を参考に、組織や部門で必要な作業を効率的に整理し、組織や部門の開発プロセスとして整備することができます。
- ② 本ガイドで提示しているドキュメント・テンプレート例を参考に、個々の作業の結果整理や成果物の内容を検討することができます。
- ③ 組込みソフトウェア開発作業において、特に注意すべき事項などを組織や部門の開発プロセス検討の際に参考とすることができます。

1.4 開発プロセスガイドの構造

■ プロセスの構造

本ガイドでは、組込みソフトウェア開発に関わる作業を「プロセス」から「アクティビティ」「タスク」「サブタスク」へとブレイクダウンし、4階層で整理しています(図1.1)。

※以降、特に断りのない限り「プロセス」は「開発プロセス」を表します。

プロセス

プロセスは、組込みソフトウェア開発を進める上で、実施することが求められる作業をいくつかの塊(作業群)でとらえたものです。

① システム・エンジニアリング・プロセス

組込みソフトウェアがベースとなって構築される組込みシステムを取りまとめる作業を中心に整理したもの

② ソフトウェア・エンジニアリング・プロセス

実際のソフトウェア開発に関係する主たる作業をまとめたもの

③ セーフティ・エンジニアリング・プロセス

安全・安心な組込みシステムを作り上げるために実施すべき作業をまとめたもの

④ サポート・プロセス

ドキュメンテーションなど開発に付随して発生するさまざまな支援作業をまとめたもの

アクティビティ

アクティビティは、上記の**プロセス**を実施する際のより具体的な作業をまとめた作業群です。たとえば、ソフトウェア・エンジニアリング・プロセスを構成するアクティビティには、

- ・ソフトウェア要求定義
- ・ソフトウェア・アーキテクチャ設計
- ・ソフトウェア詳細設計
- ・実装および単体テスト
- ・ソフトウェア結合およびソフトウェア結合テスト

- ・ソフトウェア総合テスト

があります。

タスク

タスクは、個々のアクティビティを実施し、達成する上で実施が求められる作業をいくつかのグループに分けて整理したものです。

たとえば、「ソフトウェア要求定義」アクティビティは、

- ・ソフトウェア要求仕様書の作成
- ・ソフトウェア要求仕様の確認

の2つのタスクから構成されます。

サブタスク

サブタスクは、個々のタスクを実施する場合に、実施作業をそれぞれ作業ステップに近いレベルで整理したものです。

たとえば、「ソフトウェア要求仕様書の作成」タスクは、

- ・制約条件の確認
- ・ソフトウェア機能要求事項の明確化
- ・ソフトウェア非機能要求事項の明確化
- ・要求の優先順位付け
- ・ソフトウェア要求仕様書の作成

の5つのサブタスクから構成されます。個々のサブタスクに関しては、そこで実施すべき作業の詳細内容や組込みソフトウェアという観点での注意点などをまとめています。

時間概念の扱い

本ガイドで規定する作業（プロセス、アクティビティ、タスク、サブタスク）は基本的に時間的な順序性を持っていません。

このため、本ガイドを参考に実際の組込みソフトウェア開発の進め方を考える場合には、「開発作業の選定」と「工程設計」を行う必要があります（Part 3参照）。

開発作業の選定

開発対象となるシステムやソフトウェアが持つ特性、あるいは、開発組織や開発プロジェクトの特性などを考慮して、開発に必要な作業を決める必要があります。本ガイドは、組込みソフトウェアを新規に作成する場合を想定し、必要となる作業を整理したのですが、対象ソフトウェアの性質によっては、この中から必要な作業を選定し、対象に合うように一部チューニング（調整）をするといった工夫が必要になります。

また、既存の組込みソフトウェアの改造や流用などが中心となる開発の場合には、本ガイドで整理してあるプロセスや作業の中から必要なものを取捨選択して利用していくといった工夫が必要になります。

工程設計

実際の開発プロジェクトでは、開発の開始から製品出荷までの時間や期間を考える必要があります。このため、作業の選定で検討した実施すべき作業を実際の実開発時間軸上に割り付ける作業を**工程設計**と呼びます。この工程設計の中で、実際の作業の順序関係や作業の並行性なども考慮していきます。

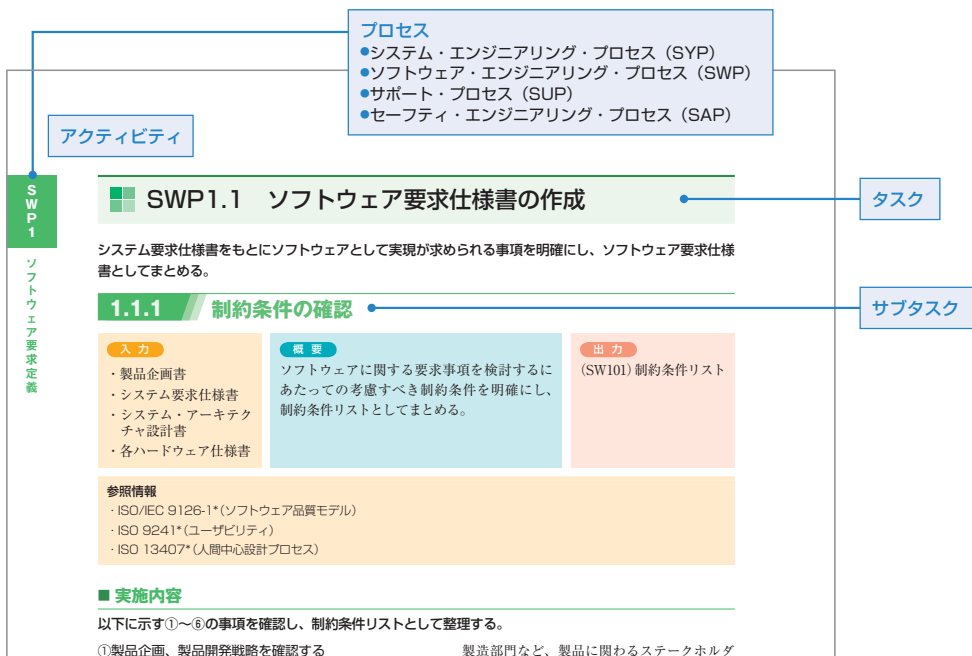


図 1.1 プロセスの構造

■ 組込みシステムの構成

組込みシステムは、「単一の機器」「複数の機器の集合」「サーバ側と一体」など、いろいろなおとらえ方ができますが、本ガイドでは「単一の機器」を組込みシステムとしています。

また、それぞれの組込みシステムはシステムで提供する機能を中心として、機能ブロックという単位を考え、それらを実現するための組込みソフトウェアについて機能ユニット、プログラムユニットへとブレイクダウンし、階層的にとらえています(図1.2)。

組込みシステム

組込みシステムは、1つ以上の機能を持った単一の機器です。たとえば、電話、FAX、コピー機、スキャナ、プリンタ機能を持つシステムが、機器が別々なら各機器を組込みシステムと呼んでいます。なお、それぞれの機能が機器として一体化されていれば、その一体化された複合機が組込みシステムの単位となります。組込みシステムは、1つ以上の組込みソフトウェアと1つ以上のハードウェアで構成されます。

機能ブロック

通常、組込みシステムは複数の機能を提供します。これらの個々の機能はそれを実現するハードウェアと組込みソフトウェアから構成されます。

組込みソフトウェア

組込みソフトウェアは、組込みシステムに組み込まれる、あるまとまったソフトウェアの単位です。組込みソフトウェアは機能ユニットで構成されます。

機能ユニット

機能ユニットは、組込みソフトウェアを構成する機能の単位です。機能ユニットは、プログラムユニットで構成されます。

プログラムユニット

プログラムユニットは、機能ユニットを構成するプログラムの最小単位(たとえばコンパイルやテストの単位)です。

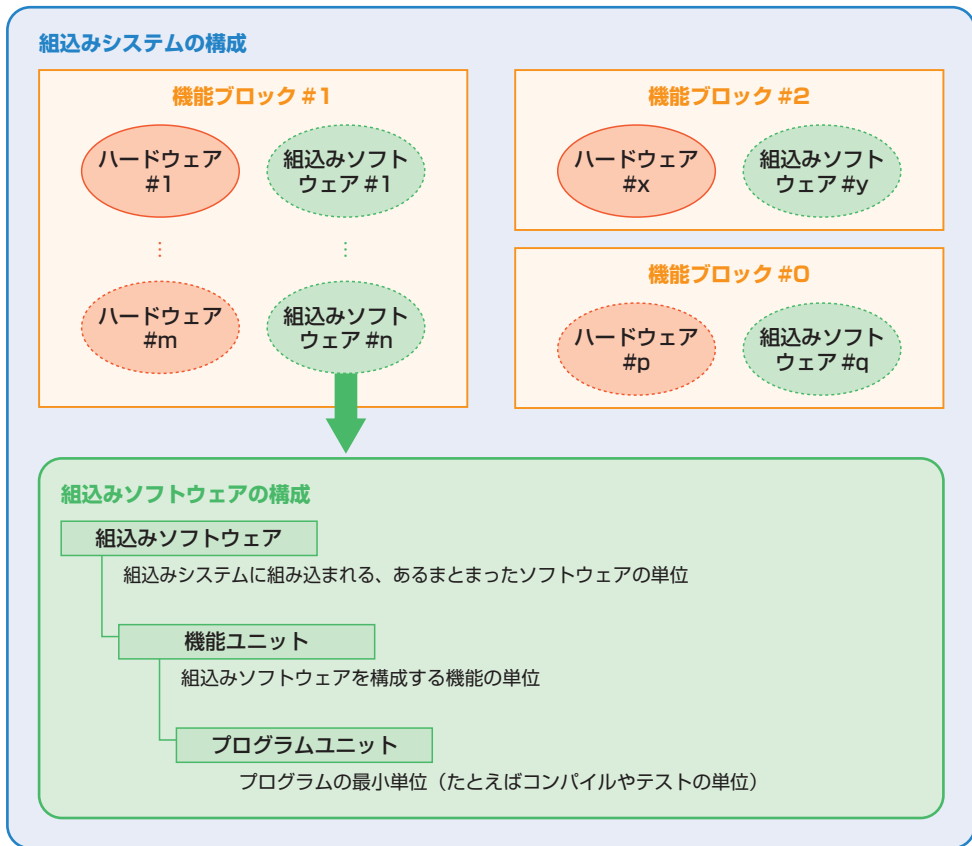


図 1.2 組込みシステムの構成

1.5 本ガイドの利用に関する注意事項など

参考情報としての扱い

本ガイドは、組込みソフトウェア開発に関して必要となる作業を整理してあります。

一般的に国際規格 (ISO など) と日本工業規格 (JIS) は、個々の規格が設定しているスコープに対して、その規格への準拠などを求める性格を有しており、規格への準拠を確認する手段 (たとえば、認定、認証など) と合わせて検討される場合があります。本ガイドを策定するにあたり、参考としている ISO/IEC12207、15288 などはいずれもこの範疇に入る規格です。

これに対し、本ガイドはこうした性格を有するものではなく、本ガイドの英語名 (ESPR : Embedded System development Process Reference) が示すとおり、あくまでも参照情報として、組込みソフトウェア開発における標準的な開発プロセスを整理したものです。したがって、本ガイドに対する準拠性に関するいかなる認証や認定といった枠組みも想定するものではありません。

プロセス定義の扱い

上記の策定に関する趣旨を踏まえ、本ガイドで提示しているプロセス、アクティビティ、タスク、サブタスクや、個々の作業の詳細は、あくまでも参考情報です。このため、特定の利用方法を指定するものではありません。

ドキュメント・テンプレート例の扱い

本ガイドでは、実際の開発作業の一助になるように、個々のタスクやサブタスクレベルでの作業を実施した結果を整理する際のドキュメントのテンプレート例を用意してあります。このドキュメント・テンプレートに関しても、あくまでも参考情報として添付してあり、特定の利用方法を指定するものではありません。

■ 用語ならびに概念の整理に関する注意点

本ガイドは、「現場の技術者に受け入れられやすい用語ならびに概念の整理」を編集方針としています。この方針をできる限り守るために、プロセス定義などで用いている用語や各プロセス、アクティビティなどの概念整理が既存の国際規格などとは若干異なっている部分があります。

しかしながら、これらに関しては、本ガイドの巻末の付録に対対応表を掲載することで、既存の国際規格などとの対応を明示する形としてあります。

用語に関する注意点

既存のISO/IEC12207などは、それぞれ日本工業規格(JIS)のJIS X0160などとして翻訳発行されています。

本ガイドの策定の過程では、これらの国際規格やこれらをベースにしたJIS規格を参考にしていますが、一部の用語については、実際の開発現場の意見をもとに、より身近な言葉に置き換えて表現してあります。

プロセスの概念整理に関する注意事項

本ガイドでは、組込みソフトウェア開発に関わる作業を、

- ・システム・エンジニアリング・プロセス
- ・ソフトウェア・エンジニアリング・プロセス
- ・セーフティ・エンジニアリング・プロセス
- ・サポート・プロセス

の4つの枠に入れる形で整理してあります。

本ガイドでは、こうした国際規格のプロセスカテゴリを参照しつつも、組込みシステム構築に関する直接作業(=システム・エンジニアリング・プロセス)、組込みソフトウェア開発に関する直接作業(=ソフトウェア・エンジニアリング・プロセス)、システムの安心・安全を担保するための作業(=セーフティ・エンジニアリング・プロセス)、これらの開発作業を支える支援作業(=サポート・プロセス)、という考え方で整理してあります(図1.3)。特に、支援作業に関しては、開発マネジメントから受発注管理にいたるまで多様な作業がありますが、ESPR Ver.2.0では、下図に示したアクティビティについて整理してあります。



図 1.3 本ガイドが規定しているプロセスと作業

1.6 関連する規格など

■ 開発プロセスに関する規格

ソフトウェアやシステムの開発プロセスについては、ISO/IEC12207、15288などが制定されています。

これらの規格については、本ガイドの中のRelated Standardsをあわせてご参照ください。

ISO/IEC12207 : 1995 Software Life Cycle Processes

ソフトウェア製品の開発に関わるプロセスを体系的に整理した国際規格です。国内では、JIS X0160ソフトウェアライフサイクルプロセスとして翻訳JISが発行されています。

この規格は、ソフトウェア開発に関する作業を整理し、その作業の呼び名などを統一することを目的として制定されたものです。対象とするソフトウェアは特定していませんが、検討のベースは、いわゆる「エンタプライズシステムの開発、特に受注者と発注者が存在する二者間契約」を念頭においています。

ISO/IEC15288 : 2002 System Life Cycle Processes

この規格は、複数のソフトウェアが同居し、あるいは、ハードウェアなども同居してサービスを提供するシステムを対象として、その開発作業を開発プロセスの視点で整理したものです。国内では、JIS X0170システムライフサイクルプロセスとして翻訳JISが発行されています。概念的には、ISO/IEC12207の対象をシステムにまで広げたものと言えますが、この規格が対象としているシステムは、特に組込みシステムなどを意識しているという形にはなっていません。

■ 安全・安心に関する規格

システムや製品の安全・安心に関してはさまざまな規格(国際規格、国内規格・法令など)が制定されています。中でも近年、電子・電機関連のシステムで計算機を利用したシステムを対象に機能的な安全を推し進めるための国際規格IEC61508などが注目を集めています。

IEC61508 : Functional safety of electrical / electronic / programmable electronic safety-related systems

計算機などを内包し高い安全性を求められるシステム(プラント制御や自動車など)について、その開発過程で安全性実現の視点から実施すべき作業を整理した国際規格です。この規格は特定の製品ドメインを意識したものとはなっておらず、安全性実現に向けた一般的な要求事項が記載されています。個々の製品分野での対応については、この規格を参照規格として分野ごとの安全性規格を別途制定する形で運用されつつあります。

■ ユーザビリティ向上に関する規格

最近の組込みシステムは機能の多様性が1つの特徴になっています。こうした多機能のシステムの使い勝手を向上させることを目的とした規格として、ISO13407、9241をはじめとして多数制定されています。ここでは、開発プロセスにも関係する規格としてISO13407を紹介します。

ISO 13407:1999 Human-centered design processes for interactive systems

コンピュータを利用したインタラクティブシステムを開発する開発過程の中で、システムの利用者である人間に着目した要素を考慮した設計(人間中心設計)の視点をどのように織り込んでいくかについて記載した規格です。この規格では人間中心設計に関して、①人間中心設計の原則(考え方)、②人間中心設計の工程計画、③人間中心設計活動、といった側面について規定しています。

■ ソフトウェア品質に関する規格

ISO 9126s: Software Engineering - Product Quality

ソフトウェア製品の品質とは何かについて、その概念を規定した規格です。代表的な特性として、機能性・信頼性・使用性・効率性・保守性・移植性の6つを規定し、その詳細定義を与えています。

また、こうした品質をどのように計測し可視化していくかについても言及しています。

■ 国際規格と本ガイドの関係

これらの国際規格は、開発プロセスや安全性、ユーザビリティ、品質などに関する基本的な概念などを整理したものであり、本ガイド策定の参考として利用しています。ただし、一般的に、国際規格は、その適用の具体的な部分まで拘束する内容を規定しないことが多く、これらの国際規格でも、規格本文の記述は比較的抽象的なレベルにとどめてあり、具体性を排除した記述になっています。このため、本ガイドでは、より具体的に組込みソフトウェア開発という対象を想定し、これらの国際規格を解釈して具現化したものです。

Part 2

技術編

| | | |
|-----|----------------|-----|
| 2.1 | 全体構成 | 18 |
| 2.2 | プロセス定義書 | 21 |
| 2.3 | ドキュメント・テンプレート例 | 163 |

2.1 全体構成

本ガイドで示したプロセス定義は、右ページの図に示すような全体構成を持っています。大別すると、次のとおりになります。

- システム・エンジニアリング・プロセス (SYP)

組込みソフトウェアが組み込まれて動作する組込みシステムとしてとらえた場合のシステム要求や、システムとしての動作検証などの作業を整理したプロセスです。

- ソフトウェア・エンジニアリング・プロセス (SWP)

ソフトウェアとしての要求定義からソフトウェア総合テストまでソフトウェアを作る際の直接作業を整理したプロセスです。

- セーフティ・エンジニアリング・プロセス (SAP)

安全・安心な組込みシステムを作り上げるために実施すべき作業を整理したプロセスです。

- サポート・プロセス (SUP)

ソフトウェア開発を円滑に進めるために必要となる支援作業や間接作業を中心に整理したプロセスです。

『組込みソフトウェア開発プロセスガイド』（ESPR Ver.2.0）では、V1.0で規定したソフトウェア・エンジニアリング・プロセス (SWP) とサポート・プロセス (SUP) に加えて、システム・エンジニアリング・プロセス (SYP) およびセーフティ・エンジニアリング・プロセス (SAP) を規定しています。図2.1にV字モデルとプロセス/アクティビティの関係を示します。

- 用語解説

- 安全性、信頼性、リスクについて

安全性：システムが規定された条件のもとで、人の生命、健康、財産またはその環境を危険にさらす状態に移行しない度合い（JIS X 0134：システムおよびソフトウェアに課せられたリスク抑制の完全性水準）

信頼性：機能単位が要求された機能を与えられた条件の下で、与えられた期間実行する能力（JIS X 0014：情報処理用語—信頼性・保守性および可用性）

リスク：財産やインフラ、プライバシーなど、価値を有するモノの一部または全部を失い、結果として損失を出す可能性。

SYP :
システム・エンジニアリング・プロセス

SYP1 システム要求定義22
1.1 システム要求仕様書の作成
1.2 システム要求仕様の確認

SYP2 システム・アーキテクチャ設計32
2.1 システム・アーキテクチャ設計書の作成
2.2 システム・アーキテクチャ設計の確認
2.3 システム・アーキテクチャ設計の共同レビュー

SYP3 システム結合テスト43
3.1 システム結合テストの準備
3.2 システム結合テストの実施
3.3 システム結合テスト結果の確認

SYP4 システムテスト51
4.1 システムテストの準備
4.2 システムテストの実施
4.3 システムテスト結果の確認
4.4 システム開発の完了確認

SWP :
ソフトウェア・エンジニアリング・プロセス

SWP1 ソフトウェア要求定義63
1.1 ソフトウェア要求仕様書の作成
1.2 ソフトウェア要求仕様の確認

SWP2 ソフトウェア・アーキテクチャ設計76
2.1 ソフトウェア・アーキテクチャ設計書の作成
2.2 ソフトウェア・アーキテクチャ設計の確認
2.3 ソフトウェア・アーキテクチャ設計の共同レビュー

SWP3 ソフトウェア詳細設計89
3.1 機能ユニット詳細設計書の作成
3.2 ソフトウェア詳細設計の確認
3.3 ハードウェア仕様との整合性の確認

SWP4 実装および単体テスト99
4.1 実装および単体テストの準備
4.2 実装および単体テストの実施
4.3 実装および単体テスト結果の確認

SWP5 ソフトウェア結合テスト109
5.1 ソフトウェア結合テストの準備
5.2 ソフトウェア結合テストの実施
5.3 ソフトウェア結合テスト結果の確認

SWP6 ソフトウェア総合テスト120
6.1 ソフトウェア総合テストの準備
6.2 ソフトウェア総合テストの実施
6.3 ソフトウェア総合テスト結果の確認
6.4 ソフトウェア開発の完了確認

SAP :
セーフティ・エンジニアリング・プロセス

SAP1 安全性要求定義132
1.1 安全要求仕様書の作成
1.2 安全要求仕様の確認

SAP2 安全性テスト141
2.1 安全性テストの準備
2.2 安全性テストの実施
2.3 安全性テスト結果の確認

SUP :
サポート・プロセス

SUP1 プロジェクトマネジメント149
1.1 プロジェクト計画書の作成
1.2 プロジェクト実施状況の把握
1.3 プロジェクトの制御
1.4 プロジェクト完了報告書の作成

SUP2 品質保証152
2.1 品質目標の設定
2.2 品質管理方法の確定
2.3 品質の可視化に基づく品質制御

SUP3 リスクマネジメント154
3.1 リスクの洗い出しと把握
3.2 リスクのモニタリング
3.3 リスク対策の決定と実行

SUP4 文書化と文書管理

SUP5 構成管理156
5.1 構成管理対象物の把握
5.2 構成管理 / 変更管理履歴の管理

SUP6 問題解決管理158
6.1 問題の記録と原因分析
6.2 影響分析と対策立案
6.3 対策の実行
6.4 対策結果の確認

SUP7 変更管理160
7.1 変更要求情報の記録
7.2 変更による影響の分析
7.3 変更計画の立案と実施
7.4 変更結果の確認

SUP8 共同レビュー161
8.1 レビューの準備
8.2 レビューの実施
8.3 レビュー結果の確認とフォロー

SUP9 開発委託管理

SUP10 開発環境整備162
10.1 開発環境整備計画の立案
10.2 開発環境の構築
10.3 開発環境の維持

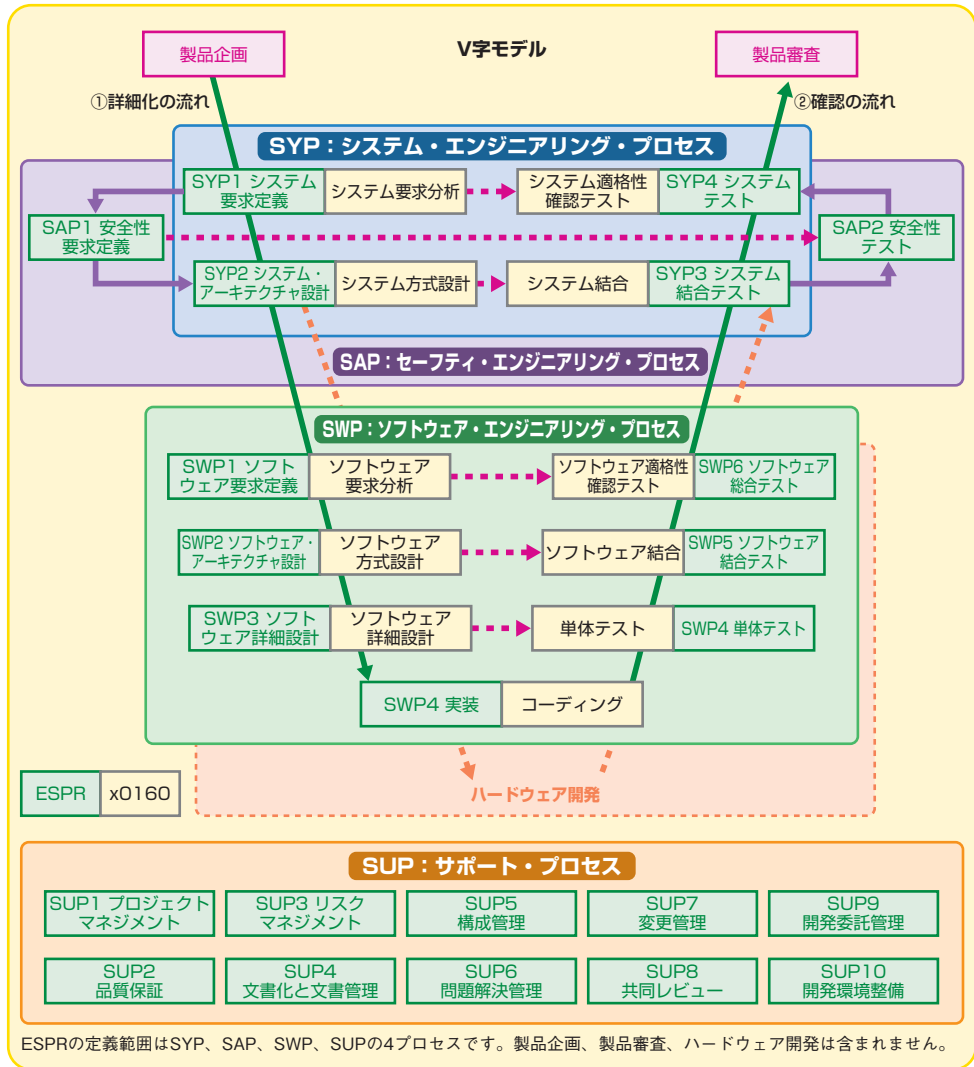


図2.1 V字モデルと開発プロセス

2.2 プロセス定義書

SYP：システム・エンジニアリング・プロセス

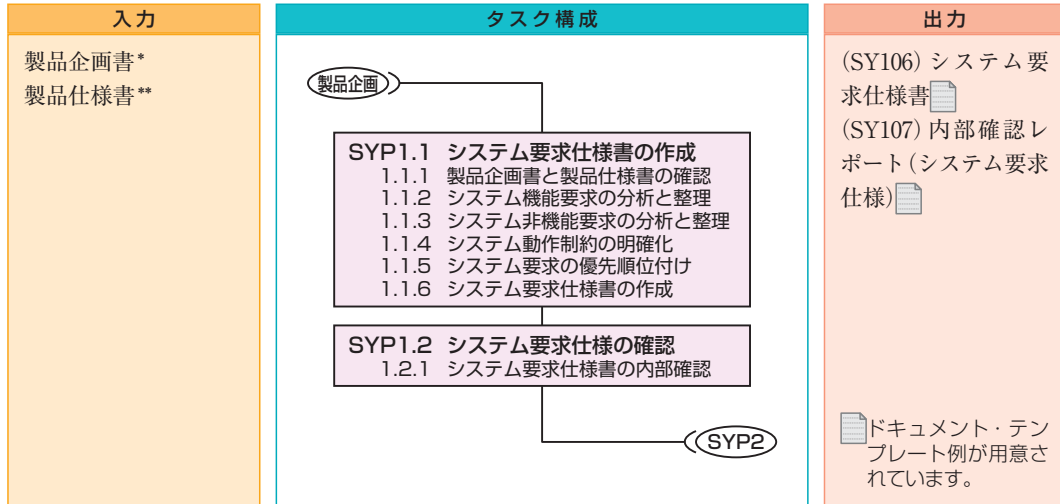
システム・エンジニアリング・プロセスは、組込みシステム開発に関する作業の中でも、ソフトウェアとハードウェアを含む製品としての要求定義からシステムテストまでの作業を定義しています。

このプロセスには、下記のアクティビティが含まれています。

| ID | アクティビティ | アクティビティの概要 | 構成するタスク |
|------|----------------|---|---|
| SYP1 | システム要求定義 | 当該製品を実現するためにシステムとして実現が必要となる要求を明確にする。 | SYP1.1 システム要求仕様書の作成 SYP1.2 システム要求仕様の確認 |
| SYP2 | システム・アーキテクチャ設計 | 開発する組込みシステムをどのように実現するか、ハード/ソフトの役割分担も含めて明確にする。 | SYP2.1 システム・アーキテクチャ設計書の作成 SYP2.2 システム・アーキテクチャ設計の確認 SYP2.3 システム・アーキテクチャ設計の共同レビュー |
| SYP3 | システム結合テスト | システムを構成するハードとソフトを組み合わせた際の機能ブロックが動作することを確認する。 | SYP3.1 システム結合テストの準備 SYP3.2 システム結合テストの実施 SYP3.3 システム結合テスト結果の確認 |
| SYP4 | システムテスト | システムの要求事項が実現できていることを確認する。 | SYP4.1 システムテストの準備 SYP4.2 システムテストの実施 SYP4.3 システムテスト結果の確認 SYP4.4 システム開発の完了確認 |

SYP1 システム要求定義

当該製品を実現するためにシステムとして実現が必要となる要求を明確にする。



■ 解説

このアクティビティは

- (1.1.1) 製品企画書と製品仕様書をもとに、製品要求仕様の内容を把握した上で、
- (1.1.2) システムに求められる機能面の要求を分析して明確にし、
- (1.1.3) また、システムに必要な非機能面の要求も合わせて明確にする。
- (1.1.4) 製品を実現するシステムについて、実際の動作環境などを考慮して、その動作条件や制約条件などを明確にする。
- (1.1.5) 実際のシステム開発では、開発期間やリソース、製品の動作環境などの制約があるため、これらを考慮したうえで、個々の機能、非機能要求に優先順位をつけていく。
- (1.1.6) これらの検討結果を整理し、システム要求仕様書を作成する。
- (1.2.1) 作成したシステム要求仕様書はあらかじめ確認ポイントを決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。

■ 留意事項

- ▶ システム要求の開始条件として以下の事項に留意する。
 - ・ 製品企画：製品戦略 (エンドユーザニーズなど) が明確になっている
 - ・ スケジュール：全体スケジュールは決まっている (発売日、対外マイルストーンなど)

● 用語解説

- * 製品企画書：製品のカタログレベルの内容で、製品戦略 (エンドユーザニーズなど) が明確になっている。
- ** 製品仕様書：製品の取扱説明書レベルの内容で、製品が提供するサービスが明確になっている。

- ▶ 組み込みシステムの場合、システムが動作する環境など外部環境の分析とそれに伴う異常処理対応の機能分析を特に重視する。
- ▶ 機能的な側面のみでなく性能や保守性など非機能的な側面の分析も重視する。
- ▶ 対象システムの外部で連携動作するシステムにも留意して要求を検討する。
- ▶ 製品としてみた場合のロングレンジの製品機能のあり方も考慮して要求仕様を決定する。

■【参考】手法およびツール

- ▶ シナリオ分析(ユースケース図、アクティビティ図など)

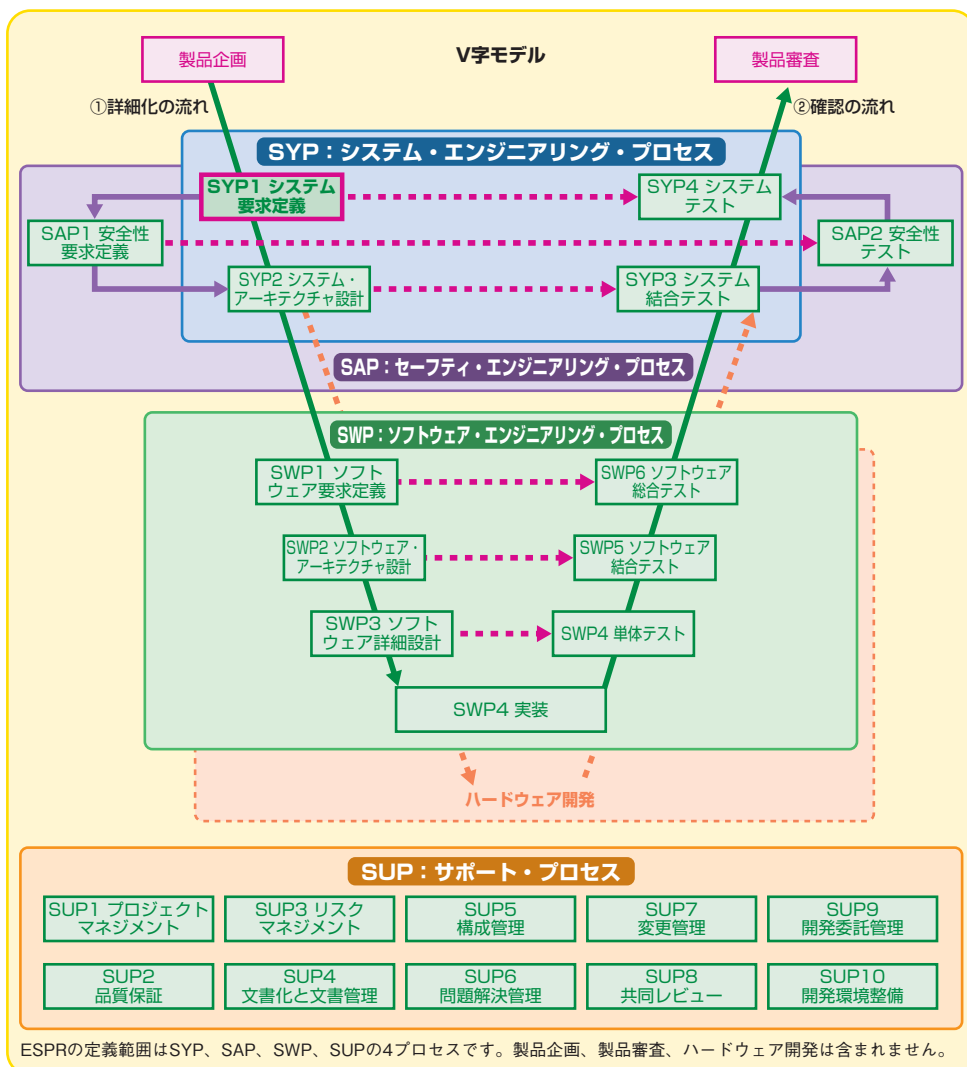


図2.2 V字モデルと開発プロセス (SYP1 システム要求定義)

■ SYP1.1 システム要求仕様書の作成

製品企画書および製品仕様書をもとにシステムとして実現が求められる機能項目を明確にし、システム要求仕様書としてまとめる。

1.1.1 製品企画書と製品仕様書の確認

入力

製品企画書
製品仕様書

概要

製品企画部門などで作成した製品企画書と製品仕様書に記載された内容を把握し確認する。

出力

(製品企画書・仕様書確認メモ)

■ 実施内容

- ① 製品企画書に記載されている以下の項目について確認する。
 - ▶ 製品の概要や特徴、既存製品からの機能面での差分など
 - ▶ 製品の市場や市場投入の時期（発売時期）
 - ▶ 中長期をみた製品戦略上の当該製品の位置づけ、重要度
 - ▶ 製品が関係する規格、規約、法律、法規
- ② 製品仕様書に記載されている以下の項目について確認する。
 - ▶ 製品のビジョンやコンセプト
 - ▶ 対象ユーザとシステム要求
 - ▶ 製品の利用シーンと利用コンテキスト*
 - ▶ 製品実現における制約事項

■ 注意すべき事項

- 製品企画書の確認で注意すべき事項としては以下のようなものがある。
 - ▶ 製品企画書は製品企画部門や営業部門などによって作成される製品に関する概要情報を整理したドキュメントであり、上記の記載事項がすべて盛り込まれているとは限らないことに注意する。
 - ▶ 製品企画書に記載された事項は、対象ユーザや市場動向などさまざまな要因によって変更される場合もあるため、その企画が作成された時期に注意する。
- 製品仕様書の確認で注意すべき事項としては以下のようなものがある。
 - ▶ 製品のビジョンやコンセプト
 - ・ 製品の戦略的な位置づけ
 - ・ 競合他社製品との優位性
 - ・ 製品を開発するサイドからみて、製品が提供する機能面やサービス面のコンセプトとともに、開発面でのQCDに関するコンセプト
 - ▶ 対象ユーザとシステム要求
 - ・ ユーザの年齢、性別、国籍、職業、居住地など

● 用語解説

- * **コンテキスト**：コンテキストとは文脈や背景あるいは前後関係といった関連性のこと。ここでは、開発する組込みシステムの目的、背景や戦略などが例として挙げられる。

- ・対象製品の利用頻度や既存製品の利用経験の有無
- ・メーカー(ベンダ)側からの要求事項
- ・典型的なユーザの要求事項
- ▶製品の利用シーンと利用コンテキスト
 - ・対象とするユーザが製品を利用する状況
 - 1) 通常の利用に関する状況
 - 2) 想定外の状況での利用
 - ・利用シーンやコンテキストでは製品の動作やユーザの製品操作に関係する要素
 - ・コンテキスト情報とユーザ操作などを含めた製品利用シナリオ(ユースケース図、アクティビティ図など)
- ▶製品実現における制約事項
 - ・製品開発に割くことのできる期間や開発コスト
 - ・製品の耐用年数や製品寿命
 - ・製品に期待される品質(信頼性、安全性、使い勝手など)
 - ・製品を利用する上で前提として利用されるシステムやハードウェア(周辺システムやハードウェア)
 - ・既存製品仕様との継続性に関する制約
 - ・既存システムの再利用に関する制約
 - ・セキュリティ、環境問題に関する制約
 - ・製品(販売)価格

1.1.2 システム機能要求の分析と整理

入力

製品企画書
製品仕様書

概要

製品仕様書の記載事項をもとに、製品の機能を実現するシステムを開発する立場で、システムに求められる機能要求事項を分析し整理する。

出力

(SY101) システム機能リスト
(SY102) システム機能動作マトリクス

■ 実施内容

- ①製品仕様書の中に記載された要求事項を分析し、製品を実現するシステムに求められる機能面の要求を洗い出す。
- ②機能要求の洗い出しについては、システムのユーザ、利用シーンを考慮したユースケース分析を行い、その結果も考慮して検討していく。
- ③機能要求については
 - ▶機能間の関係(動作の順序関係、動作の並行性など)も分析し、機能動作のマトリクスに整理しておく。
 - ▶既存システムの再利用や将来の再利用などを考慮し、個々の機能についての流用や変更の有無なども整理する。
 - ▶データ处理的な機能などについては個々の機能が関係するデータなども明確にしておく。
- ④ネットワークやバスを介したシステム(サブシステム)間の連携や外部インターフェースなどについても明確にしておく。
 - ▶通信プロトコルや形態、ネットワークを介してやり取りされるデータなども明確にしておく。
- ⑤ユーザインターフェースなどをもつシステムの場合には、表示や操作項目の整理や、ユーザインターフェースを司るデバイス類についても整理しておく。

■ 注意すべき事項

- ▶システムの機能要求については製品仕様書や製品企画書に記載された要求事項との対応関係が分かるように工夫しておく。
- ▶ユースケース分析ではユースケース・シナリオ、ユースケース図、アクティビティ図などを作成する。
- ▶製品仕様書として整理されたときの要求は、システムとして全体を見た場合には必ずしも十分とは限らない。このため、
 - ①サービスの視点から必要な機能を整理分析する
 - ②個々の機能の粒度を考え、機能を階層的に分割し整理する
- ③機能項目間の連携や関係を整理するといった分析を加え、システムとしての実現を見据えた機能要求を体系的に整理する。
- ▶機能の分析や整理では個別の機能で用いられるデータや製品と外部との関係に着目し、それらを介した機能間の関連や、時系列を意識したシステムや機能の状態遷移なども検討する。
- ▶1つのシステム内に複数のCPUが配置され機能を実現するマルチCPUタイプのシステムの場合には、内部の連携動作についての仕様も検討しておく。

1.1.3 システム非機能要求の分析と整理

入力

製品企画書
製品仕様書

概要

製品仕様書の記載事項をもとに、製品の機能を実現するシステムを開発する立場で、システムに求められる非機能要求事項を分析し整理する。

出力

(SY103) システム非機能リスト

■ 実施内容

- ①製品仕様書の中に記載された要求事項を分析し、製品を実現するシステムに求められる非機能面の要求を洗い出す。
- ②非機能面の要求には、システムとしての「信頼性」「効率性」「保守性」「移植性」や「使用性」などが含まれる。
(例)
 - ・システムに許容される不具合の数や程度
 - ・個々の機能の処理時間やレスポンスタイム
 - ・ユーザインタフェースの操作のし易さ
 - ・市場投入後の不具合修正などに関する保守の方法(リモートメンテナンスなど)
- ③ネットワーク接続などを前提とするシステムではセキュリティ面での要求事項も合わせて明確にしておく。

■ 注意すべき事項

- ▶非機能要求については、具体的な数値を決めることができるものはその数値を明記する。
(例)レスポンスタイムなど
- ▶システムの安全性などに関する項目は、システムが提供する機能の不具合によって引き起こされる影響や頻度なども検討しておく。
- ▶非機能要求については製品企画書、製品仕様書には明記されていないことも少なくない。このため、システムの利用シーンやコンテキスト情報を参考に、洗い出す必要がある。

関連プロセス：SAP1 安全性要求定義

1.1.4 システム動作制約の明確化

入力

製品企画書
製品仕様書

概要

製品を実現するシステムについて、実際の動作環境などを考慮して、その動作条件や制約条件などを明確にする。

出力

(SY104) システム動作制約リスト

■ 実施内容

- ①システムの動作環境を把握する。
 - ▶システムが設置されたり利用される場所などに起因する動作条件を明確にする。
 - ▶システムにセンサーなどを介して入力されるデータのレンジや値の範囲を明確にする。
(例)
「自動車」における「寒冷地仕様」など
- ②システムで処理するデータなどのサイズや特性、データの入出力のタイミングなども明確にしておく。
- ③システムの利用に付随する法的な制約、社会慣習面での制約なども明確にしておく。
- ④システム実装に関して、製品機器の寸法なども明確にしておく。**安全**
- ⑤他社の知的財産権、他社技術などとの関係も明確にしておく。
- ⑥システムを利用するハードウェアプラットフォーム(MPU、LSIなど)に関する制約も明確にしておく。

■ 注意すべき事項

- ▶通常、システムはある動作環境を想定して作られる。実際のシステムの利用で、この想定から外れた環境条件で利用された場合、システム障害が発生する 경우가少なくない。このため動作条件分析では出来る限りシステム利用の可能性を考慮しておく。

安全 : セーフティに関連する作業

1.1.5 システム要求の優先順位付け

入力

製品企画書
製品仕様書
(SY101) システム機能リスト
(SY103) システム非機能リスト
(SY104) システム動作制約リスト

概要

システムの要求事項について、実現の制約事項などを考慮して優先順位を決定する。

出力

(SY105) 優先順位付きシステム要求リスト

■ 実施内容

- ①(SY101)、(SY103)、(SY104)の事項を考慮し、システム要求に関する優先順位付けを行う。
- ②システムとしての実現事項に関する優先順位付けでは要求事項(機能、非機能)実現に関するコストや開発期間と実現した場合の製品価値(ユーザ側から見た価値も含めて)、実現性を参考に重要度を検討する。
- ③要求の優先度は最終的に高/中/低の3段階程度に分類するのが望ましい。

■ 注意すべき事項

- ▶ システムの要求事項の優先度付けでは
 - ・ 品質機能展開
 - ・ 要求事項の双対比較(AHP)などの考え方をとり入れた分析が有効な場合もある。
- ▶ また要求事項の中には相矛盾する事項が含まれるため、そのトレードオフ分析などを行う必要もある。
- ▶ 要求の中には実現性の観点から難しいものも含まれることがあるため、これらについての見極めも併せて行う。
- ▶ 要求事項の優先順位付けについては、必要に応じて再度、顧客も含めて関係するステークホルダ間での調整を行う。
- ▶ システム要求の優先順位については製品のシリーズ化なども考え、中長期的の視点からも検討する。

1.1.6 システム要求仕様書の作成


入力


製品企画書
製品仕様書
(SY101) システム機能リスト
(SY102) システム機能動作マトリクス
(SY103) システム非機能リスト
(SY104) システム動作制約リスト
(SY105) 優先順位付きシステム要求リスト

概要

システムの要求事項を整理しシステム要求仕様書を作成する。

出力

(SY106) システム要求仕様書 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

- ①(SY101) ～ (SY105)の事項を考慮し、システム要求仕様書を作成する。
- ②上記検討の段階で幾つかの代替案を並行して検討してきた場合には、最終案を選択決定する必要がある。
- ③システムの設計指針も合わせて整理する。

■ 注意すべき事項

- ▶ システム要求仕様書では
 - ・ システムの外側で関係するシステムやシステムが関係するハードウェア
 - ・ システムの外部との入出力情報やデータ
 - ・ システムで実現する機能/非機能(優先度をつけた)
 - ・ システム全体や個々の機能の動作制約(動作順序、動作優先度、並行動作)なども明確になっていることが望ましい。
 - ▶ 必要に応じてユースケース分析の結果(ユースケース図、ユースシナリオなど)も加えておくとうい。
 - ▶ システムの設計指針としては
 - ・ 設計に関する制約条件
 - ・ 最適な設計を行うための設計手法
 - ・ 実装を考えた場合のシステムの動作プラットフォーム
 - ・ 既存システムの再利用の可能性なども記載しておくとうい。
 - ▶ システム要求仕様書の段階で、未確定要因がある場合には、それを明記しておく。
 - ▶ 文書化での留意点
 - ・ 変更履歴を添付し、変更箇所なども明示する
 - ・ 作成文書についての責任所在を明示する
 - ・ 作成文書は構成管理および変更管理にて確実に管理する
- 関連プロセス：SUP5 構成管理、SUP7 変更管理

■ SYP1.2 システム要求仕様の確認

定義したシステム要求事項が製品としての要求事項を満たしていることを確認する。

1.2.1 システム要求仕様書の内部確認


入力


(SY106) システム要求仕様書

概要

システム要求仕様がシステムとして求められる事項を満たしているか確認し、内部確認レポートとしてまとめる。

出力

(SY107) 内部確認レポート(システム要求仕様) 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

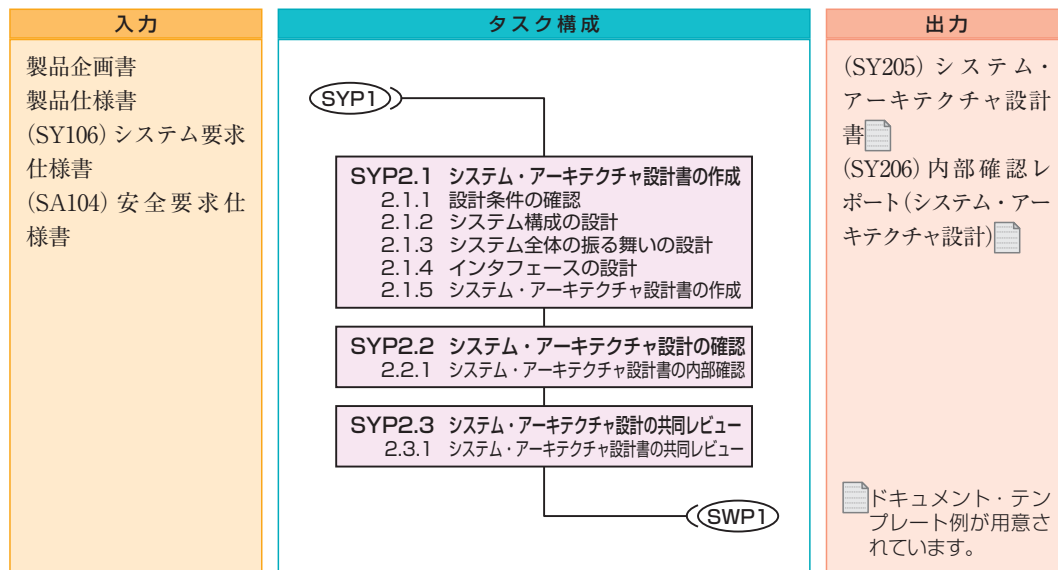
- ①以下の視点で、システム要求仕様書(SY106)の内部確認を行う。
 - ▶ システムの前提となっている動作環境や動作条件、コンテキストが明確になっているか。
 - ▶ 対象ユーザとその操作シナリオが明確になっているか。
 - ▶ 上記を踏まえてシステムで実現すべき機能/非機能面の要求事項が明確になっているか。
 - ▶ 上記要求事項は優先度が明確になっているか。
 - ▶ 未決定事項が明確になっているか。
- ▶ 仕様書に書かれた要求事項がビジネスの視点(事業性、新規性など)から見て適切か。
関連プロセス：SUP3 リスクマネジメント
- ②確認結果は内部確認レポート(SY107)として整理し、確認作業で指摘された問題およびその対応元を明記した上で、関係者に配布する。

■ 注意すべき事項

- ▶ システム要求仕様書の確認に際しては、
 - ・ システム開発に関係する開発者・技術者
 - ・ 当該システムの製品仕様書、製品企画書の検討に関わったメンバ
 - ・ 過去に類似のシステム開発に携わった技術者
 なども交えて確認を行うことが望ましい。
- ▶ 管理者は内部確認において指摘された問題事項については、解決策やその対応のためのアクションなどもあわせて記載しておくことが望ましい。
- ▶ 要求段階での問題は放置すると開発後半において大きな問題を生むため、この段階で内部確認レポートの情報は、プロジェクトマネージャ/開発リーダー、製品企画担当者などステークホルダ間で情報共有とコンセンサスを得ておくことが望ましい。

SYP2 システム・アーキテクチャ設計

開発する組込みシステムをどのように実現するか、ハード/ソフトの役割分担も含めて検討する。



■ 解説

このアクティビティは

- (2.1.1) (2.1.2) システム要求仕様をもとに実現すべき機能、非機能要求を確認する。また、要求実現上の制約などを仕様実現の視点で確認する。
- (2.1.3) システムを構成する機能をハードウェア、ソフトウェアそれぞれの視点から整理し、それぞれの役割分担を考慮してシステムとしての振る舞いと構成を検討する。
- (2.1.4) システムを構成する機能ブロック間およびシステム外部とのインタフェースを設計する。
- (2.1.5) これらの検討結果を整理し、システム・アーキテクチャ設計書を作成する。
- (2.2.1) 作成したシステム・アーキテクチャ設計書はあらかじめ確認ポイントを決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。
- (2.3.1) また、関係者を交えて共同レビューを実施し、レビューの結果を整理して共同レビュー報告書を作成する。

■ 留意事項

- ▶ 組込みシステムでは、実現コストや性能要求、信頼性・安全性、再利用性・拡張性など複数の視点から検討し、各機能についてソフトウェアで実現したほうが良い場合、ハードウェアで実現したほうが良い場合などを考える。
- ▶ 既存システムを構成する機能の一部再利用の

可能性も検討する。

- ▶ システム全体の見通しをよくするように図などを活用し、設計結果を可視化できるようにし

ておくことが望ましい。

■【参考】手法およびツール

- ▶ 設計モデリング手法
 - ・ UML (Undefined Modeling Language : 統一モデリング言語)
- ▶ シナリオ分析 (ユースケース図、アクティビティ図など)

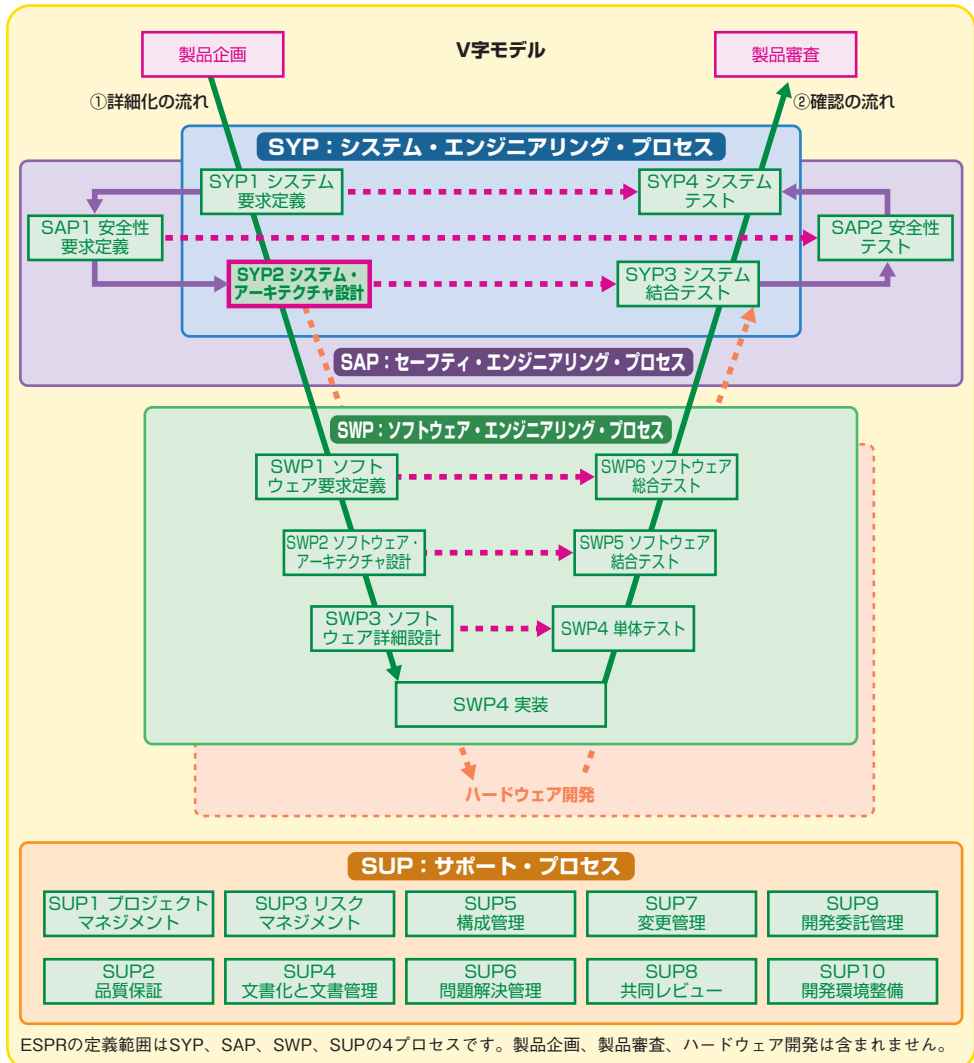


図2.3 V字モデルと開発プロセス (SYP2 システム・アーキテクチャ設計)

SYP2.1 システム・アーキテクチャ設計書の作成

システム要求仕様書で定義された要求の実現方法(システム・アーキテクチャ)をハード/ソフトの役割分担も含めて検討し、システム・アーキテクチャ設計書としてまとめる。

2.1.1 設計条件の確認

入力

製品企画書
製品仕様書
(SY106)システム要求仕様書
(SA104)安全要求仕様書

概要

システム・アーキテクチャを設計するにあたり、要求・条件などを確認する。

出力

(設計条件確認メモ)

■ 実施内容

- ①システム要求仕様書の記載内容を確認する。
 - ▶ システムの制約条件(動作制約含む)
 - ▶ システムの機能/非機能要求
- ②システムの動作環境についても再度確認する。
- ③既存システムの一部を再利用したり拡張する場合の利用条件なども確認する。
- ④システムの将来の機能拡張の可能性などについても確認する。
- ⑤今回開発するシステムの範囲を確認する。

■ 注意すべき事項

- ▶ 数値として把握できる設計条件は数値を示しておく。
- ▶ クロス開発を念頭に、開発環境と実行環境の相違点も整理しておく。

2.1.2 システム構成の設計

入力

製品企画書
製品仕様書
(SY106) システム要求仕様書
(SA104) 安全要求仕様書

概要

システムを構成する機能をハードウェア、ソフトウェアそれぞれの視点から整理し、機能ブロックとして抽出し設計する。

出力

(SY201) システム構成図
(機能ブロック図)
(SY202) 共通機能・データ一覧表

■ 実施内容

- ①システムを構成する大きなレベルの機能を洗い出し、これを機能ブロックとして抽出する。
- ②各機能ブロックについてハードウェア、ソフトウェアの役割分担を検討する。
- ③各機能ブロック間の関係を検討する。
- ④各機能ブロックのおおよその内容を検討し、システム要求仕様で求められる個別の機能を各機能ブロックに分配していく。
- ⑤システム全体から共通に利用される機能やデータを識別しておく。
これらを踏まえて、システムを構成する機能に関する機能ブロック図を整理する。なお、システムが担う処理やサービスによっては、システムで扱うデータに着目してシステム構成要素の抽出と分割を考えたほうが良い場合もある。

■ 注意すべき事項

- ▶ 各機能についてソフトウェアで実現したほうが良い場合、ハードウェアで実現したほうが良い場合を考える。
 - ・ハードウェア/ソフトウェアの分割に関しては、実現コストや性能要求、信頼性・安全性、再利用性・拡張性など複数の視点から検討し、その実現方法(ソフトウェアで実現/ハードウェアで実現)を考える
- ▶ 実現すべきシステムに対して、既存システムを構成する機能の一部再利用の可能性も検討する。
 - ・将来の製品のシリーズ化
 - ・システム全体を見通しよくする(構成、動作など)
 - ・デバッグ、保守しやすいシステム
 - ・高度な信頼性、安全性
- ▶ システムを構成する機能のうち、外部からコンポーネント・ブロックとして購入したり、導入したりする可能性についても検討しておく。
- ▶ 将来の製品シリーズ化を考え、機能ブロックレベルでの再利用を容易にする形で整理しておく。
- ▶ システム全体の見通しをよくするように図などを利用して表現しておく。
- ▶ システムの初期化を担う機能ブロックなども検討しておく。
- ▶ ハードウェアやソフトウェアの診断機能、デバッグやテストのための機能なども検討しておくことが望ましい。
- ▶ 高度な信頼性・安全性が求められる機能については多重化なども考慮する。
- ▶ また、利用するMPU、ROM/RAM容量、利用するLSI、関連する制御デバイスやセンサーなども明確にしておく。

2.1.3 システム全体の振舞いの設計

入力

製品企画書
製品仕様書
(SY106) システム要求仕様書
(SA104) 安全要求仕様書

概要

システムを構成する機能ブロックを念頭にシステム全体としての振舞いを検討し整理する。

出力

(SY203) システム動作設計書

■ 実施内容

- ① 個々の機能ブロックがどのように連携してシステムの機能サービスを実現していくかを整理する。
- ② 機能ブロックの並行動作の可能性も考慮して機能ブロックの動作タイミングなどを検討し、動作シナリオ(シーケンス)として整理する。また、割り込み動作などについても整理しておく。
- ③ システムを構成する機能ブロックの動作コンテキストを明確にする。
- ④ システム要求分析で検討したユースケースをもとにして、ユースケースと機能ブロックの対応付けをする。
- ⑤ システムの非機能要求を念頭に、システムの実作に関するレスポンスタイムなどの時間制約も検討し、ソフトウェア動作、ハードウェア動作の時間的な側面に関する概算見積りを行う。
- ⑥ システムのエラーハンドリングの考え方を整理しておく。
- ⑦ システムが使われる条件を考慮し、システムに求められる安全性を実現する仕組みを検討する。**安全**

■ 注意すべき事項

- ▶ システムで期待される正常な振舞いだけでなく、異常処理などの非正常な振舞いについても検討しておく。
- ▶ システムの内部状態などを考慮し、システムの状態遷移を明確にしておく(ハードウェア/ソフトウェアを含めたシステム総体としての状態遷移を考える)。

安全 : セーフティに関連する作業

2.1.4 インタフェースの設計

入力

(SY106) システム要求仕様書
 (SA104) 安全要求仕様書
 (SY201) システム構成図
 (機能ブロック図)
 (SY202) 共通機能・データ一覧表

概要

システムの外部および内部のインタフェースを明確にする。

出力

(SY204) システムインタフェース設計書

参照情報

各ハードウェア仕様書

■ 実施内容

- ①外部インタフェースを明確化する。
 - ▶ 開発対象のシステムとその外部にあるシステムとのインタフェースを明確にする。
 - ▶ 外部インタフェースを介してやり取りされるデータや情報を明確にするとともに、インタフェースを実現するための通信方式やデータ送受信方式などについても明確にしておく。
- ②内部インタフェースを明確にする。
 - ▶ システムを構成する機能ブロック間のインタフェースを明確にする。
 - ▶ システム内インタフェースを介してやり取りされるデータや情報を明らかにするとともに、システム内のユニット間のインタフェースとして、たとえばバスや共有メモリなどの方式を明確にしておく。
- ③ソフトウェア要素とハードウェア要素の間のインタフェースについても明確にする。
- ④システム外部とのインタフェースについては介在するセンサー、アクティエータなどの仕様や関係するデータにも着目し、必要に応じて関連データ、センサーデータ一覧表として整理する。

■ 注意すべき事項

- ▶ ソフトウェアとハードウェアの間のインタフェースについては、ハードウェアデバイスの仕様や特性を考慮し、必要に応じてハードウェアデバイスの仕様書との整合性を確認する。
- ▶ インタフェースとしてデータなどが介在する場合には、データの種類、大きさ(サイズ)、データ授受のタイミングやインターバルなどを明確にしておく。
- ▶ システム外部インタフェースの中にはユーザーインタフェース(音声、視覚等)なども含めて検討する。この場合、システム非機能要求の中のユーザビリティに関する要件も考慮する。
- ▶ ハードウェアとソフトウェアの間のインタフェースは漏れなく整理しておく必要がある。(例) 割り込みの定義
ソフトウェアから読み書きする入出力レジスタ

2.1.5 システム・アーキテクチャ設計書の作成


入力


(SY201) システム構成図
(機能ブロック図)
(SY202) 共通機能・データ一覧表
(SY203) システム動作設計書
(SY204) システムインタフェース設計書

概要

システム・アーキテクチャ設計の事項を整理し、システム・アーキテクチャ設計書としてまとめる。

出力

(SY205) システム・アーキテクチャ設計書 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

(SY201) ~ (SY204)の成果を取りまとめてシステム・アーキテクチャ設計書を作成する。

- ▶ システム・アーキテクチャ設計の過程で出力された資料を整理・体系化して文書化する。
- ▶ 内部確認・共同レビューなどでの指摘事項が適切に反映されていることも確認する。

■ 注意すべき事項

- ▶ システム・アーキテクチャ設計書としては、システム全体の静的構造と動的構造(振る舞い)を含めて開発対象のシステムの見通しをよくするように注意する。
 - ▶ システム・アーキテクチャの前提となっている条件についても明記しておく。また、なぜそのようなアーキテクチャにしたかについても理由が分かるようにしておくといよい。
 - ▶ 変更・追加内容が確認できるように版数管理および改版リストの添付が望ましい。
- ▶ 文書化での留意点
 - ・ 変更履歴を添付し、変更箇所も明示する
 - ・ 作成文書に関しての責任所在を明示する
 - ・ 作成文書は構成管理および変更管理にて確実に管理する

関連プロセス：SUP5 構成管理、SUP7 変更管理

■ SYP2.2 システム・アーキテクチャ設計の確認

設計されたシステム・アーキテクチャがシステム要求、安全性要求を満たしていることを確認する。

2.2.1 システム・アーキテクチャ設計書の内部確認

入力

(SY106) システム要求仕様書
(SA104) 安全要求仕様書
(SY205) システム・アーキテクチャ設計書
(SY201) システム構成図(機能ブロック図)
(SY203) システム動作設計書
(SY204) システムインタフェース設計書

概要

作成したシステム・アーキテクチャ設計書に記載された事項が、システム要求仕様を満たし、かつ適切であることを確認し、内部確認レポートとしてまとめる。

出力

(SY206) 内部確認レポート(システム・アーキテクチャ設計)

ドキュメント・テンプレート例が用意されています。

参照情報

各ハードウェア仕様書 他

■ 実施内容

- ①システム・アーキテクチャ設計書の内容が適切であるかどうかを確認する。
 - ▶ システムを構成する機能ブロックの分割が適切であり、システム要求で求められる事項が実現可能かどうか(トレーサビリティの確認)。
 - ▶ ソフトウェア/ハードウェアの分割が適切であるか。
 - ▶ 機能ブロック分割、ソフトウェア・ハードウェア分割の前提となる条件が正しいか。
 - ▶ システムに求められる非機能要求が実現可能であるか。
 - ・ システムの性能(効率性など)が期待通りの水準を実現できるか
 - ・ 将来のシステム拡張などを考慮して適切な保守性、移植性が実現できるか
- ②確認結果は内部確認レポート(SY206)として整理し、確認作業で指摘された問題およびその対応元を明記した上で、関係者に配布する。
 - ▶ 対象ユーザとその利用コンテキストを念頭に、利用性や安全性が考慮されているか。
 - ▶ システム全体としての動きが整理できているか。
 - ▶ システム要求やテスト仕様との対応(トレーサビリティ)が取れているか。
 - ▶ 最終的に整理されたシステム・アーキテクチャについて実現可能性を確認しておく。

■ 注意すべき事項

- ▶ システム・アーキテクチャ設計書の確認についてはレビュー会議などでの確認をすることも有効。レビュー会議の開催に関する留意事項は(SWP2.2) 参照のこと。
- ▶ システムの設計条件やアーキテクチャ検討の際の経緯や理由なども考慮して、確認を行う。
- ▶ 複数の担当者で分担してシステム・アーキテクチャの検討を進める場合には、担当者間の齟齬がないことも確認しておく。
- ▶ 管理者は内部確認において指摘された問題事項については、解決策やその対応のためのアクションなどもあわせて記載しておくことが望ましい。
- ▶ アーキテクチャ段階での問題は放置すると開発後半において大きな問題を生むため、この段階で内部確認レポートの情報は、プロジェクトマネージャ / 開発リーダー、製品企画担当者などステークホルダ間で情報共有し、コンセンサスを得ておくことが望ましい。

■ SYP2.3 システム・アーキテクチャ設計の共同レビュー

設計されたシステム・アーキテクチャが製品企画、システム要求を満たしていることをステークホルダ間で評価する。

2.3.1 システム・アーキテクチャ設計書の共同レビュー


入力


(SY106) システム要求仕様書
(SA104) 安全要求仕様書
(SY205) システム・アーキテクチャ設計書
(SY206) 内部確認レポート(システム・アーキテクチャ設計)

概要

システム・アーキテクチャ設計書の共同レビューを実施し、システム要求事項の実現方法の妥当性をステークホルダ*間で確認する。

出力

(SU801) 共同レビュー記録(システム・アーキテクチャ設計書)
(SY208) システム・アーキテクチャ設計共同レビュー報告書

ドキュメント・テンプレート例が用意されています。

■ 実施内容

- ①システム開発に関係するステークホルダ(企画部門担当者、ソフトウェア開発者、ハードウェア開発者、システム評価者、製造担当者など)を集めシステム・アーキテクチャ設計の妥当性を確認する。レビューでは特に下記の点を考慮する。
 - ▶ 検討したシステム・アーキテクチャ設計(How)はシステム要求(What)に合致しているか。
 - ▶ システム・アーキテクチャ設計の実現性が担保されているか。
 - ▶ ソフトウェア開発に対する要求事項が明確になっているか。
 - ▶ ハードウェア開発に対する要求事項が明確になっているか。
 - ▶ システムの結合や評価についても要求事項が明確になっているか。

- ▶ システムの脆弱性などセキュリティ面への配慮がされているか。
- ▶ システムの安全性の視点からの配慮がされているか。
- ▶ システム開発で利用する開発環境、システムの動作環境(OS、Lib)やシステムに組みこまれるミドルウェアなどが明確になっているか。
- ▶ 製造段階での動作評価のための方式が考慮されているか。
- ▶ システムが関係するドメインにおいて、準拠すべき法令、規則などが明らかになっており、それらに従っているか。
- ▶ システムの制約条件などが明確になっているか。

● 用語解説

* **ステークホルダ**: 企業内からエンドユーザまで製品に利害関係のある人のこと。

②レビューの記録をもとに、共同レビュー報告書を作成する。

- ▶ レビュー報告にはレビューの指摘事項やその対応についても明記し、関係するステークホルダに配布する。

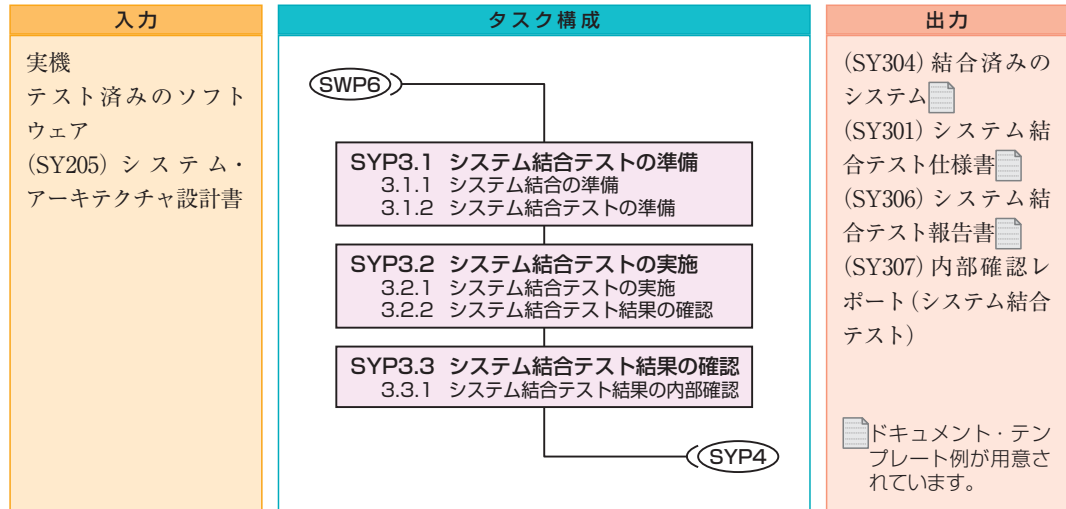
関連プロセス：SUP8 共同レビュー

■ 注意すべき事項

- ▶ ステークホルダ間の利害調整などを円滑にするために、議長、書記、レビューアなどの役割を明確にしたフォーマルレビューなどのスタイルの導入も効果的。
- ▶ 個々の機能ブロックなどに関する確認についても、複数のステークホルダの視点からレビューすることが望ましい。

SYP3 システム結合テスト

システムを構成するハードとソフトを組み合わせた際の機能ブロックが動作することを確認する。



■ 解説

このアクティビティは

- (3.1.1) ソフトウェア開発 (SWP) でソフトウェア総合テストを終えたソフトウェアと製品に用いられる実機 (ハードウェア) を結合して、最終的なシステムとして仕上げるための準備をし、
- (3.1.2) システム結合テストの準備をする。
- (3.2.1) 実機と実機に搭載可能な形式にしたソフトウェアを結合し、結合テストを行い、その結果を確認する。
- (3.2.2) (3.3.1) システム結合テスト結果はあらかじめ確認ポイント決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。

組込みシステムにおけるシステム結合テストは、ソフトウェアとハードウェアを結合して、システム要求、システム・アーキテクチャ設計通りに実現できていることを確認していく作業と位置づけることができる。

■ 留意事項

- ▶ 組込みシステムの結合はソフトウェアと実機 (ハードウェア) を結合していくため、どのような順序でどのように結合し、動作を確認していくかをあらかじめ検討しておく必要がある。
- ▶ ハードウェア担当者、ソフトウェア担当者双方の連携がスムーズに行くように結合スケジュールを調整しておく。

■【参考】手法およびツール

- ▶ バグ管理ツール
- ▶ 信頼度成長曲線

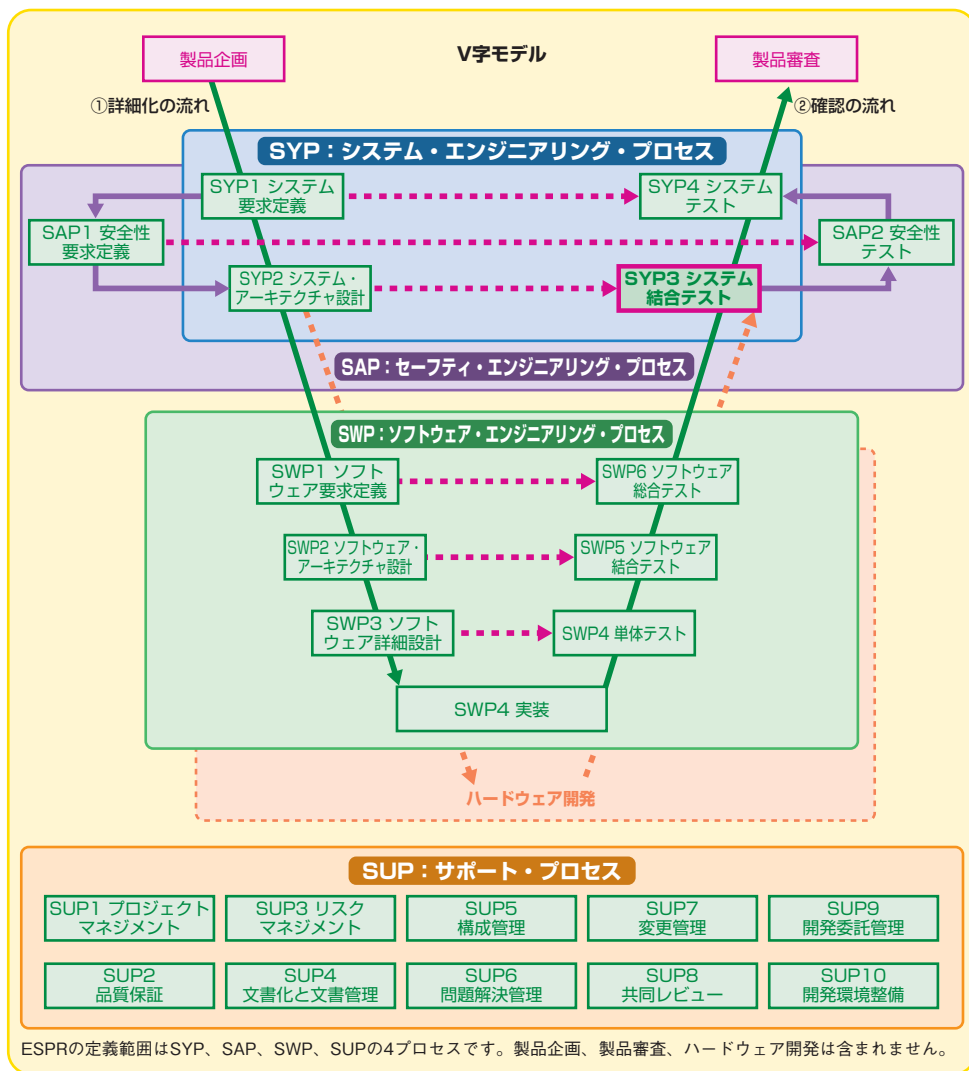


図2.4 V字モデルと開発プロセス (SYP3 システム結合テスト)

■ SYP3.1 システム結合テストの準備

システム結合テストの準備をする。

3.1.1 システム結合の準備

入力

実機
テスト済みのソフトウェア
(SY205) システム・アーキテクチャ設計書

概要

ソフトウェア開発 (SWP) でソフトウェア総合テストを終えたソフトウェアと製品に用いられる実機 (ハードウェア) を結合して、最終的なシステムとして仕上げるための準備をする。

出力

(ROMにインストールされたソフトウェアなど)

■ 実施内容

- ①システムを実現するために必要となる実機 (ハードウェア) を用意する。
- ②ソフトウェア総合テストを終了したソフトウェアを用意する。
- ③上記ソフトウェアを製品中のコンピュータで動作可能な状態にしておく。

■ 注意すべき事項

- ▶「ソフトウェアを製品中のコンピュータで動作可能な状態」とはターゲットマイコン向けにROMにインストールされた状態などを指す。
- ▶ソフトウェア、ハードウェアのバージョンなどに注意する。
- ▶システム結合については、どのような順序でどのように結合し、動作を確認していくかをあらかじめ検討しておく必要がある。
- ▶実機やICEの手配や設置などもあらかじめ準備しておく。
- ▶ハードウェア担当者、ソフトウェア担当者双方の連携がスムーズに行くように結合スケジュールの調整も重要になる。

3.1.2 システム結合テストの準備


入力


(SY205) システム・アーキテクチャ設計書
(SU601) 不具合管理票
(修正確認の場合)

概要

最終的にシステムとして結合する際の結合テストを行うための準備をする。

出力

(SY301) システム結合テスト仕様書
(SY302) システム結合テストデータ
(SY303) 内部確認メモ
(システム結合テスト仕様) 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

- ① システム結合の段階で確認すべき事項をリストアップし、結合テストのテスト項目を作成する。
- ② ソフトウェアとハードウェアを結合した場合のインタフェース部分についてもテスト項目を考える。
- ③ システムで実現するすべての機能について、結合したシステムにおいてそれらの機能の動作が実現できていることを確認するためのテスト項目を考える。
- ④ 上記のテスト項目を実施するためのテストデータを作成する。あわせてテスト結果の判定基準や、テスト全体の評価基準や完了基準なども用意しておく。
- ⑤ 上記のテスト項目、およびテストデータなどを整理し、システム結合テスト仕様書を作成する。
- ⑥ 修正確認テスト項目を準備する(修正確認の場合)。
 - ▶ システム結合テストにおいて不具合が検出されて修正を行った場合には、その不具合が解消されたこと、また修正により別の不具合が発生していないことを確認するためのテスト項目を準備する。
 - ▶ 不具合の内容により、テスト範囲を決定しテスト項目を選択する。

■ 注意すべき事項

- ▶ ソフトウェアとハードウェアのインタフェースとしては、特にハードウェアを介してソフトウェアに入出力されるデータに注意する。
- ▶ マルチCPU構成など複雑なシステムにおけるシステム結合では、システムを構成するすべてのハードウェアとソフトウェアを一度に結合するとは限らないため、システムの一部をテストとして動作させながら、システムで実現すべき機能を順次確認していくためのテスト項目を用意する。
- ▶ システム結合テスト仕様書では、テストの可否判定基準も検討しておく。
- ▶ 必要に応じて構成管理プロセスなども考慮する。システム結合および結合テストについては、結合の順序、テストの実施順序なども考えておく。
- ▶ 結合テストではシステムの一部を対象とする場合もあるため、模擬テスト装置や、テストプログラム、ICEなどを利用することも検討する。
- ▶ システム結合テスト仕様書は内容を確認しておく。
- ▶ 修正確認テストの準備の考慮点
 - ・テストの実施範囲は、影響する部分をすべて網羅しているか確認する
 - ・テストは基本的に既に作成したテスト項目を再利用する。ただし、修正量が多い場合や、影響範囲が広い場合などは、新たなテスト項目を用意することも検討する
- ▶ 文書化での留意点
 - ・変更履歴を添付し、変更箇所なども明示する
 - ・作成文書に関しての責任所在を明示する
 - ・作成文書は構成管理および変更管理にて確実に管理する

関連プロセス：SUP5 構成管理、SUP7 変更管理

SYP3.2 システム結合テストの実施

システム結合テストを実施する。

3.2.1 システム結合テストの実施

入力

テスト済みのソフトウェア
実機
(SY301) システム結合テスト仕様書
(SY302) システム結合テストデータ
(SU601) 不具合管理票
(修正確認の場合)

概要

実機と実機に搭載可能な形式にしたソフトウェアを結合し、結合テストを実施する。

出力

(SY304) 結合済みのシステム
(SY305) システム結合テスト結果

■ 実施内容

- ①製品に搭載可能な形式にしたソフトウェアを実機(ハードウェア)に組み込む。
- ②システム結合テスト仕様書に記載されたテスト項目に従い、順次、システム結合に関する動作確認を行う。
 - ▶ 代替項目やデータを用いてテストした場合にも、その理由を明記した上で記録に残す。
 - ▶ 準備したテスト項目が実施できなかった場合には、その理由を明記した上で記録に残す。また、その妥当性について判断する。
- ③修正確認テストを実施する。
 - ▶ 不具合が解消されたかテストを実施する。
 - ▶ 不具合の修正により、別の不具合が発生していないかテストを実施する。

■ 注意すべき事項

- ▶ テストにおいて検出された不具合を修正した場合には、その機能ブロックに関するブロックを洗い出し、その部分も含めて再度網羅的にテストする必要がある。
- ▶ 修正確認テストでは修正後の最新版でテストを実施しているか確認する。

3.2.2 システム結合テスト結果の確認



入力


(SY301) システム結合テスト仕様書
(SY305) システム結合テスト結果
(SU601) 不具合管理票
(修正確認の場合)

概要

システム結合テストの結果を確認し、合否判定をする。

出力

(SY306) システム結合テスト報告書 
(SU601) 不具合管理票 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

- ① システム結合テスト仕様書に記載した合否判定基準などを参考に、最終的なシステム結合の合否判定を行う。
- ② システム結合テストで検出された不具合については、ソフトウェア起因の不具合、ハードウェア起因の不具合など、不具合の原因箇所の特定を行い、不具合管理票に記載する。
- ③ テスト報告書には
 - ▶ テスト方法、テスト環境、利用したツール、データなど
 - ▶ 検出した不具合数や重大不具合、不具合内容の分析
 - ▶ テスト完了の判断と根拠などを明確に記述する。
関連プロセス：SUP6 問題解決管理

■ 注意すべき事項

- ▶ テストにおいて不具合が検出された場合には、まず、その不具合の再現を試み、その際のコンテキストやシステムの状況を明確にする。

SYP3.3 システム結合テスト結果の確認

結合された組み込みシステムがシステム・アーキテクチャ設計で定義された処理を正しく実現されているかテスト結果を確認する。

3.3.1 システム結合テスト結果の内部確認

入力

(SY205) システム・アーキテクチャ設計書
(SY301) システム結合テスト仕様書
(SY303) 内部確認メモ(システム結合テスト仕様)
(SY306) システム結合テスト報告書
(SU601) 不具合管理票

概要

システム結合テスト結果報告書の内容を確認し、システム結合段階での未解決な問題の有無や対応について確認する。

出力

(SY307) 内部確認レポート(システム結合テスト)



ドキュメント・テンプレート例が用意されています。

■ 実施内容

①下記の視点でシステム結合テスト結果を確認する。

- ▶ 未解決となっている問題が確認された場合には、
 - ・ その問題の重要度を評価し、
 - ・ システム全体としての機能への影響やシステムの信頼性・安全性に関わる重大な問題の場合には、
 - a. システム開発(ソフトウェア、ハードウェア)プロセスへの差し戻し
 - b. システム利用条件面などでの制約の付与

c. リリース計画の見直しなどを検討し、具体的な対策を実施する。

関連プロセス：SUP8 共同レビュー、SUP1 プロジェクトマネジメント

- ▶ 未実施となっているテスト項目がないか。未実施となっている場合は、未実施となった理由を確認して対応を検討する。

②確認結果は内部確認レポート(SY307)として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

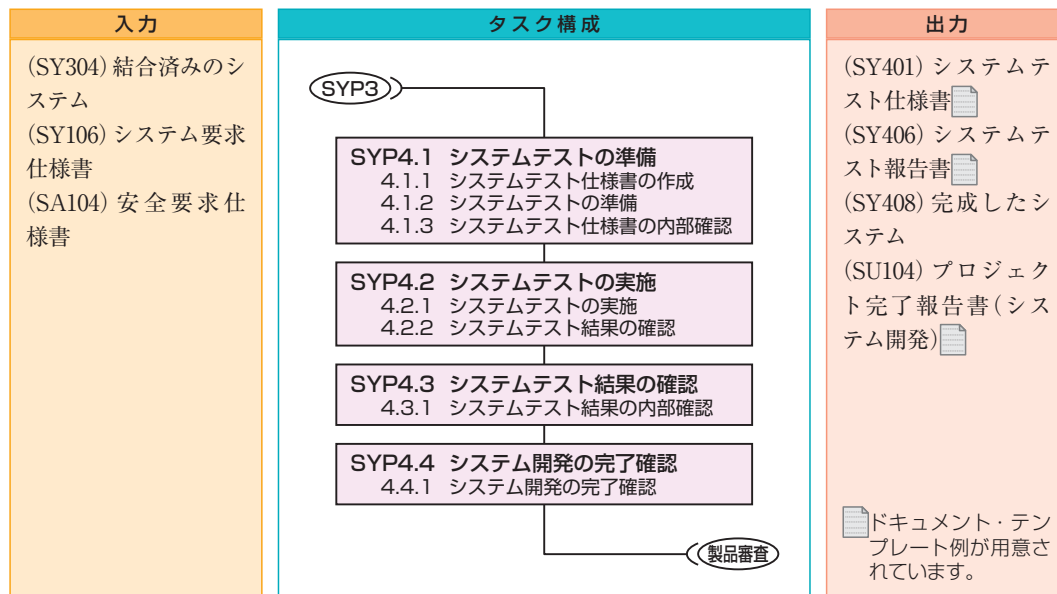
■ 注意すべき事項

- ▶ 不具合検出数は、品質基準を満たしているか確認する。
- ▶ 修正確認テストの場合は、テスト範囲と環境は適切であったか確認する。

- ▶ 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション(共同レビュー開催など)もあわせて記載しておくことが望ましい。

SYP4 システムテスト

システム要求が実現できていることを確認する。



■ 解説

このアクティビティは

- (4.1.1) (4.1.2) (4.1.3) 結合が完了したシステムが、システム要求仕様に記載された機能動作を実現しているかどうかを確認するためのテスト仕様を準備し、実際にそのテスト仕様に従い、システムを動作させ、システムとしての総合的なテストを行う。
- (4.2.1) (4.2.2) (4.3.1) システムテスト結果については、テストの合否判定基準に従って確認を行い、確認した内容を整理して内部確認レポートを作成する。
- (4.4.1) システムテストの結果報告書をもとに、製品システム部門としての最終的な合否を判定し、製品審査に必要な情報を整理する。

システムテストは、システムの開発において、最終的な確認の作業と位置づけることができる。このため、この作業は製品としての実フィールドの運用を想定して、漏れがないように実施されなければならない。

■ 留意事項

- ▶ システムテストでは、テストの際の動作環境(テスト環境)に注意する。実際にユーザが製品を利用する環境を想定したテストを実施する。
- ▶ システム開発における最終確認のための作業

として、製品の正常状態での利用条件のみだけでなく、正常な利用条件以外や、また、製品が関係する規格、規約、法律、法規の準拠など、テスト項目に漏れがないようにする。

■【参考】手法およびツール

- ▶ 自動テストツール (リグレッションテストテスト自動化など)
- ▶ バグ管理ツール
- ▶ 信頼度成長曲線

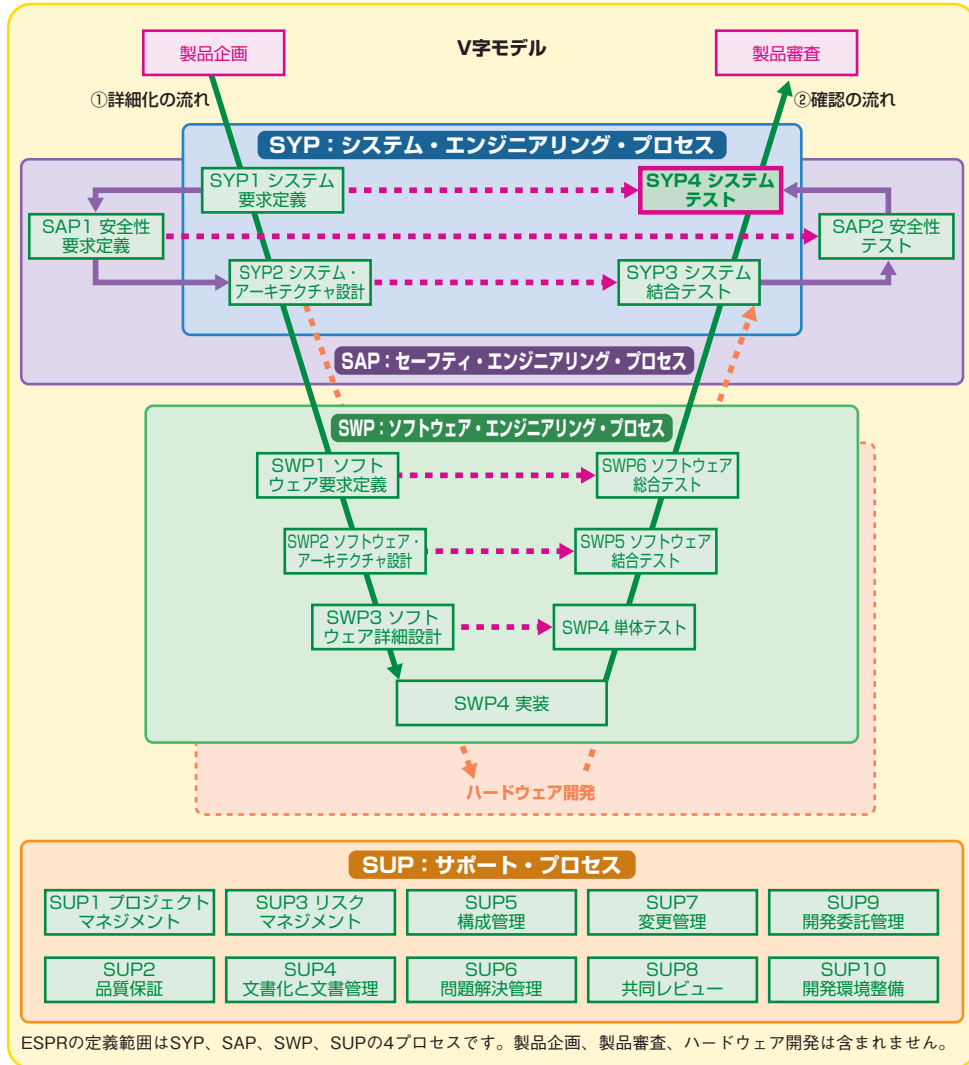


図2.5 V字モデルと開発プロセス (SYP4 システムテスト)

■ SYP4.1 システムテストの準備

システムテストの準備をする。

4.1.1 システムテスト仕様書の作成

入力


(SY106) システム要求仕様書
(SA104) 安全要求仕様書
製品マニュアル
テスト仕様書に必要な情報 (成果物)

概要

システム要求仕様書をもとに製品システムとして確認すべきことを洗い出しシステムテスト仕様書を作成する。

出力

(SY401) システムテスト仕様書 

ドキュメント・テンプレート例が用意されています。

参照情報

過去のシステムテスト仕様書

■ 実施内容

- ①システムテストでは、開発したシステムが実際の製品としてユーザ環境下で利用されることを想定して、システムテスト仕様書を作成し、さまざまな視点からのテストを行う。
システムテストの観点としては以下のようなものを考える。
 - ▶ マニュアルに記載されている機能、サービスに関する網羅的なテスト
 - ▶ システムの正常動作、非正常動作に関するテスト
 - ▶ システムの連続使用などを考えた耐久性テスト
 - ▶ システムの処理性能などを含むパフォーマンスに関するテスト
 - ▶ システムの操作性に関するテスト
 - ▶ システムの保守性 (オンラインアップデートなど) に関するテスト
 - ▶ システムのセキュリティ面に関するテスト
 - ▶ システムの安全性に関するテスト
関連プロセス: SAP2 安全性テスト
 - ▶ 製品が関係する規格、規約、法律、法規の準拠に関するテスト
- ②システムテスト仕様書では、テスト項目、前提となるシステムの動作条件 (状態)、期待されるシステムの動作などを上記の視点を参考に個々の項目単位で整理しておく。
 - ③システムテストの合否判定の基準を明確にしておく。

■ 注意すべき事項

- ▶ システムの動作条件(状態)については、製品の正常状態での利用条件のみだけではなく、正常な利用条件以外のことも考慮する。
- ▶ 過去に類似のプロダクトがある場合は、過去の不具合も考慮に入れる。
- ▶ システムに関係するデータについても漏れなく確認する。
- ▶ 開発サイドで行うテストとユーザサイドで行うテストは区別しておく。**安全**
- ▶ テスト仕様書は、設計フェーズなど上流プロセスの中で検討、作成しておくが良い。
- ▶ 文書化での留意点

- ・ 変更履歴を添付し、変更箇所なども明示する
- ・ 作成文書に関しての責任所在を明示する
- ・ 作成文書は構成管理および変更管理にて確実に管理する

関連プロセス：SUP5 構成管理、SUP7 変更管理

安全：セーフティに関連する作業

4.1.2 システムテストの準備

入力

(SY401) システムテスト仕様書
テスト対象のシステム
(SU601) 不具合管理票
(修正確認テストの場合)

概要

システムテスト仕様書をもとにシステムテストの準備(テストデータ、テスト環境など)をする。

出力

(SY402) システムテストデータ
(SY403) システムテスト環境

■ 実施内容

①テスト環境を準備する。

- ▶ システムテストでは実際の動作環境上(実機環境)でシステムを動作させることが基本となるため、実際にシステムが利用され動作する環境を用意しておく。

②テストデータを用意する。

- ▶ システムテストのテスト項目を実行するのに必要となるテストデータ(システムへの入力データやユーザ操作など)を用意する。
- ▶ システムテストで行うテストの種別なども検討しておく。
- ▶ テスト項目によっては、システムの動作環境(条件)や必要なテストデータなどが極めて稀なケースも含まれるため、必要な環境やデー

タの準備方法なども検討しておく。

③テスト結果の判定基準や、テスト全体の評価基準や完了基準なども用意しておく。

④修正確認テスト項目を準備する(修正確認の場合)。

- ▶ システム結合テストにおいて不具合が検出されて修正を行った場合には、その不具合が解消されたこと、また修正により別の不具合が発生していないことを確認するためのテスト項目を準備する。
- ▶ 不具合の内容により、テスト範囲を決定しテスト項目を選択する。

■ 注意すべき事項

- ▶ システムによっては開発サイドでシステムの実際の動作環境のすべてを準備しきれない場合もあるため、早めにテスト環境の準備や手配をしておく。
またどうしても準備しきれない場合には、必要に応じてシミュレータなど動作環境の代替手段も検討することが望ましい。
- ▶ システムテストはさまざまな観点からのテストが行われるため、複数人でテスト項目を分担して行う場合がある。このためテスト環境やテスト対象システムはテスト担当者の人数分を用意しておく必要がある。

- ▶ システムテストでは動作環境やシステムの特性によって、微妙なタイミングで動作が決定する場合があります。動作の再現が難しい場合があります。このためテストの動作再現の方法などについても工夫しておく。
- ▶ 修正確認テストの準備の考慮点
 - ・ テストの実施範囲は、影響する部分をすべて網羅しているか
 - ・ テストは基本的に既に作成したテスト項目を再利用する。ただし、修正量が多い場合や、影響範囲が広い場合などは、新たなテスト項目を用意することも検討する

4.1.3 システムテスト仕様書の内部確認

入力

(SY106) システム要求仕様書
(SA104) 安全要求仕様書
(SY401) システムテスト仕様書
テスト対象のシステム

概要

システムテスト仕様書を確認する。

出力

(SY404) 内部確認メモ
(システムテスト仕様)

■ 実施内容

作成したシステムテスト仕様書を確認する。

- ▶ システムテスト仕様書 (SY401) に記載されたテスト項目が、システム要求仕様書 (SY106) に定義された機能要求、非機能要求、動作制約を検証する項目となっているかを確認する。

■ 注意すべき事項

- ▶ (SYP4.1.1) に示したシステムテストで確認すべき視点について、どの程度カバーしているかを確認する。
- ▶ 製品マニュアル、取扱説明書などとの対応関係が取れていること、網羅していることを確認する。

■ SYP4.2 システムテストの実施

システムテストを実施する。

4.2.1 システムテストの実施

入力

(SY401) システムテスト仕様書
テスト対象のシステム
(SY402) システムテストデータ
(SY403) システムテスト環境

概要

システムテスト仕様書に従ってシステムテストを実施する。

出力

(SY405) システムテスト結果

■ 実施内容

システムテスト仕様書に従いテストを実施する。

①システムテストを実施する。

- ▶ システムテスト仕様書(SY401)をもとに、システムテストを実施し、出力結果を得る。
- ▶ 代替テストを実施した場合は、理由を明記した上で記録を残す。
- ▶ 不具合検出時、継続してテストを実施して残りのテストを実施するか、不具合が修正されるまでペンディングするかの判断をする。
- ▶ 出力結果を収集する(各種ログなど)。
- ▶ 準備したテスト項目が実施できなかった場合は、理由を明記した上で記録を残す。また、理由の妥当性について判断する。

②修正確認テストを実施する。

- ▶ 不具合が解消されたかテストを実施する。
- ▶ 不具合の修正により、別の不具合が発生していないかテストを実施する。

■ 注意すべき事項

- ▶ テストの再現性を考え、再テストに必要なデータ等はすべて保存しておくことが望ましい。
- ▶ 修正確認テストでは修正後の最新版でテストを実施しているか確認する。
- ▶ システムに修正、変更の必要が生じた場合には、影響解析などを行った上でシステムに手を加える。

4.2.2 システムテスト結果の確認



入力


(SY401) システムテスト仕様書
テスト対象のシステム
(SY405) システムテスト結果
(SU601) 不具合管理票
(修正確認テストの場合)

概要

テスト結果を確認し、テストの合否を判定する。

出力

(SY406) システムテスト報告書 
(SU601) 不具合管理票 

 ドキュメント・テンプレート例が用意されています。

■ 実施内容

システムテスト結果を確認し、実施したテストの各項目の合否を判定する。

- ▶ 不具合を発見した場合、不具合管理票に不具合内容を記録する。
- ▶ 修正確認テストの結果、不具合が解消された場合は、修正確認が済んだことを不具合管理票に記録する。

関連プロセス：SUP6 問題解決管理

■ 注意すべき事項

- ▶ テスト結果に関しては、システムテスト仕様書の誤りの可能性についても考慮する。
- ▶ 結果の合否判定については、システムテスト仕様書に記載された判定基準に基づいて行う。

SYP4.3 システムテスト結果の確認

システム要求定義で定義された要求事項を正しく実現されているかテスト結果を確認する。

4.3.1 システムテスト結果の内部確認


入力


(SY106) システム要求仕様書
(SA104) 安全要求仕様書
(SY401) システムテスト仕様書
(SY404) 内部確認メモ (システムテスト仕様)
(SY406) システムテスト報告書
(SU601) 不具合管理票

概要

システムテスト結果報告書の内容を確認し、システム開発の最終段階での未解決な問題の有無や対応について確認し、内部確認レポートとしてまとめる。

出力

(SY407) 内部確認レポート (システムテスト) 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

① 下記の視点でシステムテスト結果を確認する。

- ▶ 未解決となっている問題が確認された場合には、
 - ・ その問題の重要度を評価し、
 - ・ システム全体としての機能への影響やシステムの信頼性・安全性に関わる重大な問題の場合には、
 - a. システム開発 (ソフトウェア、ハードウェア) プロセスへの差し戻し
 - b. システム利用条件面などでの制約の付与
 - c. リリース計画の見直しなどを検討し、具体的な対策を実施する。

関連プロセス：SUP8 共同レビュー、SUP1 プロジェクトマネジメント

- ▶ 未実施となっているテスト項目が確認された場合は、未実施となった理由を確認して対応を検討する。

関連プロセス：SUP6 問題解決管理

② 確認結果は内部確認レポート (SY407) として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

■ 注意すべき事項

- ▶ 不具合検出率は、品質基準を満たしているかを確認する。
- ▶ 修正確認テストの場合は、テスト範囲と環境は適切であったかを確認する。
- ▶ 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション (共同レビュー開催など) などともあわせて記載しておくことが望ましい。

SYP4.4 システム開発の完了確認

システム要求定義で定義された要求事項を正しく実現していることをステークホルダ間で評価する。

4.4.1 システム開発の完了確認

入力

(SY106) システム要求仕様書
(SY401) システムテスト仕様書
(SY406) システムテスト報告書
(SY407) 内部確認レポート(システムテスト)
(SA104) 安全要求仕様書
(SA201) 安全性テスト仕様書
(SA206) 内部確認レポート(安全性テスト)
(SU101) プロジェクト計画書
(SU601) 不具合管理票

概要

システムテストの結果報告書をもとに共同レビューを実施し、製品システム部門としての最終的な合否の判定をするとともに、製品審査に必要な情報を整理する。

出力

(SU801) 共同レビュー記録(システムテスト)
(SU104) プロジェクト完了報告書(システム開発)
(SY408) 完成したシステム

ドキュメント・テンプレート例が用意されています。

■ 実施内容

①システム開発に関するステークホルダ(企画部門担当者、ソフトウェア開発者、ハードウェア開発者、システム評価者、製造担当者など)を集めシステムの最終的な合否判定をする。

レビューでは特に下記の点を考慮する。

- ▶製品システムとして想定される利用者、利用環境下で、
 - ・期待される機能要求が確実に保証されていることを確認する
 - ・求められる非機能要求が確実に保証されていることを確認する
- ▶システム開発に関しての残件が無いことを確認する。

- ▶システムテストにおける不具合検出数、修正数、重大不具合の検出数、修正数などの指標を用いて品質面での問題が無いことを確認する。
- ▶システムテスト以前に実施されたレビューなどにおけるすべての指摘事項について、適切な対応がとられていることを確認する。

関連プロセス：SUP2 品質保証、SUP6 問題解決管理、SUP8 共同レビュー

②レビューの記録をもとに、完了報告書を作成する。

完了報告書の作成については「SUP1.4 プロジェクト完了報告書の作成」を参照のこと。

完了報告書は、システム開発に関するステークホルダに配布する。

関連プロセス：SUP1 プロジェクトマネジメント

■ 注意すべき事項

- ▶ レビューでは開発について、定量的側面、定性的側面の両面から開発過程の妥当性とその結果として作成されたプロダクトとしてのシステムの妥当性を確認する。
- ▶ システム開発における最終段階でのレビューとなるため、その開発の責任を負う担当者、管理者がレビュー結果の判定を行う。

SWP：ソフトウェア・エンジニアリング・プロセス

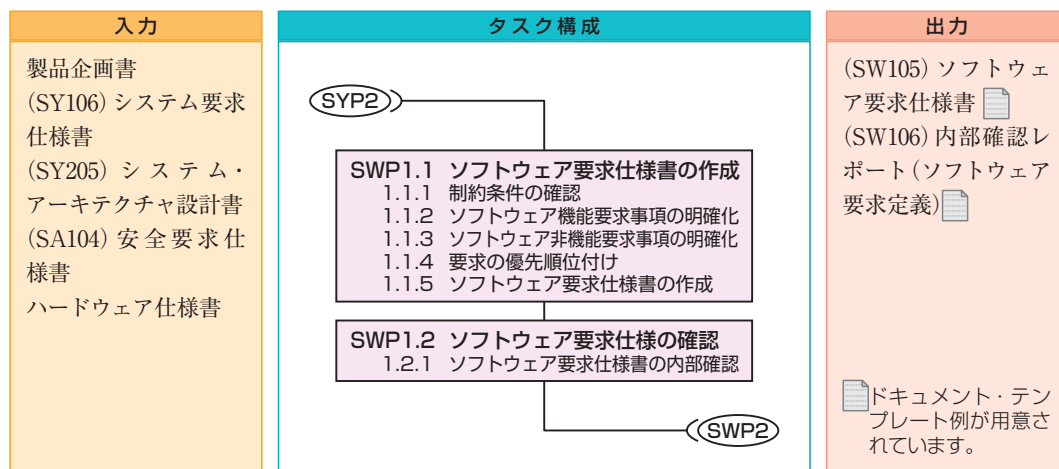
ソフトウェア・エンジニアリング・プロセスは、組込みソフトウェア開発に関する作業の中でも、直接ものづくりに関係するソフトウェアとしての要求定義からソフトウェア総合テストまでの作業を定義しています。

このプロセスには、下記のアクティビティが含まれています。

| ID | アクティビティ | アクティビティの概要 | 構成するタスク |
|------|------------------|---|--------------------------------|
| SWP1 | ソフトウェア要求定義 | 当該製品を実現するためにソフトウェアとして実現が必要となる要求を明確にする。 | SWP1.1 ソフトウェア要求仕様書の作成 |
| | | | SWP1.2 ソフトウェア要求仕様の確認 |
| SWP2 | ソフトウェア・アーキテクチャ設計 | 開発する組込みソフトウェアのアーキテクチャ(=動作[振る舞い]と構造)を決定する。 | SWP2.1 ソフトウェア・アーキテクチャ設計書の作成 |
| | | | SWP2.2 ソフトウェア・アーキテクチャ設計の確認 |
| | | | SWP2.3 ソフトウェア・アーキテクチャ設計の共同レビュー |
| SWP3 | ソフトウェア詳細設計 | ソフトウェア・アーキテクチャ設計で定義された機能ユニットをプログラムユニットに分割し、詳細な振る舞いや論理構造などを設計する。 | SWP3.1 機能ユニット詳細設計書の作成 |
| | | | SWP3.2 ソフトウェア詳細設計の確認 |
| | | | SWP3.3 ハードウェア仕様との整合性の確認 |
| SWP4 | 実装および単体テスト | ソフトウェアを構成する個々のユニットの実装と単体レベルでの動作確認を行う。 | SWP4.1 実装および単体テストの準備 |
| | | | SWP4.2 実装および単体テストの実施 |
| | | | SWP4.3 実装および単体テスト結果の確認 |
| SWP5 | ソフトウェア結合テスト | ソフトウェアを構成する個々のプログラムユニットを順次組み立て、それぞれ組み合わせた際の機能が動作するかどうかを確認する。 | SWP5.1 ソフトウェア結合テストの準備 |
| | | | SWP5.2 ソフトウェア結合テストの実施 |
| | | | SWP5.3 ソフトウェア結合テスト結果の確認 |
| SWP6 | ソフトウェア総合テスト | ソフトウェアを構成する個々の要素(機能ユニット)をすべて結合した状態で、ソフトウェアとしての総合的なテストを実施する。 | SWP6.1 ソフトウェア総合テストの準備 |
| | | | SWP6.2 ソフトウェア総合テストの実施 |
| | | | SWP6.3 ソフトウェア総合テスト結果の確認 |
| | | | SWP6.4 ソフトウェア開発の完了確認 |

SWP1 ソフトウェア要求定義

当該製品を実現するためにソフトウェアとして実現が必要となる要求を明確にする。



■ 解説

このアクティビティは

- (1.1.1) システム要求仕様とハードウェア仕様をもとに、ソフトウェア要求仕様を検討する際の制約条件を確認した上で、
- (1.1.2) ソフトウェアに必要な機能面の要求を明確にし、
- (1.1.3) また、ソフトウェアに必要な非機能面の要求も合わせて明確にする。
- (1.1.4) 実際のソフトウェア開発では開発期間やリソース面の制約、あるいは、(1.1.1) で整理したシステムとしての制約などがあるため、これらを考慮したうえで、個々の機能、非機能要求に優先順位をつけていく。
- (1.1.5) これらの検討結果を整理し、ソフトウェア要求仕様書を作成する。
- (1.2.1) 作成したソフトウェア要求仕様書はあらかじめ確認ポイントを決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。

■ 留意事項

- ▶ ソフトウェア要求定義の開始条件として以下の事項に留意する。
 - ・ 製品企画：製品戦略（エンドユーザーニーズなど）が明確になっている
 - ・ 製品仕様：取扱説明書レベルの内容は決まっている
 - ・ スケジュール：全体スケジュールは決まっている（発売日、対外的なマイルストーンなど）
- ・ システム・アーキテクチャ：以下の事項が明確になっている
 - ハードとソフトの機能分担（ソフトウェア要求は明確化されている）、ハードウェア構成、外部インタフェース、要求性能の実現方法、保守機能、セキュリティ

- ・ 前提条件：使用するソフトウェア (OS、ライブラリなど)、既存製品の利用
- ▶ ソフトウェア要求定義にあたっては、システム要求定義で検討した以下の事項に留意する。
 - ・ 組込みシステムの場合、システムが動作する環境など外部環境の分析とそれに伴う異常処理対応の機能分析を考慮する
 - ・ 機能的な側面のみでなく性能や保守性など非機能的な側面を考慮する
 - ・ 対象システムの外部で連携動作するシステムにも留意する
 - ・ 製品として見た場合のロングレンジの製品機能のあり方も考慮する

■【参考】手法およびツール

- ▶ OOA (Object Oriented Analysis：オブジェクト指向分析)
- ▶ 構造化分析
- ▶ DFD (Data Flow Diagram：構造化分析で使われる分析技法)
- ▶ シナリオ分析
- ▶ プロトタイピング
- ▶ 品質機能展開

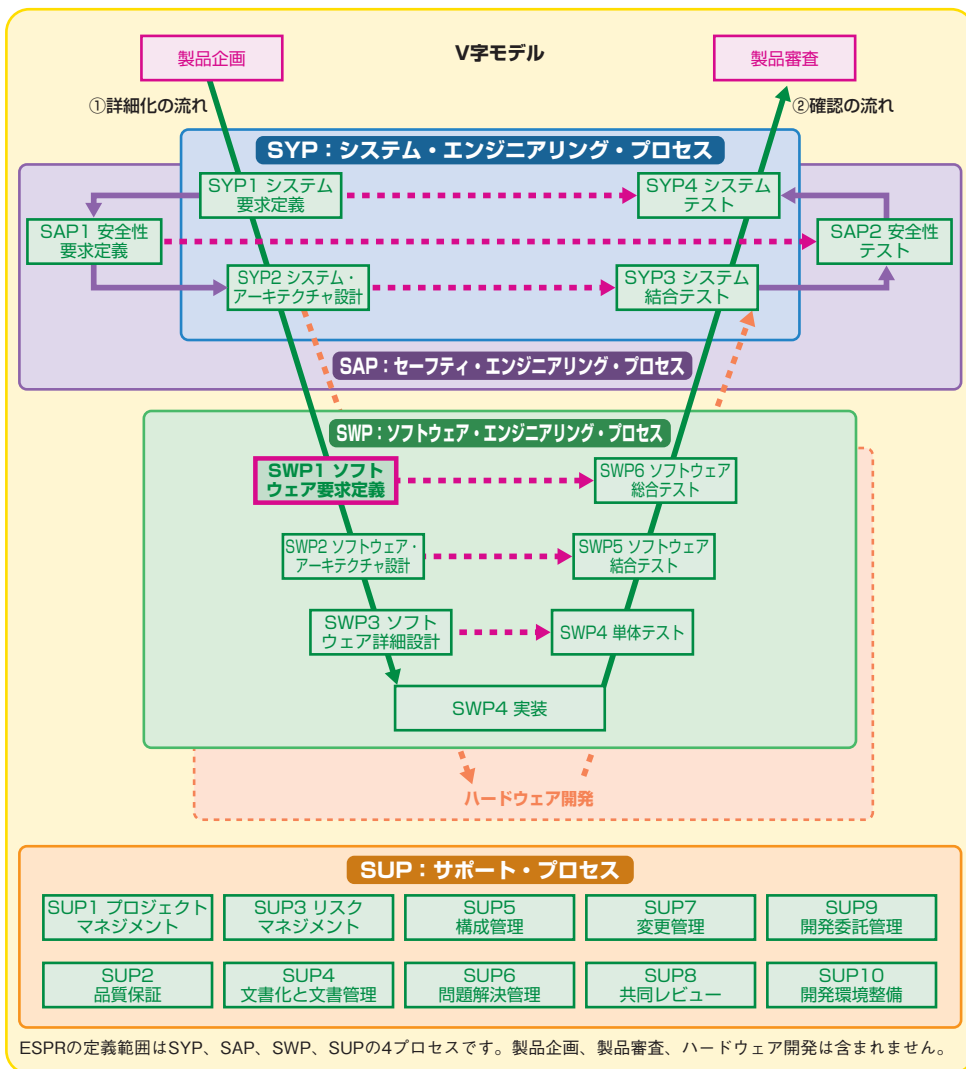


図2.6 V字モデルと開発プロセス (SWP1 ソフトウェア要求定義)

SWP1.1 ソフトウェア要求仕様書の作成

システム要求仕様書をもとにソフトウェアとして実現が求められる事項を明確にし、ソフトウェア要求仕様書としてまとめる。

1.1.1 制約条件の確認

入力

製品企画書
(SY106) システム要求仕様書
(SY205) システム・アーキテクチャ設計書
(SA104) 安全要求仕様書
各ハードウェア仕様書

概要

ソフトウェアに関する要求事項を検討するにあたっての考慮すべき制約条件を明確にし、制約条件リストとしてまとめる。

出力

(SW101) 制約条件リスト

参照情報

- ・ ISO/IEC 9126-1* (ソフトウェア品質モデル)
- ・ ISO 9241* (ユーザビリティ)
- ・ ISO 13407* (人間中心設計プロセス)
- ・ IEC 61508** (安全性)

■ 実施内容

以下に示す①～⑥の事項を確認し、制約条件リストとして整理する。

- ①製品企画、製品開発戦略を確認する。ソフトウェア要求を定義するにあたって、考慮すべき製品目標を確認する。
 - ▶ 特徴的な新機能の有無
 - ▶ プロダクトライン開発の適用 など
- ②製品特性を確認する。
 - ▶ 信頼性要求、安全性要求 **安全**
 - ▶ 耐用年数、製品寿命
- ③製品のステークホルダ*** (利害関係者)を確認する。
 - ▶ 想定される利用状況、利用環境
 - ▶ 準拠しなければならない規格・規約の有無など
- ④製品のステークホルダ*** (利害関係者)を確認する。
 - ▶ サービス部門、営業、企画、ハード開発部門、製造部門など、製品に関わるステークホルダを確認する。

● 用語解説

* ISO/IEC 9126-1 (ソフトウェア品質モデル)、ISO 9241 (ユーザビリティ)、ISO 13407 (人間中心設計プロセス) : ソフトウェアの取り扱いの容易さや操作の分かりやすさなどの使い勝手 (ユーザビリティ) に関連する規格。

** IEC 61508 (安全性) : 計算機などを内包し高い安全性を求められるシステム (プラント制御や自動車など) について、その開発過程で安全性実現の視点から実施すべき作業を整理した規格。

*** ステークホルダ : 企業内からエンドユーザまで、製品に利害関係のある人のこと。

安全 : セーフティに関連する作業

- ▶ 製品のエンドユーザを明確にし、ユーザグループ別の特徴を確認する。
 - ▶ ステークホルダ別に対応しなければならない制約条件をリストアップする。
- ④製品構成を確認する。
- ▶ ハードウェア構成とその制約。
 - ▶ 利用するOS、ミドルウェアなどを明確にし、それぞれの制約をリストアップする。
 - ▶ 製品が連携動作する周辺のソフトウェア、システムやハードウェアなどとのインタフェースを明確にする(センサー、アクチュエータなど)。
- ⑤再利用ソフトウェアを確認する。
- ▶ 既存ソフトウェアを再利用するか否かを検討する。
 - ▶ 再利用する場合は、再利用ソフトウェアの仕様や特徴、ならびに再利用の方針を確認する。
- ⑥ソフトウェアの開発環境、テスト環境、導入環境を確認する。
- ▶ 開発に利用するツール
 - ▶ テスト環境、テスト用ツール、テスト方法、テストデータの利用可能性
 - ▶ インストール時の制約なども明らかにしておく。

■ 注意すべき事項

● 製品企画、製品開発戦略

- ▶ エンドユーザニーズ、製品バリエーションや中長期レンジのマーケット戦略などを確認する。
- ▶ 全体スケジュール(発売日、対外的なマイルストーンなど)を確認し、ソフトウェア開発に充当できる期間を明確にしておく。
- ▶ 競合製品に対して優位性を確保するための機能は何かについても確認する。
- ▶ ハードウェアプラットフォームやミドルウェア、周辺デバイスの技術進化とその時期などの製品戦略を確認する。
- ▶ 製品仕様(取扱説明書レベルの内容は決まっているか)を確認する。
- ▶ 達成すべき性能、機能について、TBD (To Be Determined : 未定) 事項などを明確にしておく。

● 製品特性

- ▶ 信頼性：どの程度の信頼性が求められるか MTBF (Mean Time Between Failure : 平均故障間隔) や MTTR (Mean Time To Repair : 平均復旧時間) などの具体的な指標値を検討しておく。
- ▶ 動作環境：製品が利用されるコンテキストを明確にしておく(例：温度、ノイズ、静電気)。組込みシステムでは正常系のみでなく異常系の動作が発生しうるコンテキストについての

調査も必須となる。

- ▶ 保守環境：保守のタイミングや方法についても検討しておく。
- ▶ 準拠規約：PL法、環境基準
- ▶ 安全性：システムに求められる安全に関する要求(安全度のレベルとそれを実現する機能など) **安全**

● ステークホルダ

- ▶ ユーザについては、製品の一次ユーザのみだけでなく、その先の二次ユーザなども確認する。

● 製品構成

- ▶ 使用するMPU/MCUの種類
- ▶ 利用可能なメモリ容量
- ▶ 入出力機器
- ▶ OS、ライブラリ
- ▶ ハードウェアとソフトウェアの機能分担

● 再利用ソフトウェア

- ▶ ソフトウェアの再利用については、再利用の粒度に留意する(アーキテクチャの再利用なども視野に入れて考える)。

安全 : セーフティに関連する作業

1.1.2 ソフトウェア機能要求事項の明確化

入力

製品企画書
(SY106) システム要求仕様書
(SY205) システム・アーキテクチャ設計書
(SA104) 安全要求仕様書
各ハードウェア仕様書

概要

開発対象ソフトウェアに求められる機能要求*
事項を検討し、ソフトウェア機能要求リストと
してまとめる。

出力

(SW102) ソフトウェア機能要求リスト

参照情報

ISO/IEC 9126-1 (ソフトウェア品質モデル)
ISO 9241 (ユーザビリティ)
ISO 13407 (人間中心設計プロセス)
IEC 61508 (安全性)

■ 実施内容

システムとして実現・提供する機能のうち、ソフトウェアで実現する機能を明確にし、ソフトウェア機能要求リストを作成する。

機能要求の洗い出しについては、ソフトウェアとしてのユースケース分析の結果も考慮して検討していく。

■ 注意すべき事項

- ソフトウェア機能要求の検討の考慮点
 - ▶ システム要求、システム・アーキテクチャ設計を参考に、ソフトウェアとして実現する機能を明確に切り出しておく。
 - ▶ ソフトウェア機能に関連するハードウェア機能、プラットフォームなども明確にしておく。
 - ▶ 競合製品に対して優位性を確保するために必要な機能は何かを検討する。
 - ▶ ソフトウェアでは性能が出ない可能性が高い場合は、一部の機能をハードウェアで実現する。
- ▶ ユースケース分析ではユースケース・シナリオ、ユースケース図、アクティビティ図などを作成する。
- ▶ 要求漏れがないことを確認する。
- ▶ 異常時を十分考慮していることを確認する。
- 安全性に関して要求されるソフトウェア機能の例 **安全**
 - ▶ フェールセーフのための機構
 - ▶ 故障時におけるデータ保護

用語解説

* 機能要求：「xxx というデータを入力できること」「メールを送信できること」といった、製品や利用者のニーズをもとにソフトウェアに求められる機能のこと。

安全：セーフティに関連する作業

Related Standards

ISO/IEC 9126-1 Product Quality – Quality Model JIS X0129-1 ソフトウェア製品の品質 – 品質モデル

成立・改訂年：ISO/IEC 2001、JIS 2003

目的： ソフトウェア製品の品質指標

対象分野： ソフトウェア製品

URL： <http://www.iso.org/>

国内入手先： 日本規格協会

概要：

ソフトウェアの品質を示す特性(ソフトウェア品質特性)を定めた規格です。ソフトウェア品質特性では、ソフトウェアの特性を6つ(機能性、信頼性、使用性、効率性、保守性、移植性)に分類し、それぞれに副特性を定義しています。

ソフトウェア品質特性

| 特性 | 副特性 |
|-----|--------------------------------|
| 機能性 | 合目的性、正確性、相互運用性、セキュリティ、機能性標準適合性 |
| 信頼性 | 成熟性、障害許容性、回復性、信頼性標準適合性 |
| 使用性 | 理解性、習得性、運用性、魅力性、使用性標準適合性 |
| 効率性 | 時間効率性、資源効率性、効率性標準適合性 |
| 保守性 | 解析性、変更性、安定性、試験性、保守性標準適合性 |
| 移植性 | 環境適応性、設置性、共存性、置換性、移植性標準適合性 |

1.1.3 ソフトウェア非機能要求事項の明確化

入力

製品企画書
(SY106) システム要求仕様書
(SY205) システム・アーキテクチャ設計書
(SA104) 安全要求仕様書
各ハードウェア仕様書

概要

開発対象ソフトウェアに求められる非機能要求*事項を検討し、ソフトウェア非機能要求リストとしてまとめる。

出力

(SW103) ソフトウェア非機能要求リスト

参照情報

ISO/IEC 9126-1 (ソフトウェア品質モデル)
ISO 9241 (ユーザビリティ)
ISO 13407 (人間中心設計プロセス)

■ 実施内容

システムとしての機能の実現において関係すると考えられるソフトウェアの非機能的な側面の要求を明確にし、ソフトウェア非機能要求リストにまとめる。

- ▶ 信頼性要求
- ▶ 使用性要求
- ▶ 効率性要求
- ▶ 保守性要求
- ▶ 移植性要求
- ▶ その他のソフトウェア非機能要求

■ 注意すべき事項

● 信頼性要求の検討例

- ▶ 特定のシステム動作条件において、ハードウェアあるいはソフトウェアの想定外の動作が必要になる状況などを検討し、システムの異常処理方式を決めておく。
- ▶ システムにとって望ましくない状況が発生した際にも、システムとしての最低限の機能は動作するよう、ソフトウェア面での工夫も検討しておく。
- ▶ システムの異常動作モードからの復帰手順や復帰方式を明確にしておく。

● 使用性要求の検討例

- 組込みシステムの多くは不特定多数のエンドユーザを対象とする場合が多いことを念頭にソフトウェアで実現する部分の操作性などを検討する(システム全体としてのユーザインタフェースの統一感など)。
- また、ハードウェアで実現する部分との接点(たとえば、画面表示や演算に要する時間など)も、使用性の視点から検討することが望ましい。

● 用語解説

- * **非機能要求**:「nn 秒以内に処理を終えること」「利用者がマニュアルを見ないで操作できること」「再利用できること」など、ソフトウェアに求められる効率性や使用性、移植性などのこと。

● 効率性要求の検討例

- ▶ システムの実行性能 (例：処理速度、起動時間、応答時間) について考慮する。特にシステムのハードウェア制約や外部動作環境からくる時間制約に注意する。
- ▶ システムのハードウェアに起因するリソース効率 (例：メモリ容量、データサイズ) に注意する。システムで扱うデータの生存期間なども考慮に入れる。

● 保守性要求の検討例

フィールドトラブルが発生した場合のトラブル原因の解析を可能とするための機構の検討や動作ログ情報の記録メカニズムをソフトウェア面でも検討する。リモートメンテナンスなど保守の方式とその実現方法も検討しておく。

● 移植性要求の検討例

OSやCPU、周辺回路などの変更に伴うソフトウェアの移植しやすさなども考慮する。また既存ソフトウェアの一部を利用する場合も想定し、ソフトウェアユニットの独立性をあらかじめ考慮しておく。

● その他の非機能要求の検討例

- ▶ 再利用のために求められる機構やアーキテクチャの確認
- ▶ セキュリティ要求 (例：データ暗号化、ユーザー認証、ウイルス対策)
- ▶ 相互運用性 (例：通信プロトコル)
- ▶ 外部インタフェース要求 (例：連携ソフトウェアとの関数インタフェース、通信プロトコル、ユーザインタフェース)
- ▶ データ定義

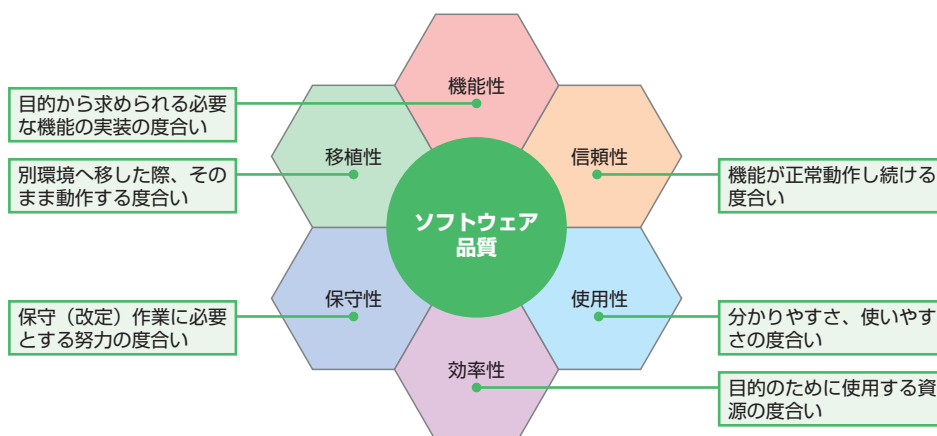


図2.7 ISO/IEC9126 (JIS X 0129-1) ソフトウェア品質特性

1.1.4 要求の優先順位付け

入力

(SW102) ソフトウェア機能要求リスト
(SW103) ソフトウェア非機能要求リスト

概要

ソフトウェア要求の優先順位を決定する。

出力

(SW104) 優先順位付けされたソフトウェア要求リスト

■ 実施内容

(SW102)、(SW103)をもとに、ソフトウェアとして実現する要求に優先順位付けを行う。

- ▶ 優先順位は必須/優先度高/優先度低/任意などとし、優先順位をつけた根拠や理由も記録しておく。

■ 注意すべき事項

● 優先順位付けの考慮点

優先順位の評価においては、

- ▶ 開発プロジェクトの期間や投入コスト、リソースなどの視点
- ▶ 新規の技術導入や技術習熟などの視点
- ▶ ビジネス面からの要求

などいくつかの視点を踏まえて、個々の要求の開発リスクの高い要求事項を考慮する(プロジェクトマネジメントなども交える)。

また、高リスクの要求は、ロングスパンでの製品戦略なども考慮して実現可能性を評価する。

1.1.5 ソフトウェア要求仕様書の作成

入力


(SW101) 制約条件リスト
 (SW102) ソフトウェア機能要求リスト
 (SW103) ソフトウェア非機能要求リスト
 (SW104) 優先順位付けされたソフトウェア要求リスト

概要

ソフトウェアの要求事項を整理し、ソフトウェア要求仕様書としてまとめる。

出力

(SW105) ソフトウェア要求仕様書 

 ドキュメント・テンプレート例が用意されています。

参照情報

IEEE Std 830 (ソフトウェア要求仕様書テンプレート)

■ 実施内容

(SW101)～(SW104)の成果を取りまとめてソフトウェア要求仕様書(SW105)を作成する。

- ▶ ソフトウェア要求定義の過程で出力された資料を整理・体系化して文書化する。
- ▶ 内部確認等での指摘事項が適切に反映されていることも確認する。

■ 注意すべき事項

- 文書化での留意点
 - ▶ 開発対象ソフトウェアに関連したエンドユーザ向けマニュアルも必要に応じて記述する。
 - ▶ 変更履歴を添付し、変更箇所なども明示する。
 - ▶ 作成文書に関しての責任所在を明示する。
- ▶ 作成文書は構成管理および変更管理にて確実に管理する。
 関連プロセス：SUP5 構成管理、SUP7 変更管理

SWP1.2 ソフトウェア要求仕様の確認

定義したソフトウェア要求事項がシステムとしての要求事項等を満たしていることを確認する。

1.2.1 ソフトウェア要求仕様書の内部確認


入力


製品企画書
(SY106) システム要求仕様書
(SY205) システム・アーキテクチャ設計書
(SA104) 安全要求仕様書
(SW105) ソフトウェア要求仕様書

概要

ソフトウェア要求仕様書がソフトウェアとして求められる事項を満たしているか確認し、内部確認レポートとしてまとめる。

出力

(SW106) 内部確認レポート (ソフトウェア要求定義) 

 ドキュメント・テンプレート例が用意されています。

■ 実施内容

以下の①～⑦の視点で、ソフトウェア要求仕様書 (SW105) の内部確認を行う。確認結果は内部確認レポート (SW106) として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

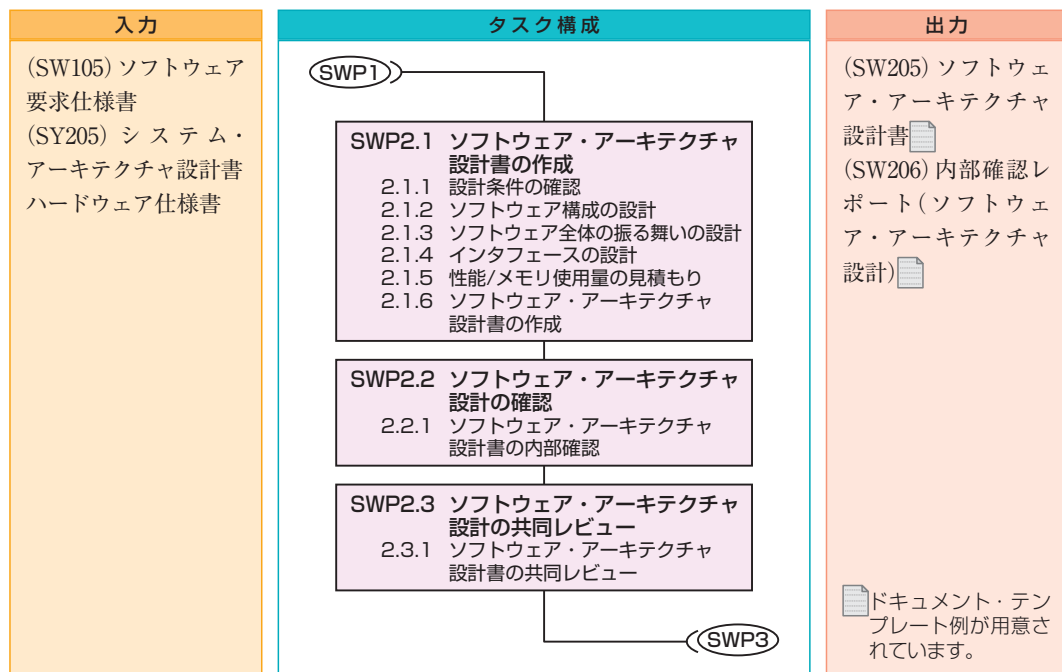
- ① 妥当性を評価する
 - ▶ 記述されている要求事項がシステム要求などと照らし合わせて妥当であるか。
- ② 実現可能性を評価する
 - ▶ 記述されている要求事項は開発部門が保有する技術によって実現可能かテスト可能性の評価。
- ③ テスト可能性 (ソフトウェア要求はテスト可能か) を評価する
 - ▶ システムが想定する動作フィールドにおいてのテストや動作確認が可能な要求であるか。
- ④ 運用・保守性を評価する
 - ▶ ソフトウェアのバージョンアップの考え方やその方式は適切であるか。
 - ▶ 不具合があった場合の対処方法が適切に検討されているか。
- ⑤ 追跡可能性を評価する
 - ▶ ここで定義されたソフトウェア要求事項に沿って以降の開発作業が進められていることが確認できるように、個々のソフトウェア要求事項に識別情報が付けられているか。
- ⑥ 一貫性を評価する
 - ▶ 関連する成果物の内容と矛盾がないか、また、ソフトウェア要求仕様書の内部で矛盾がないか。
- ⑦ 完全性を評価する
 - ▶ 機能、性能、設計上の制約、属性、外部インタフェースなど、必要事項がすべて記述されているか。
 - ▶ システムの動作において想定可能なあらゆるコンテキストに対して、その条件下でのソフトウェアの振る舞いが定義されているか。
 - ▶ 要求事項は一意にしか解釈できない記述になっているか。

■ 注意すべき事項

- 妥当性の観点から以下を考慮する。
 - ▶ エンドユーザのニーズを満たした機能が提供されているか。
 - ▶ 製品開発戦略に合致した機能が提供されているか。
 - ▶ システムで用いるハードウェアなどの制約事項に適合しているか。
 - ▶ システム要求仕様に合致しているか。
- 実現可能性の観点から以下を考慮する。
 - ▶ ソフトウェアでは性能が出ない可能性が高い場合は、一部の機能をハードウェアで実現することも考慮しているか。
 - ▶ 再利用ソフトウェアの制約を考慮しているか。
- テスト可能性の観点から以下を考慮する。
 - ▶ テスト環境を整備することが可能か
 - ▶ テスト環境の整備にかかる工数が、製品開発スケジュールの範囲内で収まるか
 - ▶ 実機を伴ったテストの場合、実機を破壊することなくテストが可能か
 - ▶ ハードウェアの整備も含めテスト環境の準備やテストデータの作成が可能か
- 運用・保守性の観点から以下を考慮する。
 - ▶ 修正したソフトウェアは、ネットワークにより配信可能か
 - ▶ 保守要員がROM交換などにより対応するのか
- 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション(共同レビューの開催など)などもあわせて記載しておくことが望ましい。
- 要求段階での問題は放置すると開発後半において大きな問題を生むため、この段階で内部確認レポートの情報は、プロジェクトマネージャ/開発リーダーなどステークホルダ間で情報共有とコンセンサスを得ておくことが望ましい。

SWP2 ソフトウェア・アーキテクチャ設計

開発する組込みソフトウェアのアーキテクチャ（＝動作（振る舞い）と構造）を決定し、アーキテクチャ設計書を作成する。



■ 解説

このアクティビティは

- (2.1.1) ソフトウェア要求仕様をもとに、実現すべき機能、非機能要求を確認する。また、要求実現上の制約などを仕様実現の視点で確認する。
- (2.1.2) (2.1.3) 要求仕様を実現するためのソフトウェアとしての振る舞いと構成を検討し、
- (2.1.4) ソフトウェアを構成する機能ユニット間のインタフェースを設計する。
- (2.1.5) また、ソフトウェアを実装するハードウェアを考慮し、性能やメモリ量なども検討し見積もる。
- (2.1.6) これらの検討結果を整理し、ソフトウェア・アーキテクチャ設計書を作成する。
- (2.2.1) 作成したソフトウェア・アーキテクチャ設計書はあらかじめ確認ポイントを決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。
- (2.3.1) また、関係者を交えて共同レビューを実施し、レビューの結果を整理して共同レビュー報告書を作成する。

ソフトウェア・コンポーネントの利用や、過去のソフトウェア資産の再利用も視野に入れて考えていく。

本ガイドでは、以下のように組込みソフトウェアを構成する要素を階層的にとらえている。

組込みソフトウェア = {機能ユニット群} の集合

機能ユニット = {プログラムユニット群} の集合

■ 留意事項

- ▶ 組込みシステムの振る舞いは動作環境条件(コンテキスト*)によって異なる。このためシステムがおかれるさまざまなコンテキストを分析し、それに対応するソフトウェアの振る舞いを設計する。
- ▶ 共通する処理などを局所化、共通化を図りながら機能ユニットを抽出する。
- ▶ アーキテクチャ設計はモデリング手法などを活用し、設計結果を可視化できるようにしておくことが望ましい。
- ▶ アーキテクチャ設計については求められる安全度のレベルなどを考慮し以下の事項についても検討しておく。**安全**
- ・設計手法およびその選択理由を明確にする
- ・アーキテクチャの前提や構成要素を明確にする
- ・ソフトウェア/ハードウェアの関係とインタラクションを明確にする
- ・アーキテクチャの設計表現に曖昧性を含まないようにする
- ・システムで扱う全てのデータを明らかにする
- ・設計したアーキテクチャの確認をするためのテスト方法を明確にする

■ 【参考】手法およびツール

- ▶ OOD (Object Oriented Design : オブジェクト指向設計)
- ▶ 設計モデリング手法
 - ・ UML (Unified Modeling Language : 統一モデリング言語) など
- ▶ 構造化分析設計

● 用語解説

* コンテキスト: コンテキストとは文脈や背景あるいは前後関係といった関連性のこと。ここでは、開発する組込みシステムの目的、背景や戦略などが例として挙げられる。

安全: セーフティに関連する作業

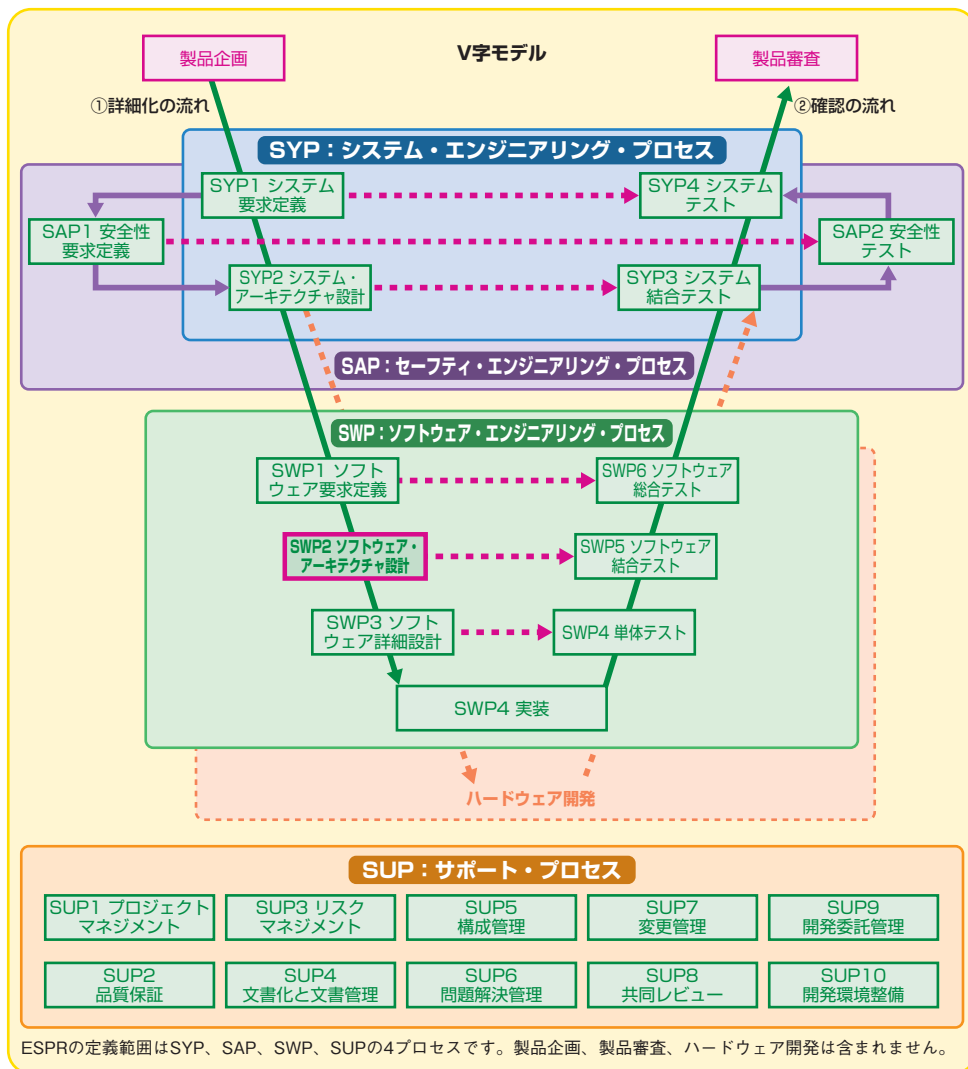


図2.8 V字モデルと開発プロセス (SWP2 ソフトウェア・アーキテクチャ設計)

■ SWP2.1 ソフトウェア・アーキテクチャ設計書の作成

ソフトウェア要求仕様書で定義された要求の実現方法(ソフトウェア・アーキテクチャ)を検討し、ソフトウェア・アーキテクチャ設計書としてまとめる。

2.1.1 設計条件の確認

入力

(SW105)ソフトウェア要求仕様書
(SY205)システム・アーキテクチャ設計書

概要

ソフトウェア・アーキテクチャを設計するにあたっての要求・条件を確認する。

出力

(設計条件確認メモ)

参照情報

各ハードウェア仕様書

■ 実施内容

ソフトウェアの設計を進めるにあたり、ソフトウェア要求仕様書(SW105)に記述された機能要求、非機能要求や前提とする制約事項などを確認しておく。

①ソフトウェア機能要求を確認する

- ▶ソフトウェア要求仕様書に記述されたソフトウェア機能要求の内容をもとに、どのような機能をソフトウェアとして実現することが求められているか確認する。

②ソフトウェア非機能要求を確認する

- ▶ソフトウェア要求仕様書に記載されたソフトウェア非機能要求を確認する。

③制約条件を確認する

- ▶ソフトウェア要求仕様書に記載されたソフト

ウェア機能を再考し、それらの機能を実現するためのソフトウェア・アーキテクチャの実現という視点から、下記のような制約条件も併せて確認する。

- ・ソフトウェア作成条件(使用OS、言語など)
- ・ハードウェア条件(割込み方式など)
- ・性能条件、稼動環境など
- ・安全性要求 **安全**

■ 注意すべき事項

- 特に下記に示す事項については重点的に確認する。
 - ▶性能要求の具体的条件、異常/例外の動作仕様
 - ▶メモリ量、外部記憶媒体の種類と容量
 - ▶ハードウェア機器/素子の種類・仕様・処理能力
 - ▶既存ソフトウェア(OS等を含む)利用時の調査
- システムに求められるセキュリティなどの面についても検討しておく。
- システムに求められる安全に関する要求(安全度のレベルとそれを実現する機能など)を確認する。 **安全**
- ソフトウェア要求仕様書の漏れや未決定事項などがないかを確認する。

安全 : セーフティに関連する作業

2.1.2 ソフトウェア構成の設計

入力

(SW105) ソフトウェア要求仕様書
(SY205) システム・アーキテクチャ設計書

概要

ソフトウェアの構成および機能ユニットの機能を設計する。

出力

(SW201) ソフトウェア構成設計書
(SW202) 機能ユニット設計書

参照情報

各ハードウェア仕様書

■ 実施内容

ソフトウェア要求仕様を実現するために必要となる、ソフトウェアとしての個別要素(機能ユニット)を検討し、整理する。

①機能ユニットを抽出する。

- ▶ ソフトウェアとして実現する機能を明確にし、共通する処理などを局所化、共通化を図りながら機能ユニットを抽出する。また、機能ユニットの抽出の際には右に示すようなソフトウェア非機能要求にも十分考慮する。

- ▶ 信頼性(リカバリ/データ保証など)
- ▶ 保守性(不具合追跡など)

②機能ユニットを詳細化する。

- ▶ 個々の機能ユニットは、ソフトウェア詳細設計ができるレベル(粒度)まで構成を詳細化する。

■ 注意すべき事項

●ソフトウェア構成の設計では以下に注意する

- ▶ 保守性、開発効率、テスト効率性、信頼性、拡張性、安全性を考慮する。
- ▶ 変更/修正のコスト、品質確保を考慮した機能の局所化/カプセル化
- ▶ 開発効率/メモリ効率を考慮した部品化/共通化
- ▶ デバッグ/テスト/保守のための動作ログ/取り出し機構具備
- ▶ 出荷後のソフトウェア変更機構の具備
- ▶ フェイルセーフ機構の具備

●設計の進め方

- ▶ ソフトウェア・アーキテクチャ設計では、各ソフトウェア詳細設計担当者に割り振れるレベルまで機能ユニットを詳細化する。
- ▶ 構成設計は、振る舞い、インタフェース設計と連携して実施される。

●その他の考慮点

- ▶ 機能ユニットの位置/関係が分かる構成図(機能ユニット関連図)を作成する。
- ▶ 製品シリーズ化などを考慮し、ソフトウェアのコアとなる機能ユニットは明確に区分しておく。

2.1.3 ソフトウェア全体の振る舞いの設計

入力

(SW105) ソフトウェア要求仕様書
(SY205) システム・アーキテクチャ設計書

概要

ソフトウェア全体の振る舞いを設計する。

出力

(SW203) ソフトウェア動作設計書

参照情報

各ハードウェア仕様書

■ 実施内容

ハードウェアを含めたシステムがどのような動的振る舞いをするかを考え、整理する。

- ▶ システムの振る舞いのなかで、ソフトウェアが担う動作を明確にする。
- ▶ ソフトウェア動作の前提となるコンテキストを明確にし、ソフトウェア要求仕様に記載された動作シナリオや動作シーケンスを検討する。特に具体的な動作に伴うシステムとしての動作シーケンス上の操作性も考慮する。特に組み込みソフトウェアでは下記の事項を考慮する。
 - ・ 並行処理動作：ハード割込み動作、タスク・プライオリティと並行処理/タスク遷移など
 - ・ 異常・例外発生：過負荷、ハード障害など
 - ・ データ処理の流れ：入力から出力までのデータ処理過程

■ 注意すべき事項

- 動作/制御の表現方法は、ビジュアルなものを併用し分かりやすくする
 - ▶ 振る舞いの整理については状態遷移モデル他の手法を利用して整理する。
- 組み込みソフトウェアの振る舞い設計では下記のような点に留意する
 - ▶ 性能考慮時：並行動作とタスクプライオリティ、SRAM/キャッシュへの割り当て
 - ▶ リアルタイム性：割込み制御とハード割込みレベル設定
 - ▶ 多重処理：排他制御(資源の確保/開放/競合)
 - ▶ 異常/例外処理：DoS攻撃対応、リカバリ動作、ノイズによるデータ化け、MPU誤動作、チャタリング
 - ▶ メモリ配置の考慮：常駐/非常駐、ROM/RAM
 - ▶ マルチ・プロセッサ構成時：負荷分散、資源排他制御
 - ▶ 各機能ユニット内部および機能ユニット間の状態定義、状態遷移を明確化
 - ▶ 各機能ユニットの時間制約

2.1.4 インタフェースの設計

入力

(SW105) ソフトウェア要求仕様書
(SY205) システム・アーキテクチャ設計書

概要

機能ユニット間のインタフェースを設計する。

出力

(SW204) ソフトウェア・インタフェース設計書

参照情報

各ハードウェア仕様書

■ 実施内容

ソフトウェアを構成する機能ユニット間のインタフェースを設計する。

- ①メモリー・レイアウトを設計する
 - ▶ メモリ全体のレイアウトを設計する。
- ②メモリー空間・領域を詳細化する
 - ▶ ソフトウェアとして利用する各領域の詳細を決める。
 - ・ 領域全体の名称/構成/サイズを決める
 - ・ 領域内フィールドの名称/位置/形式/サイズ/意味/初期値/アクセス元など
- ③機能ユニット間インタフェースを設計する
 - ▶ 呼び出し手順、パラメータ、復帰情報等を設計する。
- ④共通情報を一元化、論理値化する
 - ▶ 変更に対して修正負荷の高い情報は、一元化し、かつ論理値化しておく。
 - ・ テーブル・オフセット
 - ・ 共通定数など

■ 注意すべき事項

- インタフェース設計の考慮点
 - ▶ 異常/例外コード、メッセージを早めに抽出し体系化し、一覧表として整理する。
 - ▶ 排他的処理が必要な共通データについては更新/参照元を明確化する。
 - ▶ 共通リソースについては、確保/開放タイミングを明らかにする。
 - ▶ ビット構成等では拡張性を考慮する。

2.1.5 性能/メモリ使用量の見積もり

入力

(SY205) システム・アーキテクチャ設計書
(SW201) ソフトウェア構成設計書
(SW203) ソフトウェア動作設計書
(SW204) ソフトウェア・インタフェース設計書

概要

性能条件、メモリ量条件を検討し、性能、メモリ使用量を見積もる。

出力

(性能試算資料)
(メモリ使用試算資料)

■ 実施内容

①性能を見積もる。

- ▶ 正確で、最悪条件での性能を見積もるようにする。
- ・ クリティカルな要素は実測に基づくデータを参考にする
- ・ 過負荷時のシステム全体のスループットを算出する

- ・ プログラム・コード部とテーブル、バッファ等のデータ部の両者
- ・ 動的な観点での見積もり
- ・ ROM、RAM (SRAMを含む)、キャッシュなどすべて

- ▶ 必要により、ハードディスクなど他のハードリソースについても見積もる。

②メモリ使用量を見積もる。

- ▶ 過負荷時のスタックなど漏れないように見積もる。

■ 注意すべき事項

● 性能見積もりの考慮点

- ▶ ハードウェア特性(入出力機器の処理性能/応答性能、メモリ・ウェイトなど)
- ▶ 立ち上がり性能、最悪のケースを考慮
- ▶ OS処理時間の考慮など
ソフトウェアで実現する個々の処理の時間や応答性能は、ユーザの操作性などにも影響を与えるため実現方法を含め慎重に検討する。

● メモリ量見積もりの考慮点

- ▶ 処理多重度およびネスト数によるスタック量、OS領域(タスク制御テーブルなど)
- ▶ コード部の見積もりは、規模見積もりから換算する。

2.1.6 ソフトウェア・アーキテクチャ設計書の作成

入力

(SW201) ソフトウェア構成設計書
(SW202) 機能ユニット設計書
(SW203) ソフトウェア動作設計書
(SW204) ソフトウェア・インタフェース設計書
(性能試算資料)
(メモリ使用試算資料)

概要

ソフトウェア・アーキテクチャ設計の事項を整理し、ソフトウェア・アーキテクチャ設計書とまとめる。

出力

(SW205) ソフトウェア・アーキテクチャ設計書



ドキュメント・テンプレート例が用意されています。

■ 実施内容

(SW201)～(SW204)の成果を取りまとめてソフトウェア・アーキテクチャ設計書(SW205)を作成する。

- ▶ ソフトウェア・アーキテクチャ設計の過程で出力された資料を整理・体系化して文書化する。
- ▶ 内部確認、共同レビュー等での指摘事項が適切に反映されていることも確認する。

■ 注意すべき事項

● 参考資料等について

- ▶ 設計の妥当性を示す資料(性能見積り、資源使用量見積り、制御方式等)、およびどのような観点で設計したか(設計方針)を別資料で残すと、仕様変更、トラブル対処時に有効である。
- ▶ 変更/追加内容が確認できるような版数管理および改版リストの添付が望ましい。

● 文書化での留意点

- ▶ 機能ユニット名、タスク/プロセス名を体系的に整理しておく
- ▶ 変更履歴を添付し、変更箇所を明示する。
- ▶ 作成文書に関しての責任所在を明示する。
- ▶ 作成文書は構成管理および変更管理にて確実に管理する。

関連プロセス：SUP5 構成管理、SUP7 変更管理

■ SWP2.2 ソフトウェア・アーキテクチャ設計の確認

設計されたソフトウェア・アーキテクチャがシステム要求、ソフトウェア要求を満たしていることを確認する。

2.2.1 ソフトウェア・アーキテクチャ設計書の内部確認

入力

(SW105) ソフトウェア要求仕様書
(SW205) ソフトウェア・アーキテクチャ設計書
(SW201) ソフトウェア構成設計書
(SW202) 機能ユニット設計書
(SW203) ソフトウェア動作設計書
(SW204) ソフトウェア・インタフェース設計書
他

概要

ソフトウェア・アーキテクチャ設計がソフトウェアの要求事項を満たし、かつ、機器/システムとしての動作がシステム仕様を満たしているか確認し、内部確認レポートとしてまとめる。

出力

(SW206) 内部確認レポート(ソフトウェア・アーキテクチャ設計)

ドキュメント・テンプレート例が用意されています。

参照情報

各ハードウェア仕様書 他

■ 実施内容

- ①ソフトウェア・アーキテクチャ設計書(SW205)の内容が適切であるかどうかを確認する。確認結果は内部確認レポート(SW206)として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。
 - ▶構成要素である機能ユニットの集合が、システム要求、ソフトウェア要求を正しく実現できるかを確認する。
 - ・機能ユニットの機能の明確性・妥当性
 - ・機能ユニットの集合として振る舞いの明確性・妥当性
 - ・機能ユニット間のインタフェースの明確性・妥当性
 - ▶ソフトウェア非機能要求を十分に反映し、運用/テスト/保守の実現が可能かを評価する。
 - ・使用者のレベルを考慮した分かりやすさと安全性が考慮されているか
 - ・稼働状態(性能、負荷ピーク、長時間稼働)の考慮がされているか
 - ・テスト/保守性の容易性が考慮されているか
- ②機能ユニットの詳細化設計を確認する
 - ▶下記の基準を考慮して、個別のソフトウェア・アーキテクチャ設計を評価する。
 - ・機能、インタフェース、振る舞いの妥当性
 - ・具体化レベルと詳細設計の実現可能性

- ・設計標準への準拠(設計手法/表記法/用語/分かりやすさ/名称の体系化)
- ・既存ソフトウェア/市販ソフトウェア/オープンソフトウェアを利用する場合の十分な調査
- ・他プラットフォームへの移植性、機能拡張性の容易性

- ③ソフトウェア要求との対応(トレーサビリティ)が取れているか確認する

■ 注意すべき事項

- ソフトウェア・アーキテクチャのレビューを開催する際の留意点
 - ▶ **開催時期**：大きな設計手戻りを生じないように配慮して決める。すべての設計の完了を待たずに、部分的にも完了次第実施する。
 - ▶ **対象とする部分**：クリティカルな部分、新規参加者の設計分を中心に実施する。
 - ▶ **レビューア**：技術力の高いメンバをアサインする。必要に応じ、ハードウェア設計者等の参加も検討する。
 - ▶ **記録内容**：プロジェクトの標準に則るとともに、設計品質改善を考慮した事項を記録することが望ましい。(実施日時、実施時間、評価モジュール、レビューア、問題、原因、課題、対策等)
- 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション(共同レビューの開催など)などもあわせて記載しておくことが望ましい。
 - ▶ アーキテクチャ設計段階での問題は放置すると開発後半において大きな問題を生むため、この段階で内部確認レポートの情報は、プロジェクトマネージャ/開発リーダーなどステークホルダ間で情報共有とコンセンサスを得ておくことが望ましい。

レビューはソフトウェア品質を確保するための最も手軽で確実な方法

設計・実装途中の定量データに基づき問題箇所をチェック

複雑な箇所
 やたらに遅れている箇所
 未経験者が作った箇所
 周辺とのI/Fが複雑な箇所
 使用が曖昧な箇所

過去の不具合事例を参考に重要箇所をチェック

類似製品でバグを出した箇所
 顧客の利用頻度が高い箇所
 異常処理などが重視される箇所

関係者がFace to Faceで情報交換&確認

図2.9 レビューのチェックポイント

■ SWP2.3 ソフトウェア・アーキテクチャ設計の共同レビュー

設計されたソフトウェア・アーキテクチャがシステム要求、ソフトウェア要求を満たしていることをステークホルダ間で評価する。

2.3.1 ソフトウェア・アーキテクチャ設計書の共同レビュー

入力

(SW205) ソフトウェア・アーキテクチャ設計書
(SW206) 内部確認レポート(ソフトウェア・アーキテクチャ設計)

概要

ソフトウェア・アーキテクチャ設計書の共同レビュー*を実施し、ソフトウェア要求事項の実現方法の妥当性をステークホルダ間で確認する。

出力

(SU801) 共同レビュー記録(ソフトウェア・アーキテクチャ設計)
(SW207) ソフトウェア・アーキテクチャ設計共同レビュー報告書

ドキュメント・テンプレート例が用意されています。

参照情報

- ・(SY106) システム要求仕様書
- ・(SY205) システム・アーキテクチャ設計書
- ・(SW105) ソフトウェア要求仕様書

■ 実施内容

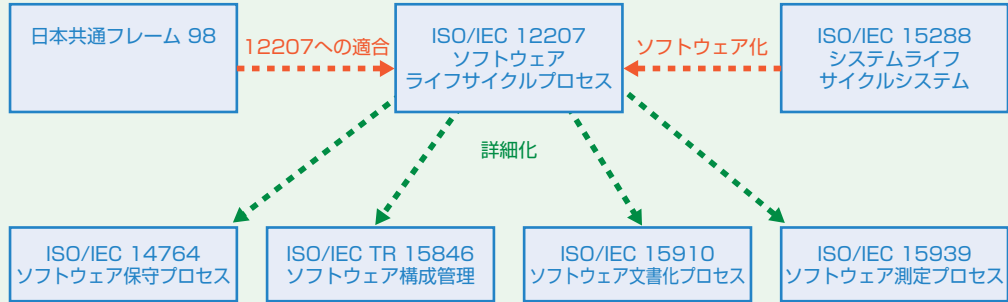
- ①プロジェクト計画に従い関係者全員により、機器/システムとしてのソフトウェア・アーキテクチャ設計の妥当性を確認する。
 - ▶レビューでは特に下記の点を考慮する。
 - ・ソフトウェア要求の実現可能性と要求の変更点/制限の明確性
 - ・保守/セキュリティを含めた実運用への対応性
 - ・後工程へのスムーズな移行性
 - ・ハードウェアを含めたプロジェクト計画/標準/規約の遵守(スケジュール、ドキュメント、品質記録、技法/手法)
- ②調整課題発生時には共同レビューを開催する
 - ▶アーキテクチャ設計中に発生した、他の開発グループに跨る問題の解決を図る場合に開催する。
 - ▶設計遅れ、手戻りの発生を避けるべく、発生した問題の早期解決を図ることに重点を置く。
- ③レビューの記録をもとに、共同レビュー報告書を作成する。
 - ・レビュー報告にはレビューの指摘事項やその対応についても明記し、関係するステークホルダーに配布する関連プロセス：SUP8 共同レビュー

●用語解説

- * **共同レビュー**：開発作業の節目ごとに各プロセスでの作業結果が適切であったかどうかを、関係者間で技術面、管理面の両面から確認するレビュー。共同レビューでは、成果物の作成を担当した技術者だけでなく、製品開発に関係するステークホルダなども参加し、多視点でのレビューを行う。

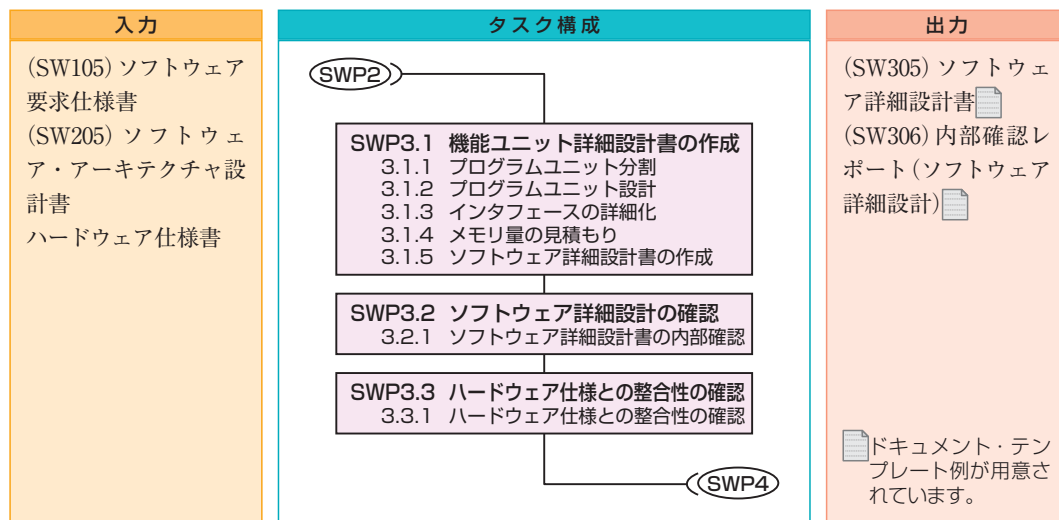
Related Standards

開発プロセス関連規格



SWP3 ソフトウェア詳細設計

ソフトウェア・アーキテクチャ設計で定義された機能ユニットをプログラムユニットに分割し、詳細な振る舞いや論理構造などを設計する。



■ 解説

このアクティビティは

- (3.1.1) ソフトウェア要求仕様とソフトウェア・アーキテクチャ設計をもとに、当該システムとして提供する機能を実現するために、組込みソフトウェアを実装の視点からいくつかの部分 (プログラムユニットと呼ぶ) に分割し、
- (3.1.2) それぞれのプログラムユニットで行うべき処理内容や論理処理を明確にするとともに、
- (3.1.3) それぞれのプログラムユニット間の関係を明確にし、そのインタフェースを決め、
- (3.1.4) 実装の視点から、使用するメモリ量の見積もりを出す。
- (3.1.5) これらの検討結果を整理し、ソフトウェア詳細設計書を作成する。
- (3.2.1) 作成したソフトウェア詳細設計書はあらかじめ確認ポイントを決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。
- (3.3.1) また、関係者を交えてハードウェア仕様との整合性の確認を行い、確認した内容を整理してハードウェア仕様との整合性の確認レポートを作成する。

■ 留意事項

- ▶ 組込みシステムの場合、システムが動作する環境によって決まるデータ (エアコンの外気温等)などを十分に考慮して各ユニットの詳細を検査する。
- ▶ ソフトウェアが動作するハードウェア環境や条件についても十分に検討する。
- ▶ ハードウェア設計グループとの情報のすりあわせに注意する。

▶ 詳細設計着手にあたっては、システムの安全要求、ソフトウェア・アーキテクチャ設計、ソ

フトウェアの安全性の確認方法が文書化されていることを確認する。**安全**

■【参考】手法およびツール

▶ OOP (Object Oriented Programming : オブジェクト指向プログラミング)

▶ 構造化分析設計

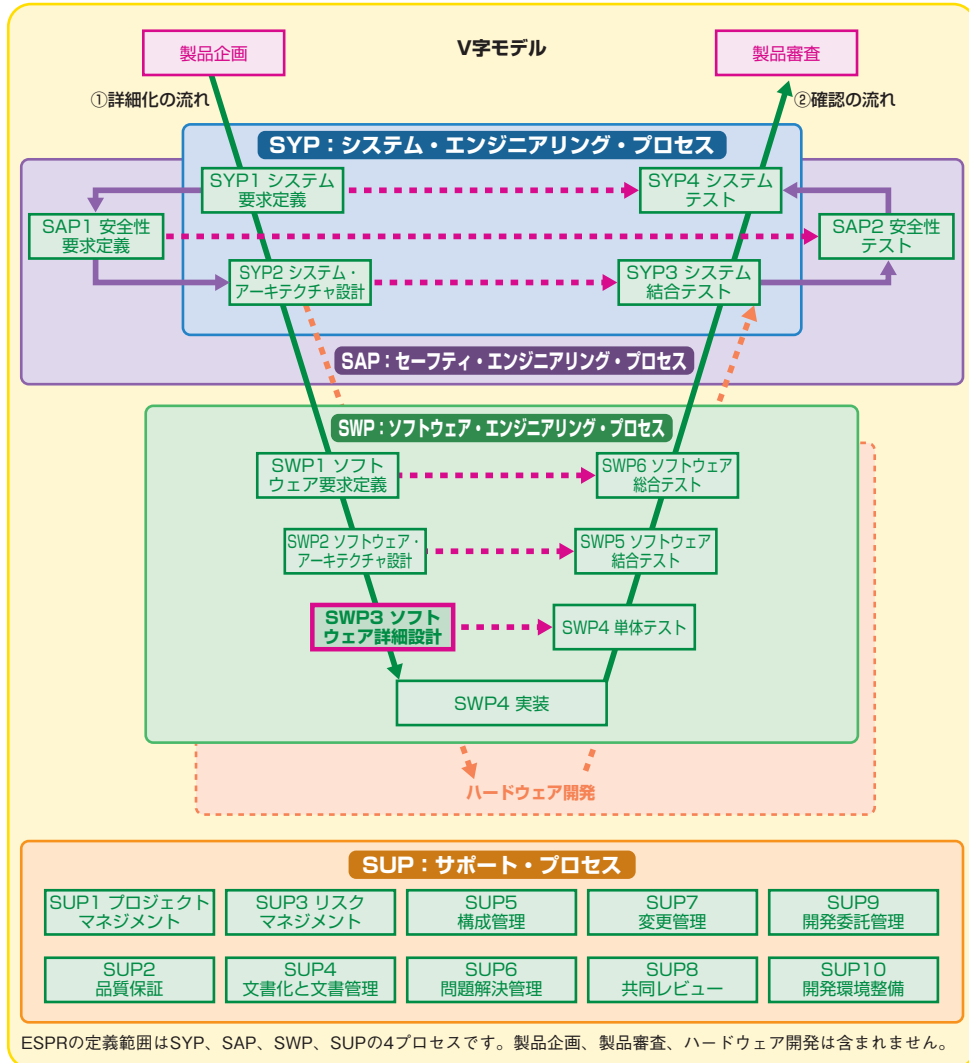


図2.10 V字モデルと開発プロセス (SWP3 ソフトウェア詳細設計)

安全 : セーフティに関連する作業

■ SWP3.1 機能ユニット詳細設計書の作成

ソフトウェア・アーキテクチャ設計書の機能ユニット仕様をもとに実装可能なレベル(プログラムユニット)まで詳細化し、ソフトウェア詳細設計書としてまとめる。

3.1.1 プログラムユニット分割

入力

(SW105)ソフトウェア要求仕様書
(SW205)ソフトウェア・アーキテクチャ設計書

概要

機能ユニットをプログラムユニットに分割する。

出力

(SW301)プログラムユニット機能/構成設計書

■ 実施内容

機能ユニットをプログラムユニットに分割し、プログラムユニットの構成と各々の機能を定義する。

- ▶ プログラムユニットは、実装・コンパイルおよび単体テストを実施する最小レベルまで分割する。

■ 注意すべき事項

● 分割の考慮点

- ▶ データフロー（データを処理する流れ）に着目する。
- ▶ 複数のプログラムユニットに共通の機能を洗い出して、共通機能として定義する。
- ▶ プログラムユニットの独立性(カプセル化)や相互の結合度などに注意する。

- ▶ テスト性、保守性、拡張性
- ▶ レビューのしやすさ
- ▶ 再利用性
- ▶ 処理漏れや処理間での矛盾がないように注意する。

3.1.2 プログラムユニット設計

入力

(SW205) ソフトウェア・アーキテクチャ設計書
(SW301) プログラムユニット機能/構成設計書

概要

プログラムユニットの処理内容を、実装可能なレベルまで詳細化する。

出力

(SW302) プログラムユニット設計書

参照情報

各ハードウェア仕様書

■ 実施内容

① プログラムユニットの処理内容を実装可能なレベルまで詳細化する

詳細化すべき事項としては、以下のようなものがある。

- ▶ ハードウェア制御方法、タイミング、設定値
- ▶ OSシステムコール、利用機能(汎用ライブラリ、共通機能等)の引数値
- ▶ 高速化が必要な処理は、その条件と手法
- ▶ 状態管理(状態遷移表等で、入力イベントに対してソフトウェアの振る舞いを決定するための状態を管理する)

- ▶ エラー処理
- ▶ リソース定義(ユニット内で使用するリソース)
- ▶ システム初期化処理

② 不具合を解析する際に利用する機能もプログラムユニットとして詳細内容を検討する

設計する事項としては、以下のようなものがある。

- ▶ ハードウェア制御時の状態(特にエラー発生時の情報)
- ▶ ソフトウェア実行状態の確認

■ 注意すべき事項

- ▶ 共通に使用する値を抽出して、具体的な数値を定める(define値として使用する値)。ソフトウェア・アーキテクチャ設計で定義された項目がある場合は、その項目について具体的な数値を定める。共通に使用する値の例としては、以下のようなものがある。
 - ・メモリのサイズ(テーブル、バッファなど)
 - ・エラー値
 - ・コンパイル条件
- ▶ 非同期で動作するプログラムユニット間の排他、イベント等を考慮する。
- ▶ 要求定義やアーキテクチャ設計に従って、製品シリーズ化などを考慮し、ソフトウェアのコアの部分を構成するプログラムユニット、製品バリエーションに対応するプログラムユニットなどを明確に区分する。
 - ・プロダクトライン*
- ▶ ユーザーインタフェースを実現するプログラムユニットについては、ユーザの操作性(ユーザの誤操作をまねくことがないよう)などを考慮する。
- ▶ 求められる安全度のレベルを考慮する(正確性、検証性・耐故障性など)。**安全**

● 用語解説

* **プロダクトライン**：ソフトウェアを小さな単位(ドメイン)に細分化して開発し、新たなソフトウェアを開発する際には既存のドメインを組み合わせる効率よく開発を進める手法。類似の仕様で多数のソフトウェアを開発しなければならない場合に有効である。

安全：セーフティに関連する作業

3.1.3 インタフェースの詳細化

入力

(SW205) ソフトウェア・アーキテクチャ設計書
(SW302) プログラムユニット設計書

概要

機能ユニット間およびプログラムユニット間のインタフェースを、実装可能なレベルまで詳細化する。

出力

(SW303) プログラムユニット・インタフェース設計書

■ 実施内容

ソフトウェアを構成する個々の機能ユニット、プログラムユニットそれぞれの間のインタフェースを詳細に定義する。

①機能ユニット間インタフェースを詳細化する

- ▶ アーキテクチャ設計で定義されたインタフェース仕様を実装可能なレベルまで詳細化する。以下について、構造、サイズ、意味、初期値を明確にする。
 - ・機能ユニットの入力
 - ・機能ユニットの出力
 - ・参照・設定するメモリ(テーブル、バッファ等)

②プログラムユニット間インタフェースを設計する

- ▶ インタフェースを実装可能なレベルまで詳細化する。以下について、構造、サイズ、意味、初期値を明確にする。
 - ・プログラムユニットの入力
 - ・プログラムユニットの出力
 - ・参照・設定するメモリ(テーブル、バッファ等)

■ 注意すべき事項

- ▶ メモリ(テーブル、バッファ等)を割り込み処理で使用する場合、タイミング等による不整合が発生しないようにする。

3.1.4 メモリ量の見積もり

入力

(SW205) ソフトウェア・アーキテクチャ設計書
(SW301) プログラムユニット機能/構成設計書
(SW302) プログラムユニット設計書
(SW303) プログラムユニット・インタフェース設計書

概要

メモリ量の詳細見積もりを行い、実装可能か否か確認する。

出力

(SW304) メモリ使用量(メモ)

■ 実施内容

ソフトウェアの実現に際して必要となるメモリ量を見積もる。

①メモリ量の詳細を見積もる。

- ▶ メモリの詳細な使用量を見積もる。見積もりはメモリ種別ごとに算出する(ROM、RAM、スタックエリア等)。

②実装可否を確認する。

- ▶ メモリ種別ごとに実装可能か否か確認する。容量を超える場合は詳細設計の見直し、またはメモリ量の変更をハードウェア開発側と検討する。

■ 注意すべき事項

●メモリ量見積もりでは以下に注意する。

- ▶ 仕様変更・追加を考慮した容量か(余裕があるか)。

- ▶ 不具合解析環境(エミュレータ等のエリア)を考慮した容量か。

3.1.5 ソフトウェア詳細設計書の作成

入力


(SW301) プログラムユニット機能/構成設計書
(SW302) プログラムユニット設計書
(SW303) プログラムユニット・インタフェース設計書
(SW304) メモリ使用量(メモ)
(SW309) ハードウェア仕様との整合性確認結果レポート(指摘事項反映の場合)

概要

ソフトウェア詳細設計の事項を整理し、ソフトウェア詳細設計書としてまとめる。

出力

(SW305) ソフトウェア詳細設計書 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

(SW301)～(SW304)の成果を取りまとめてソフトウェア詳細設計書(SW305)を作成する。

- ▶ ソフトウェア詳細設計の過程で出力された資料を整理・体系化して文書化する。
- ▶ 内部確認、ハードウェア仕様との整合性の確認等での指摘事項が適切に反映されていることも確認する。

■ 注意すべき事項

● 文書化での留意点

- ▶ 変更履歴を添付し、変更箇所を明示する。
- ▶ 作成文書に関しての責任所在を明示する。
- ▶ 作成文書は構成管理および変更管理にて確実に管理する。

関連プロセス：SUP5 構成管理、SUP7 変更管理

SWP3.2 ソフトウェア詳細設計の確認

詳細化された処理が実装可能なレベルであることを確認する。

3.2.1 ソフトウェア詳細設計書の内部確認

入力

(SW205) ソフトウェア・アーキテクチャ設計書
(SW305) ソフトウェア詳細設計書

概要

ソフトウェア詳細設計がソフトウェア・アーキテクチャの設計事項を満たし、かつ、実装可能なレベルか確認し、内部確認レポートとしてまとめる。

出力

(SW306) 内部確認レポート (ソフトウェア詳細設計)

ドキュメント・テンプレート例が用意されています。

■ 実施内容

- ① 下記の視点でソフトウェア詳細設計書の内容を確認する。
 - ▶ 個々のユニットの切り分けの正しさ、および各ユニットの処理内容の明確さ
 - ▶ ユニット間のインタフェース情報の整合性
 - ▶ 個々のユニットとハードウェアとの関係
- ② 確認結果は内部確認レポート (SW306) として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

■ 注意すべき事項

- 確認すべき事項としては、以下のようなものがある。
 - ▶ プログラムユニット構成の妥当性
 - ▶ 機能ユニットとプログラムユニット間の一貫性
 - ▶ 機能ユニット間インタフェースの整合性
 - ▶ プログラムユニット間インタフェースの整合性
 - ▶ メモリ (テーブル、バッファなど) の構造、排他制御、無駄なエリア
 - ▶ 各種設定値、引数値 (OS システムコール、汎用ライブラリなど)
 - ▶ 高速化が必要な処理は、その実現方法
 - ▶ ハードウェア制御方法、設定値
 - ▶ ハードウェア機能の効果の利用 (処理の高速化などを検討したか)
 - ▶ 無限ループ、デッドロック (発生する条件がなにか確認する)
 - ▶ 割り込み処理 (多重割り込み、終了処理など)
 - ▶ 実装に必要な情報は漏れなくソフトウェア詳細設計書に記載されているか
- 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション (共同レビューの開催など) などもあわせて記載しておくことが望ましい。

■ SWP3.3 ハードウェア仕様との整合性の確認

詳細化された処理がソフトウェアおよびハードウェア両方の側面から仕様の整合がとれていることを確認する。

3.3.1 ハードウェア仕様との整合性の確認

入力

(SW205) ソフトウェア・アーキテクチャ設計書
(SW305) ソフトウェア詳細設計書

概要

ソフトウェアおよびハードウェア両方の側面から仕様の整合がとれているか確認し、整合性確認結果レポートとしてまとめる。

出力

(SW307) ハードウェア仕様との整合性確認結果レポート

■ 実施内容

ハードウェアおよびソフトウェア双方の仕様を提示し、仕様の整合がとれているか確認する。確認結果はハードウェア仕様との整合性確認結果レポート(SW307)として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

■ 注意すべき事項

- ▶ ソフトウェアおよびハードウェアの設計を行う過程で、新たに発覚した不明瞭事項や、仕様の認識違いがないか確認し、次工程以降の手戻りを防止する。確認すべき事項としては、以下のようなものがある。
 - ・ ハードウェアの初期化手順
 - ・ タイミング
 - ・ ハードウェアの設定値
- ▶ 詳細設計レベルでは、ソフトウェアとハードウェアの不整合は、システムとしての不具合に直結する。このため、ここで作成した整合確認レポートは、ソフトウェア開発関係者のみでなく、ハードウェア開発関係者にも配布し確認をとる必要がある。

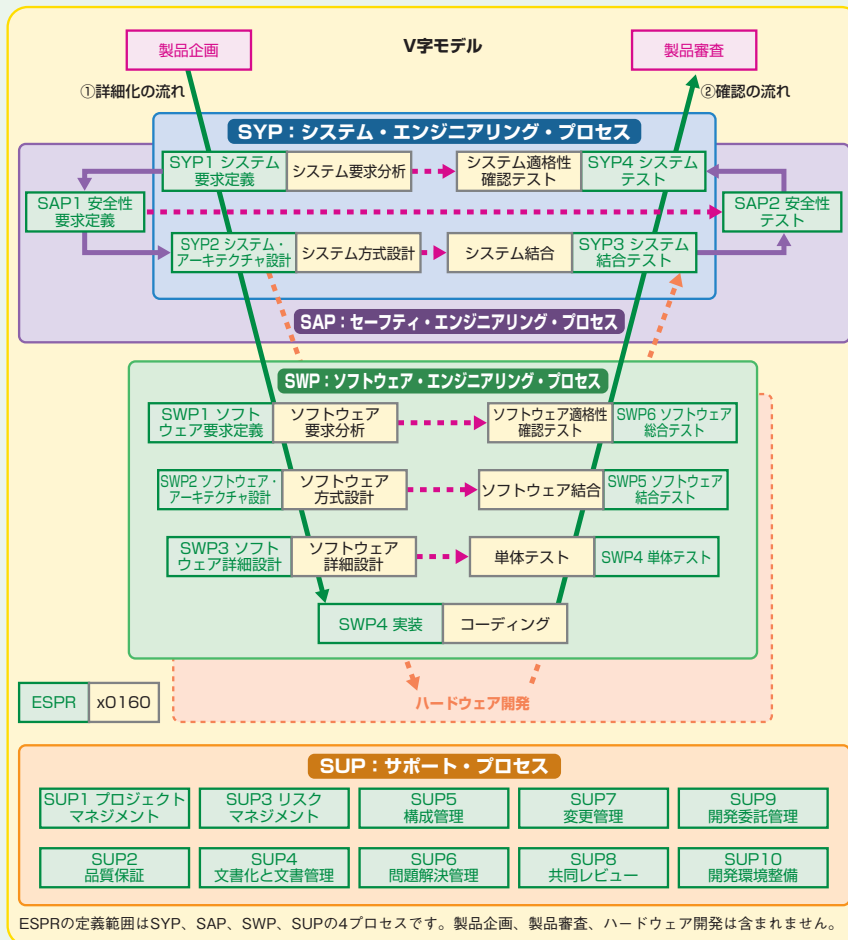
Related Techniques

V字モデル

本ガイドに記述されているアクティビティ間の関係はV字モデルで表すことができます。

V字モデルは、要求を詳細化してソフトウェアを実装する流れ(下図の①の線)と、ソフトウェアが要求のとおり正しく機能していることを確認する流れ(下図の②の線)を分け、それぞれの作業(アクティビティ)の対応関係を示したものです。

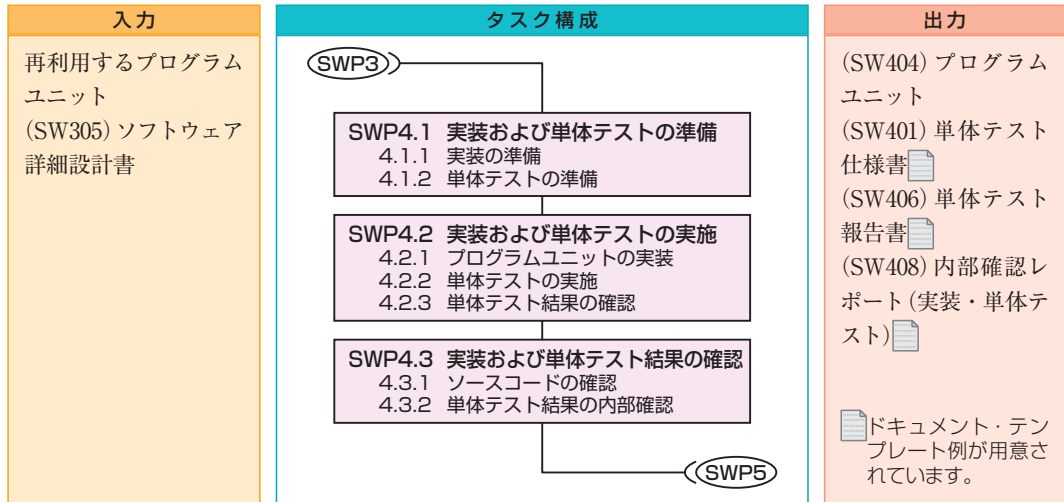
詳細化の流れでは、各作業の成果物としてドキュメント(仕様書や設計書)を作成していきます。その対応関係にある確認の作業では、作成されたドキュメントを参照してソフトウェアが正しく機能しているか確認していきます。従って、ドキュメントを作成する際にはテスト作業において確認することを意識して、機能要求、非機能要求等を明確に記述することが大切です。



本ガイド(Ver.2.0)の記述範囲

SWP4 実装および単体テスト

ソフトウェアを構成する個々のプログラムユニットの実装と単体レベルでの動作確認を行う。



■ 解説

このアクティビティでは

- (4.1.1) 組込みソフトウェアのプログラム実装を行うための開発環境や再利用するプログラムなどを用意し、
- (4.1.2) 実装したプログラムユニットの単体レベルでの確認の準備を行い、
- (4.2.1)(4.2.2)(4.2.3) 各プログラムユニットをソフトウェア詳細設計に従って実装し単体テストを実施する。
- (4.3.1)(4.3.2) また実装完了したプログラムユニットの実装内容および単体テスト結果は、あらかじめ確認ポイントを決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。

■ 留意事項

- ▶ 実装では、あらかじめ採用するコーディング作法・ルールを決めておく。
- ▶ 単体テストのためのテスト環境などの準備は遅れがちなので、計画どおりの手配に留意する。
- ▶ 単体テストのためのテスト環境などの準備は遅

■ [参考] 手法およびツール

- ▶ 言語処理ツール (コンパイラ、リンカ)
- ▶ ソースコード管理ツール
- ▶ デバッグツール (デバッガ、ICE : In-Circuit Emulator)
- ▶ テスト支援ツール
- ▶ ソースコード静的・動的チェックツール
- ▶ 品質メトリクス (ソースコード) 計測ツール
- ▶ コードクローン検出ツール (ソースコードの修正、リファクタリング、品質評価、コード剽窃の検出などに使われる技術)

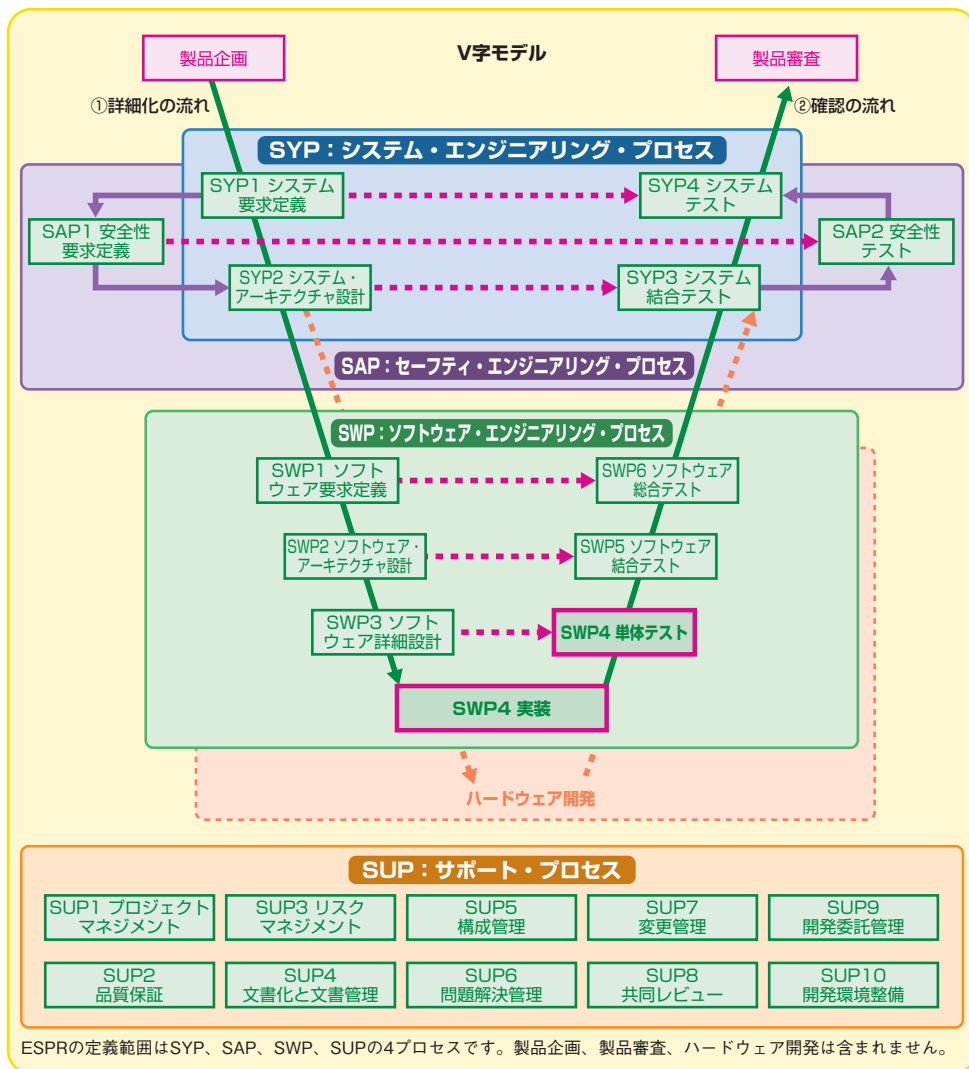


図2.11 V字モデルと開発プロセス (SWP4 実装および単体テスト)

■ SWP4.1 実装および単体テストの準備

ソフトウェア詳細設計で定義されたプログラムユニットの実装および単体テストの準備をする。

4.1.1 実装の準備

入力

再利用するプログラムユニット*

概要

再利用するプログラムや開発環境などを準備し、プログラムユニットの実装が実施可能な状態にする。

出力

(SU1002) ソフトウェア開発環境

■ 実施内容

- ① 再利用するプログラムユニットを準備する。
 - ▶ 開発済みのプログラムユニットを利用する場合は、該当するプログラムユニットを用意し、利用可能状態にあるか確認する。
- ② 開発環境を準備する。
 - ▶ プログラムユニットを実装するための開発環境を準備する。

■ 注意すべき事項

- 再利用するプログラムユニットについては以下の点を確認しておく。
 - ▶ バージョン
 - ▶ 品質 (既知の不具合など)
 - ▶ 利用する際の制約事項

● 用語解説

* 再利用するプログラムユニット: 新たなソフトウェアを開発する際に、既存のソフトウェアの「部品」を再利用すること。

4.1.2 単体テストの準備


入力


(SW305) ソフトウェア詳細設計書
(SU1002) ソフトウェア開発環境
(SU601) 不具合管理票
(修正確認の場合)

概要

単体テスト仕様書を作成し、単体テストが実施可能な状態にする。

出力

(SW401) 単体テスト仕様書 
(SW402) 単体テストデータ
(SW403) 内部確認メモ
(単体テスト仕様)

ドキュメント・テンプレート例が用意されています。

■ 実施内容

単体テストを実施するための下記①～④を準備する。

①単体テスト項目を準備する。

- ▶ ソフトウェア詳細設計書 (SW305) に定義された処理が、正しく実現されているか否かを確認するためのテスト項目を準備し、テスト仕様書としてまとめる。テスト項目として準備すべき項目としては、以下のようなものがある。
 - ・機能テスト
 - ・条件網羅テスト/データ境界値テスト

②テストデータを準備する。

- ▶ ①で準備したテスト項目を実行するために必要となるテストデータ (具体的な入力データなど) を作成する。

③スタブ/テストドライバ(擬似ソフトウェア)を作成する。

- ▶ テスト対象の機能ユニットを動作させるため、必要に応じてスタブ/テストドライバを作成する。

④テスト結果の判定基準や、テスト全体の評価基準や完了基準なども用意しておく。

⑤修正確認テスト項目を準備する(修正確認の場合)。

- ▶ テストで不具合などが検出され、その部分を修正した場合、不具合が解消されたこと、また修正により別の不具合が発生していないことを確認するためのテスト項目を準備する。
 - ・不具合の内容および修正内容などを考慮し、テスト範囲を決定しテスト項目を選択する

■ 注意すべき事項

● 単体テスト仕様書を確認する。

ここで確認すべき事項としては、以下のようなものがある。

- ▶ テスト項目数の妥当性
- ▶ テスト項目の網羅性(命令、条件、判定)

- ▶ テスト項目の矛盾/重複
- ▶ ハードウェア制御部、割り込みルーチンのテスト方法
- ▶ 高速化が必要な部分のテスト方法
- ▶ エラー処理

● テスト項目に与える入力値の考慮点

- ▶ 正常データ
- ▶ データ未設定 (NULL 値)
- ▶ 境界値 / 仕様値を大きく超える値 / 条件
- ▶ 最小値、最大値

● 修正確認テストの準備の考慮点

- ▶ テストの実施範囲は、影響する部分をすべて網羅しているか。
- ▶ テストは基本的に既に作成したテスト項目を再利用する。ただし、修正量が多い場合や、影響範囲が広い場合などは、新たなテスト項目を用意することも検討する(グローバル変数 / 共有メモリの使用範囲等に留意する)。

● 文書化での留意点

- ▶ 変更履歴を添付し、変更箇所を明示する。
- ▶ 作成文書に関しての責任所在を明示する。
- ▶ 作成文書は構成管理および変更管理にて確実に管理する。

関連プロセス：SUP5 構成管理、SUP7 変更管理

SWP4.2 実装および単体テストの実施

ソフトウェア詳細設計で定義されたプログラムユニットの実装および単体テストを実施する。

4.2.1 プログラムユニットの実装

入力

(SW305) ソフトウェア詳細設計書
(SU1002) ソフトウェア開発環境
(SU601) 不具合管理票
(不具合修正の場合)

概要

プログラムユニットを実装する。

出力

(SW404) プログラムユニット

参照情報

コーディング作法・規約

■ 実施内容

① プログラムユニットを実装する。

- ▶ ソフトウェア詳細設計書に基づいてプログラムユニットを実装する。
- ▶ コンパイルを実行して、文法上の誤りがあれば修正する。

② 利用プログラムユニットを確認する。

- ▶ 既存のソフトウェア資産や商用のソフトウェア

などを利用する場合には、それらのソースコードなどを確認し、利用する範囲を決める。

③ 不具合を修正する。

- ▶ ソースコードの確認やテストなどで不具合が検出された場合には、不具合管理票を確認してプログラムユニットの不具合を修正する。

■ 注意すべき事項

● 実装の際に注意すべき事項としては、以下のようなものがある。

- ▶ 高速化を実現する手法を用いる。(インラインアセンブラなど)
- ▶ メモリ制約に応じた最適化を実現する手法を用いる。
- ▶ 読みやすさやテストのしやすさに留意する。

- ▶ コーディング規約等を守る。
- ▶ コメントを記入して処理内容が分かるようにする。
- ▶ 不具合修正の場合は、修正履歴、修正内容が分かるようにする。

● 「実装」以降については、ソースコードの版管理(チェックイン・アウト)に留意する。

4.2.2 単体テストの実施

入力

(SW401) 単体テスト仕様書
(SW402) 単体テストデータ
(SW404) プログラムユニット
(SU1002) ソフトウェア開発環境
(SU601) 不具合管理票
(修正確認の場合)

概要

単体テスト仕様書に従って、単体テストを実施する。

出力

(SW405) 単体テスト結果(メモ)

■ 実施内容

①単体テストを実施する。

- ▶ 単体テスト仕様書(SW401)をもとに、単体テストを実施し、出力結果を得る。
- ▶ 代替テストを実施した場合は、理由を明記した上で記録を残す。
- ▶ 不具合検出時、継続して残りのテストを実施するか、不具合が修正されるまでペンディングするかを判断をする。
- ▶ 出力結果を収集する(各種ログなど)。
- ▶ 準備したテスト項目が実施できなかった場合は、理由を明記した上で記録を残す。また、その理由の妥当性について判断する。

②修正確認テストを実施する。

- ▶ 単体テストなどで検出された不具合を修正した場合には、不具合が解消されたかテストを実施する。
- ▶ 不具合の修正により、別の不具合が発生していないかテストを実施する。

■ 注意すべき事項

- 修正確認テストでは修正後の最新版でテストを実施しているかを確認する。

4.2.3 単体テスト結果の確認




入力

(SW401) 単体テスト仕様書
(SW405) 単体テスト結果(メモ)
(SU601) 不具合管理票
(修正確認の場合)

概要

テスト結果を確認し、テストの合否を判定する。

出力

(SW406) 単体テスト報告書 
(SU601) 不具合管理票 
ドキュメント・テンプレート例が用意されています。

■ 実施内容

- ①単体テスト結果を確認する。
 - ▶ テスト結果を確認して、実施したテスト項目の合否を判定する。
 - ▶ 修正確認テストの結果、不具合が解消された場合は、修正確認が済んだことを不具合管理票に記録する。
- ②不具合記録を作成する。
 - ▶ 不具合を発見した場合、不具合管理票に不具合内容を記録する。

■ 注意すべき事項

- テスト結果に関しては、単体テスト仕様書の誤りの可能性についても考慮する。
(単体テスト報告書は、単体テスト仕様書に合否を記入することで代替しても良い)

■ SWP4.3 実装および単体テスト結果の確認

実装されたプログラムユニットがソフトウェア詳細設計で定義された処理を正しく実現されているか、実装およびテスト結果を確認する。

4.3.1 ソースコードの確認

入力

(SW305) ソフトウェア詳細設計書
(SW404) プログラムユニット

概要

ソフトウェア詳細設計書に書かれている機能を実現する上で正しく実装(コーディング)されているか確認する。

出力

(SW407) 内部確認メモ
(ソースコード)

■ 実施内容

個々のプログラムユニットを実現するソースコードが正しく実装されているかどうかを確認する。

- ▶ ソフトウェア詳細設計書に書かれている機能が実現できるか確認する。
- ▶ 部門で採用しているコーディング作法やコーディング規約などつき合わせて確認する。確認すべき事項としては、以下のようなものがある。
 - ・ 高速化が要求されている処理は、要求が実現できるコードになっているか
 - ・ メモリ制約に応じた最適化を実現する手法を用いているか
 - ・ 読みやすい、テストしやすいコーディングになっているか
 - ・ コーディング規約等を守っているか
 - ・ コメントを記入して処理内容が分かるようにしているか
 - ・ 不具合修正の場合は、修正履歴、修正内容が分かるようにしているか

4.3.2 単体テスト結果の内部確認

入力

(SW305) ソフトウェア詳細設計書
(SW401) 単体テスト仕様書
(SW403) 内部確認メモ(単体テスト仕様)
(SW406) 単体テスト報告書
(SW407) 内部確認メモ(ソースコード)
(SU601) 不具合管理票

概要

単体テストで未解決の問題や未実施のテスト項目がないか確認し、内部確認レポートとしてまとめる。

出力

(SW408) 内部確認レポート(実装・単体テスト)

ドキュメント・テンプレート例が用意されています。

■ 実施内容

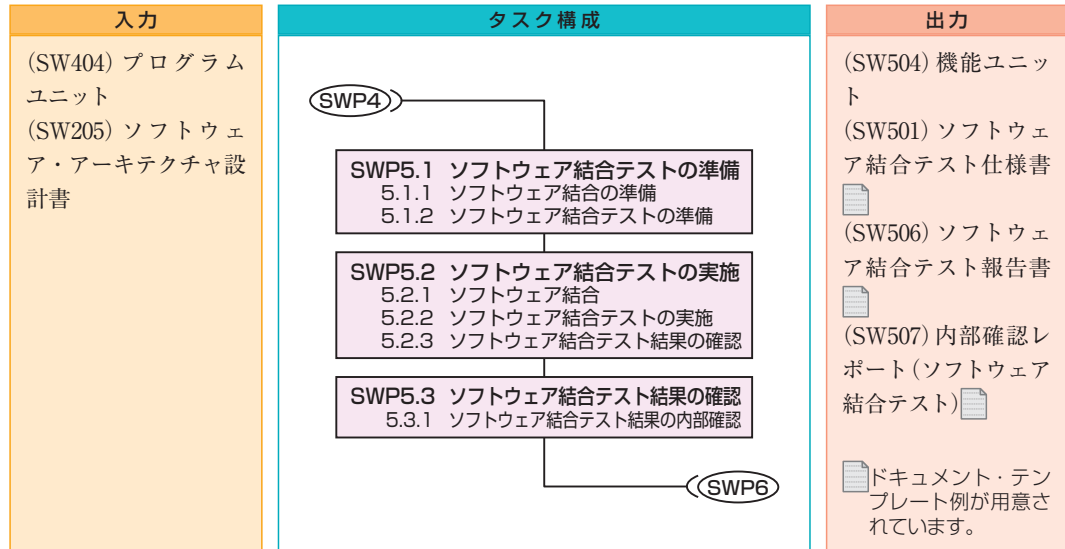
- ① 下記の視点で単体テストの結果を確認する。
 - ▶ 未解決となっている問題がないか。未解決となっている問題がある場合は、問題の内容を確認して早急に対応すべきか、次テストフェーズに持ち越すべきかを判断する。
 - ▶ 未実施となっているテスト項目がないか。未実施となっているテスト項目がある場合は、未実施となった理由を確認して対応を検討する。
- ② 確認結果は内部確認レポート(SW408)として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

■ 注意すべき事項

- 単体テストの結果については、以下の点も含めて確認する。
 - ▶ 不具合検出数は、品質基準を満たしているか。
 - ▶ 修正確認テストの場合は、範囲と環境は適切であったか。
- 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション(共同レビューの開催など)などもあわせて記載しておくことが望ましい。

SWP5 ソフトウェア結合テスト

ソフトウェアを構成する個々のプログラムユニットを順次組み立て、それぞれ組み合わせた際の機能が動作するかどうかを確認する。



■ 解説

このアクティビティでは

- (5.1.1) 個々のプログラムユニットを結合するためのmakeファイルをはじめとするソフトウェア結合の準備をして、
- (5.1.2) ソフトウェア結合レベルのテストの準備を進め、
- (5.2.1) (5.2.2) (5.2.3) 個々のプログラムユニットの結合と結合テストを行い、その結果を確認する。
- (5.3.1) ソフトウェア結合テスト結果はあらかじめ確認ポイントを決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。

組込みソフトウェアにおけるソフトウェア結合およびソフトウェア結合テストは、ソフトウェア要求、ソフトウェア・アーキテクチャ設計通りに実現できていることを確認していく作業と位置づけることができる。

■ 留意事項

- ▶ 組込みソフトウェアの結合は、スタブ/テストドライバなどを利用しながら順次プログラムユニットを結合していく。
- ▶ 結合テストの際の不具合などに対応する場合、結合対象のプログラムユニットのソースコードなどのバージョン管理に留意する。

■【参考】手法およびツール

- ▶ 境界値/限界値分析法など
- ▶ クリーンルーム手法
- ▶ コードクローン検出ツール

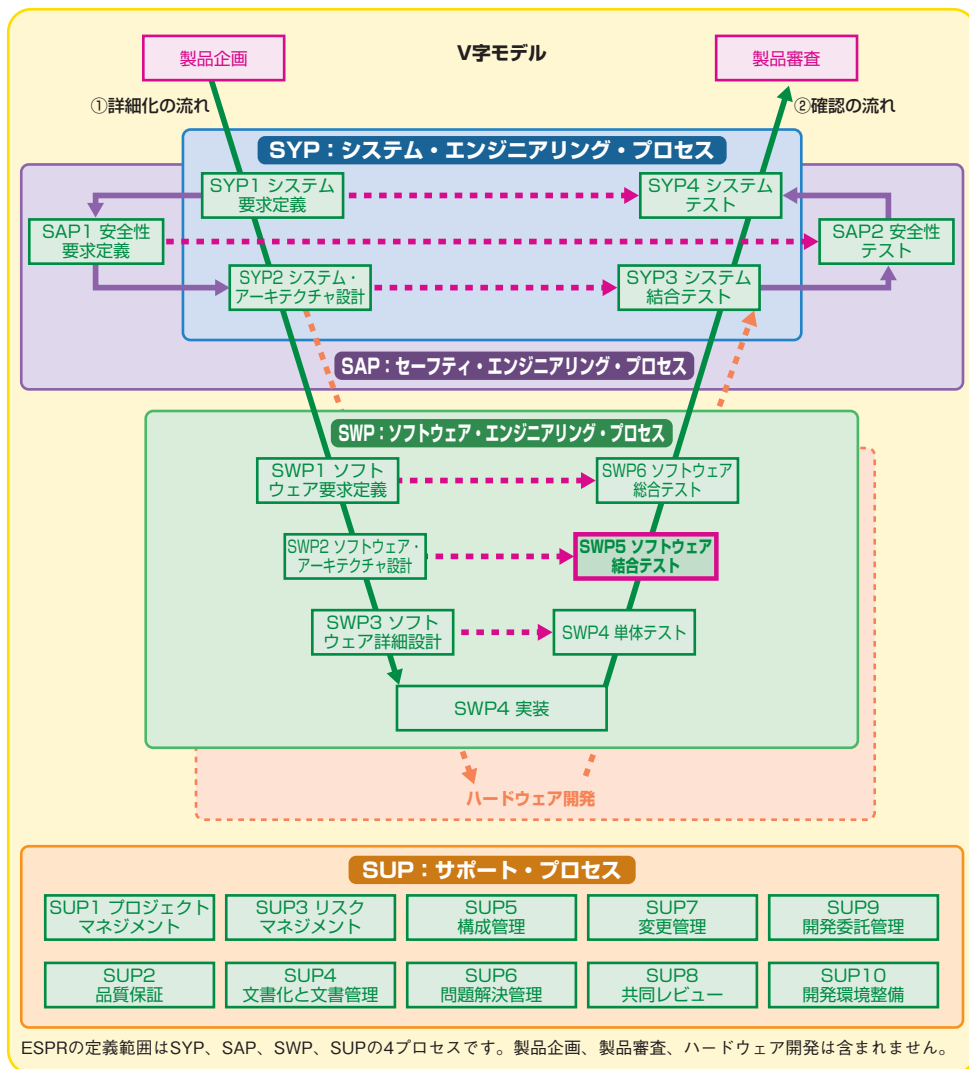


図2.12 V字モデルと開発プロセス (SWP5 ソフトウェア結合テスト)

■ SWP5.1 ソフトウェア結合テストの準備

ソフトウェア結合およびソフトウェア結合テストの準備をする。

5.1.1 ソフトウェア結合の準備

入力

(SW404) プログラムユニット
(SW205) ソフトウェア・アーキテクチャ設計書

概要

結合するプログラムユニットおよび結合環境を用意する。

出力

(SU1002) ソフトウェア開発環境 (make ファイルなど)

■ 実施内容

ソフトウェア結合の準備をする。

- ▶ 結合するプログラムユニットを用意する。
- ▶ ソフトウェアを結合するためのコンパイル、リンク環境などを準備 (make ファイルの作成など) をする。

■ 注意すべき事項

- 結合するプログラムユニットについては以下の点を確認しておく。
 - ▶ バージョン
 - ▶ 品質 (既知の不具合など)
- ▶ コンパイル、リンクの条件設定 (ローカル変数のレジスタへの割当、メモリ効率優先/実行速度優先など) は適切か。

バグカーブと信頼性

ソフトウェアの信頼性は、潜在するバグに関係します。しかし、すべてのバグを発見することは非常に困難です。

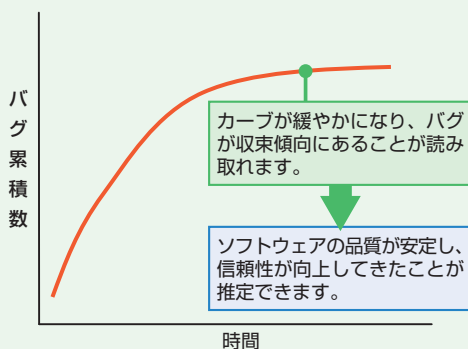
一般にテストを開始してから間もなくするとバグは多く発見されますが、徐々に減少し最後にはほとんど発見されなくなるという傾向があります。このことから、時間が経過するに従ってバグの累積個数が収束してくれば、そのソフトウェアの品質が安定してきたことが分かります。逆に、バグの累積個数が収束しない場合は、まだ潜在バグが残っており、品質が安定していないことが分かります。

これを視覚的に分かりやすくするために、縦軸にバグの累積個数、横軸にテストを行った時間をとって、その関係をグラフで表したのがバグカーブです(下グラフ)。バグカーブは信頼度成長曲線とも呼ばれ、テスト工程におけるバグが収束し、信頼性が向上してきたことを測るために利用することができます。信頼度成長曲線にはいくつか種類がありますが、よく知られている曲線に「ゴンペルツ(Gompertz)曲線」や「ロジスティック(logistic)曲線」があります。

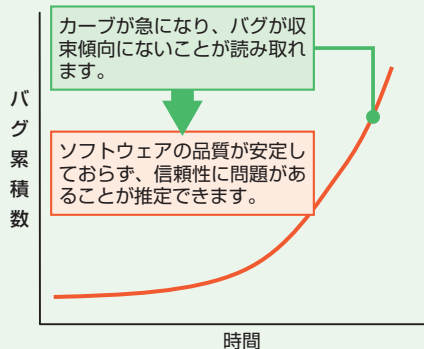
ここで注意しなければならないのは、テストの終わり近くになって発見されるバグの重要度です。たとえば、ユーザインタフェースのヘルプのチェックをテストの最後に行った場合に誤字や脱字といった問題が発見され、バグ累積数が一時的に増える場合もあります。たとえ、バグの累積数が増えても重要度の高いバグが発見されていなければ品質が安定していると判断することができます。しかし、仕様の変更を必要とするような重要度の高いバグが発見されているようであれば、まだ品質が安定しているとは判断できません。

バグカーブは、ソフトウェアの信頼性を測る上で利用することができます。また同時に、発見されているバグの重要度を見極めることも大切です。

①テスト後期になってバグ累積数が増えていない場合



②テスト後期になってバグ累積数が増えている場合



5.1.2 ソフトウェア結合テストの準備


入力


(SW205) ソフトウェア・アーキテクチャ設計書
(SU1002) ソフトウェア開発環境
(SU601) 不具合管理票
(修正確認の場合)

概要

ソフトウェア結合テストが実施可能な状態にする。

出力

(SW501) ソフトウェア結合テスト仕様書 
(SW502) ソフトウェア結合テストデータ
(SW503) 内部確認メモ
(ソフトウェア結合テスト仕様)

ドキュメント・テンプレート例が用意されています。

■ 実施内容

ソフトウェア結合テストを行うための、下記①～④を準備する。

①結合テスト項目を準備する。

- ▶ ソフトウェア・アーキテクチャ設計書で定義された機能が、正しく実現されているか否かを確認するためのテスト項目を準備し、テスト仕様書としてまとめる。テスト項目として準備すべき項目としては、以下のようなものがある。
 - ・ 機能テスト
 - ・ 条件網羅テスト
 - ・ データ境界値テスト
 - ・ プログラムユニット間および機能ユニット間のインタフェーステスト

②テストデータを作成する。

- ▶ ①で準備したテスト項目を実行するために必要となるテストデータ(具体的な入力データなど)を作成する。

③スタブ/テストドライバ* (擬似ソフトウェア)を作成する。

- ▶ 当該ソフトウェアを構成する一部のユニットから結合し動作確認する場合には、スタブ/テ

ストドライバなどを用意しコンパイル&リンクし動作可能な状態を作り上げる。

④テスト結果の判定基準や、テスト全体の評価基準や完了基準なども用意しておく。

⑤修正確認テスト項目を準備する(修正確認の場合)。

- ▶ 結合テストにおいて不具合が検出されて修正を行った場合には、その不具合が解消されたこと、また修正により別の不具合が発生していないことを確認するためのテスト項目を準備する。
- ▶ 不具合の内容により、テスト範囲を決定しテスト項目を選択する。

●用語解説

* **スタブ/テストドライバ**:スタブ (stub) とは、テスト対象部分 (プログラムユニット) から呼び出される下位モジュールの代替品のこと。テストドライバとは、テスト対象部分 (プログラムユニット) にテストデータを引き渡す上位モジュールの代替品のこと。

■ 注意すべき事項

- ソフトウェア結合テスト仕様書は確認しておく。この段階で確認すべき事項としては、以下のようなものがある。
 - ▶ ソフトウェア規模からみたテスト項目数の妥当性
 - ▶ 機能の組み合わせ
 - ▶ テスト項目の網羅性(ソフトウェア・アーキテクチャ設計書との対応)
 - ▶ テスト項目の矛盾/重複
 - ▶ ハードウェア制御部、割り込みルーチンのテスト方法(多重割込み等)
 - ▶ 高速化が必要な部分のテスト方法
 - ▶ ソフトウェア非機能要求の評価基準の妥当性(パフォーマンスなど)
 - ▶ エラー処理
- テスト項目に与える入力値については、システム動作にとっての特異点なども考慮しておく。
 - ▶ 正常データ
 - ▶ データ未設定(NULL値)
 - ▶ 境界値/仕様値を大きく超える値/条件
 - ▶ 最小値、最大値
- 修正確認テストの準備の考慮点
 - ▶ テストの実施範囲は、影響する部分をすべて網羅しているか。
 - ▶ テストは基本的に既に作成したテスト項目を再利用する。ただし、修正量が多い場合や、影響範囲が広い場合などは、新たなテスト項目を用意することも検討する。(グローバル変数/共有メモリの使用範囲等に留意する)
- 文書化での留意点
 - ▶ 変更履歴を添付し、変更箇所を明示する。
 - ▶ 作成文書に関しての責任所在を明示する。
 - ▶ 作成文書は構成管理および変更管理にて確実に管理する。関連プロセス：SUP5 構成管理、SUP7 変更管理

■ SWP5.2 ソフトウェア結合テストの実施

ソフトウェア結合およびソフトウェア結合テストを実施する。

5.2.1 ソフトウェア結合

入力

(SW404) プログラムユニット
(SW205) ソフトウェア・アーキテクチャ設計書
(SU1002) ソフトウェア開発環境

概要

プログラムユニットを結合する。

出力

(SW504) 機能ユニット

■ 実施内容

コンパイルとリンクを実行して、プログラムユニットを結合する。

■ 注意すべき事項

- ソフトウェア結合の考慮点
 - ▶ 結合するプログラムユニットのバージョンは適切であることを確認する。

5.2.2 ソフトウェア結合テストの実施

入力

(SU1002) ソフトウェア開発環境
(SW504) 機能ユニット
(SW501) ソフトウェア結合テスト仕様書
(SW502) ソフトウェア結合テストデータ
(SU601) 不具合管理票
(修正確認の場合)

概要

ソフトウェア結合テスト仕様書に従って、結合テストを実施する。

出力

(SW505) ソフトウェア結合テスト結果

■ 実施内容

- ①ソフトウェア結合テストを実施する。
 - ▶ ソフトウェア結合テスト仕様書(SW501)をもとに、ソフトウェア結合テストを実施し、出力結果を得る。
 - ▶ 代替テストを実施した場合は、理由を明記した上で記録を残す。
 - ▶ 不具合検出時、継続して残りのテストを実施するか、不具合が修正されるまでペンディングするか判断をする。
 - ▶ 出力結果を収集する(各種ログなど)。
 - ▶ 準備したテスト項目が実施できなかった場合は、理由を明記した上で記録を残す。また、その理由の妥当性について判断する。
- ②修正確認テストを実施する。
 - ▶ 不具合が解消されたかテストを実施する。
 - ▶ 不具合の修正により、別の不具合が発生していないかテストを実施する。

■ 注意すべき事項

- 修正確認テストでは修正後の最新版でテストを実施しているかを確認する。

5.2.3 ソフトウェア結合テスト結果の確認



入力


(SW501) ソフトウェア結合テスト仕様書
(SW505) ソフトウェア結合テスト結果
(SU601) 不具合管理票
(修正確認の場合)

概要

テスト結果を確認し、テストの合否を判定する。

出力

(SW506) ソフトウェア結合テスト報告書 
(SU601) 不具合管理票 

 ドキュメント・テンプレート例が用意されています。

■ 実施内容

①ソフトウェア結合テスト結果を確認する。

- ▶ テスト結果を確認して、実施したテスト項目の合否を判定する。
- ▶ 修正確認テストの結果、不具合が解消された場合は、修正確認が済んだことを不具合管理票に記録する。

②不具合記録を作成する。

- ▶ 不具合を発見した場合、不具合管理票に不具合内容を記録する。

関連プロセス：SUP6 問題解決管理

■ 注意すべき事項

- テスト結果に関しては、ソフトウェア結合テスト仕様書の誤りの可能性についても考慮する。

■ SWP5.3 ソフトウェア結合テスト結果の確認

結合されたソフトウェアがソフトウェア・アーキテクチャ設計で定義された処理を正しく実現されているかテスト結果を確認する。

5.3.1 ソフトウェア結合テスト結果の内部確認


入力


(SW205) ソフトウェア・アーキテクチャ設計書
(SW501) ソフトウェア結合テスト仕様書
(SW503) 内部確認メモ (ソフトウェア結合テスト仕様)
(SW506) ソフトウェア結合テスト報告書
(SU601) 不具合管理票

概要

ソフトウェア結合テストで未解決の問題や未実施のテスト項目がないか確認し、内部確認レポートとしてまとめる。

出力

(SW507) 内部確認レポート (ソフトウェア結合テスト) 

 ドキュメント・テンプレート例が用意されています。

■ 実施内容

- ① 下記の視点でソフトウェア結合テストの結果を確認する。
 - ▶ 未解決となっている問題がないか。未解決となっている問題がある場合は、問題の内容を確認して早急に対応すべきか、次テストフェーズに持ち越すべきかを判断する。
関連プロセス：SUP8 共同レビュー、SUP1 プロジェクトマネジメント
 - ▶ 未実施となっているテスト項目がないか。未実施となっているテスト項目がある場合は、未実施となった理由を確認して対応を検討する。
関連プロセス：SUP6 問題解決管理
- ② 確認結果は内部確認レポート (SW408) として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

■ 注意すべき事項

- 不具合検出数は、品質基準を満たしているか確認する。
- 修正確認テストの場合は、テスト範囲と環境は適切であったかを確認する。
- 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション (共同レビューの開催など) などあわせて記載しておくことが望ましい。

Related Standards

ISO/IEC 12207 Information technology – Software life cycle processes JIS X 0160 ソフトウェアライフサイクルプロセス

成立・改訂年： ISO/IEC 1995、AMD.1 2002、AMD.2 2004、JIS 1996

目的： ソフトウェアのライフサイクルプロセス群の概念の標準化

対象分野： ソフトウェア開発プロセス

URL： <http://www.iso.org/>

国内入手先： 日本規格協会

概要：

ソフトウェア製品開発のプロセスを体系的に整理した規格であり、プロセスに関する規格としては基礎となるものです。

この規格は、情報技術分野のみでなく、広く産業用システムにソフトウェアの利用が普及する中で、ソフトウェア開発に関する作業を整理し、その名称などを統一することを目的に制定されました。

この規格の枠組みは、ソフトウェアの開発を構想する段階から、そのソフトウェアの利用を停止する段階に至るまでのライフサイクルに従って整理され、プロセスの管理および改善について規定しています。

この規格では、プロセス、アクティビティ、タスクという階層により、プロセス概念を定義しています。本ガイドでは、その階層概念を踏襲しながら、組込みソフトウェア開発に即して、具体的にプロセスを整理しています。

プロセス記述

－ [プロセス名称]

－ アクティビティ一覧

[アクティビティ 1]

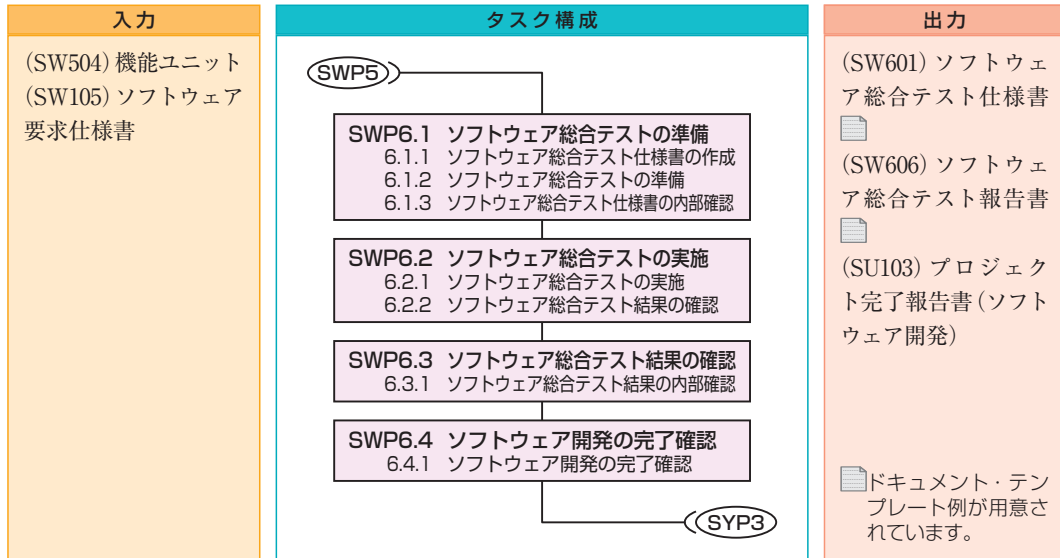
－ タスク 1

－ タスク 2

－ タスク 3

SWP6 ソフトウェア総合テスト

ソフトウェアを構成する個々の要素(機能ユニット)をすべて結合した状態で、ソフトウェアとしての総合的なテストを実施する。



■ 解説

このアクティビティは

- (6.1.1) (6.1.2) (6.1.3) 結合が完了したソフトウェアが、ソフトウェア要求仕様に記載された機能動作を実現しているかどうかを確認するためのテスト仕様を準備し、
- (6.2.1) (6.2.2) 実際にそのテスト仕様に従い、ソフトウェアを動作させ、ソフトウェアとしての総合的なテストを行う。
- (6.3.1) ソフトウェア総合テスト結果については、テストの合否判定基準に従って確認を行い、確認した内容を整理して内部確認レポートを作成する。
- (6.4.1) また、関係者を交えてソフトウェア総合テスト結果の共同レビューを実施してソフトウェア開発の最終確認を行い、レビューの結果等を整理してソフトウェア開発完了報告書を作成する。

ソフトウェア総合テストは、ソフトウェアの開発において、基本的に最終的な確認の作業と位置づけることができる。このため、この作業は製品としての実フィールドの運用を想定して、漏れがないように実施されなければならない。

■ 留意事項

- ▶ ソフトウェア総合テストでは、テストの際の動作環境(テスト環境)に注意する。実際にソフトウェアが動作する環境を想定したテストを実施する。
- ▶ ソフトウェア開発における最終確認のための作業として、テスト項目の漏れ、テストで検出

された不具合に関するリグレッションテストなどにも注意する。

- ▶ テストで検出された不具合は、不具合管理の手順・ルールなどに従い、その検出・対応などの状況を適切に管理することが必要である。

■ 【参考】手法およびツール

- ▶ システム動作に関するシミュレータ装置など
- ▶ 計測器(ロジックアナライザ、オシロスコープなど)
- ▶ 自動テストツール(リグレッションテスト自動化

など)

- ▶ バグ管理ツール
- ▶ 信頼度成長曲線

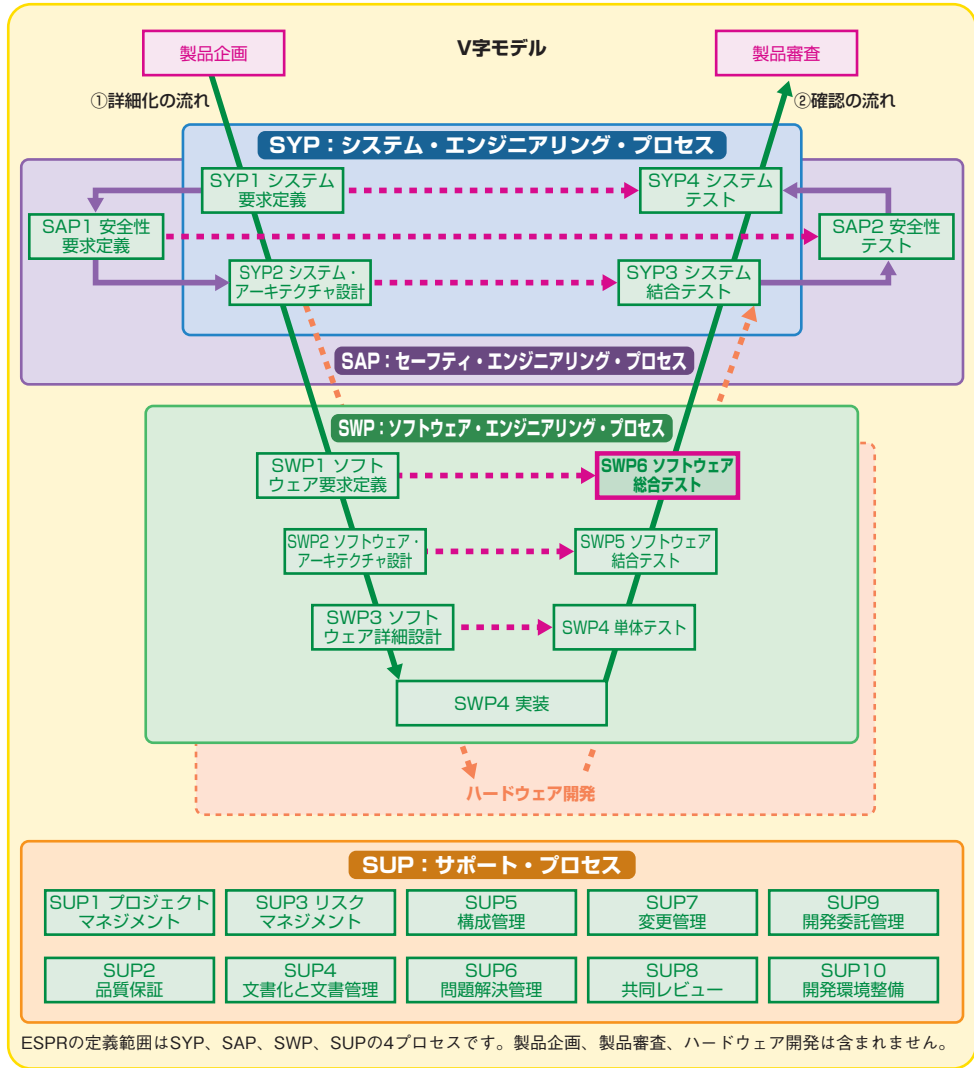


図2.13 V字モデルと開発プロセス (SWP6 ソフトウェア総合テスト)

■ SWP6.1 ソフトウェア総合テストの準備

ソフトウェア総合テストの準備をする。

6.1.1 ソフトウェア総合テスト仕様書の作成


入力


(SW105) ソフトウェア要求仕様書
テスト仕様書作成に必要な情報 (成果物)

概要

ソフトウェア総合テスト仕様書を作成する。

出力

(SW601) ソフトウェア総合テスト仕様書 

 ドキュメント・テンプレート例が用意されています。

参照情報

過去のソフトウェア総合テスト仕様書

■ 実施内容

ソフトウェア要求仕様書をもとにソフトウェア総合テスト仕様書を作成する。

- ▶ ソフトウェア要求仕様書に書かれた内容が正しく実現されているか確認するためのテスト項目を準備する。テスト項目としては、ソフトウェアの機能要求、非機能要求などを網羅する形で準備する。
- ▶ 個々のテスト項目に関する合否判定の基準を明確にしておく。

■ 注意すべき事項

- ▶ 過去のソフトウェア総合テスト仕様書の再利用について検討する。
- ▶ 過去の不具合事例を考慮しているか確認する。
- ▶ シリーズ製品の場合、新規機能のテスト項目が十分であるか。また、変更機能の影響把握はできているか確認する。
- ▶ 状態遷移表、マトリックス網羅法等を用い、漏れのないテスト項目を抽出する。
- ▶ テスト手順や完了条件が、法令、規格等に準拠しているか確認する。
- ▶ 想定される出力結果が明確になっているか確認する。
- ▶ 実際のエンドユーザの使用状況を考慮したテスト項目を作成する。
- ▶ 正常系および異常/例外系のテスト項目を作成する。
- ▶ 他システムとの整合やハードウェアとの整合を考慮する。
- テスト仕様書は、設計フェーズなど上流プロセスの中で検討、作成しておくが良い。
- 文書化での留意点
 - ▶ 変更履歴を添付し、変更箇所を明示する。
 - ▶ 作成文書についての責任所在を明示する。
 - ▶ 作成文書は構成管理および変更管理にて確実に管理する。

関連プロセス：SUP5 構成管理、SUP7 変更管理

6.1.2 ソフトウェア総合テストの準備

入力

(SW601) ソフトウェア総合テスト仕様書
(SW504) 実行可能形式化された最終のソースコード
(SU601) 不具合管理票
(修正確認の場合)

概要

ソフトウェア総合テストを実施するために必要なテストデータやテスト環境を準備する。

出力

(SW602) 実機
(SW603) ソフトウェア総合テストデータ

■ 実施内容

総合テストの実施に必要なテストデータ、テスト環境などを準備する。

- ① テストデータを作成する。
 - ▶ 6.1.1 で準備したテスト項目を実行するために必要となるテストデータ (具体的な入力データなど) を作成する。
- ② テスト環境を準備する。
 - ▶ ソフトウェア総合テストに必要なテスト環境を準備する。
 - ・ テスト治具 (通信ターゲット等) を作成する
 - ・ シミュレーションソフトを作成する
 - ▶ ソースコードを実行可能形式に変換し、半導体メモリに書き込む。
- ③ テスト結果の判定基準や、テスト全体の評価基準や完了基準なども用意しておく。
- ④ 修正確認テスト項目を準備する (修正確認の場合)。
 - ▶ メモリや回路などを実装し、実環境で動作させるための実機 (あるいは試作機) を調達する。
 - ▶ 不具合が解消されたこと、また修正により別の不具合が発生していないことを確認するためのテスト項目を準備する。
 - ・ 不具合の内容により、テスト範囲を決定しテスト項目を選択する

■ 注意すべき事項

- ▶ ソフトウェアとしての全体動作に関して、組込みソフトウェアの場合、特に異常処理関連などの事象を確認するための、テストデータのバリエーションに留意する。
- ▶ テスト治具を用意する場合、その精度は許容範囲内であるかを確認する。
- ▶ ハードウェアが正常に動作する環境 (ノイズ、電源供給等) であるかを確認する。
- ▶ テストするバージョンが正しいバージョンであるか確認する。
- ▶ テスト環境に関しては不具合発生時の原因分析などを考慮し、テストの再現性のための工夫をしておく。
- ▶ 修正確認テストの準備の考慮点
 - ・ テストの実施範囲は、影響する部分をすべて網羅しているか
 - ・ テストは基本的に既に作成したテスト項目を再利用する。ただし、修正量が多い場合や、影響範囲が広い場合などは、新たなテスト項目を用意することも検討する

6.1.3 ソフトウェア総合テスト仕様書の内部確認

入力

(SW105) ソフトウェア要求仕様書
(SW601) ソフトウェア総合テスト仕様書

概要

ソフトウェア総合テスト仕様書を確認する。

出力

(SW604) 内部確認メモ
(ソフトウェア総合テスト仕様)

■ 実施内容

作成したソフトウェア総合テスト仕様書を確認する。

- ▶ ソフトウェア総合テスト仕様書(SW601)に記載されたテスト項目が、ソフトウェア要求仕様書(SW105)に定義された機能要求、非機能

要求を検証する項目となっているかを開発グループ内で確認する。

■ 注意すべき事項

- 確認すべき事項としては、以下のようなものがある。
 - ▶ ソフトウェア規模からみたテスト項目数の妥当性
 - ▶ テスト項目の網羅性(ソフトウェア要求仕様書との対応)
 - ▶ テスト項目の矛盾/重複
 - ▶ ソフトウェア非機能要求の評価基準の妥当性(パフォーマンスなど)

- ▶ エラー処理
- ▶ フェールセーフ処理
- ▶ 期待値

- 開発者がテスト項目を作成している場合は、特にテスト項目の抽出やテスト合否の判定基準などが甘くないか(不十分でないか)注意する。

■ SWP6.2 ソフトウェア総合テストの実施

ソフトウェア総合テストを実施する。

6.2.1 ソフトウェア総合テストの実施

入力

(SW601) ソフトウェア総合テスト仕様書
(SW603) ソフトウェア総合テストデータ
(SW602) 実機
(SU1002) ソフトウェア開発環境

概要

ソフトウェア総合テスト仕様書に従って、テストを実施する。

出力

(SW605) ソフトウェア総合テスト結果

■ 実施内容

ソフトウェア総合テスト仕様書に従いテストを実施する。

①ソフトウェア総合テストを実施する。

- ▶ ソフトウェア総合テスト仕様書(SW601)をもとに、ソフトウェア総合テストを実施し、出力結果を得る。
- ▶ 代替テストを実施した場合は、理由を明記した上で記録を残す。
- ▶ 不具合検出時、継続して残りのテストを実施するか、不具合が修正されるまでペンディングするかを判断をする。
- ▶ 出力結果を収集する(各種ログなど)。
- ▶ 準備したテスト項目が実施できなかった場合は、理由を明記した上で記録を残す。また、その理由の妥当性について判断する。

②修正確認テストを実施する。

- ▶ 不具合が発生し、それを修正した場合の確認として、不具合が解消されたかテストを実施する。
- ▶ 不具合の修正により、別の不具合が発生していないかテストを実施する。

■ 注意すべき事項

- 修正確認テストでは、修正後の最新版でテストしているかどうかなどを確認する。

6.2.2 ソフトウェア総合テスト結果の確認



入力


(SW601) ソフトウェア総合テスト仕様書
(SW504) 実行可能形式化された最終のソースコード
(SW605) ソフトウェア総合テスト結果
(SU601) 不具合管理票 (修正確認の場合)

概要

テスト結果を確認し、テストの合否を判定する。

出力

(SW606) ソフトウェア総合テスト報告書 
(SU601) 不具合管理票 

 ドキュメント・テンプレート例が用意されています。

■ 実施内容

ソフトウェア総合テストの結果を確認し、実施したテスト項目の合否を判定する。

- ▶ 不具合を発見した場合、不具合管理票に不具合内容を記録する。
- ▶ 修正確認テストの結果、不具合が解消された場合は、修正確認が済んだことを不具合管理票に記録する。

関連プロセス：SUP6 問題解決管理

■ 注意すべき事項

- テスト結果に関しては、ソフトウェア総合テスト仕様書の誤りの可能性についても考慮する。
- 結果の合否判定については、ソフトウェア総合テスト仕様書に記載された判定基準に基づいて行う。

■ SWP6.3 ソフトウェア総合テスト結果の確認

ソフトウェア要求定義で定義された要求事項を正しく実現されているかテスト結果を確認する。

6.3.1 ソフトウェア総合テスト結果の内部確認


入力


(SW105) ソフトウェア要求仕様書
(SW601) ソフトウェア総合テスト仕様書
(SW604) 内部確認メモ (ソフトウェア総合テスト仕様)
(SW606) ソフトウェア総合テスト報告書
(SU601) 不具合管理票

概要

ソフトウェア総合テストで未解決の問題や未実施のテスト項目がないか確認し、内部確認レポートとしてまとめる。

出力

(SW607) 内部確認レポート (ソフトウェア総合テスト) 

 ドキュメント・テンプレート例が用意されています。

■ 実施内容

① 下記の視点でソフトウェア総合テスト結果を確認する。

- ▶ 未解決となっている問題がないか。未解決となっている問題がある場合は、問題の内容を確認して早急に対応すべきか、次テストフェーズに持ち越すべきかを判断する。

関連プロセス：SUP8 共同レビュー、SUP1 プロジェクトマネジメント

- ▶ 未実施となっているテスト項目がないか。未実施となっているテスト項目がある場合は、未実施となった理由を確認して対応を検討する。

関連プロセス：SUP6 問題解決管理

② 確認結果は内部確認レポート (SW607) として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

■ 注意すべき事項

● 不具合検出数は、品質基準を満たしているかを確認する。

- ▶ 修正確認テストの場合は、テスト範囲と環境は適切であったか。

● 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション (共同レビューの開催など) などともあわせて記載しておくことが望ましい。

SWP6.4 ソフトウェア開発の完了確認

ソフトウェア要求定義で定義された要求事項を正しく実現していることをステークホルダ間で評価し、ソフトウェア開発の最終確認を行う。

6.4.1 ソフトウェア開発の完了確認

入力

(SW105) ソフトウェア要求仕様書
(SW601) ソフトウェア総合テスト仕様書
(SW606) ソフトウェア総合テスト報告書
(SW607) 内部確認レポート(ソフトウェア総合テスト)
(SU101) プロジェクト計画書
(SU601) 不具合管理票

概要

ソフトウェア総合テストの結果報告書をもとに共同レビューを実施し、ソフトウェア開発部門としての最終的な合否の判定をするとともに、製品システム部門、製品審査に必要な情報を整理する。

出力

(SU801) 共同レビュー記録(ソフトウェア総合テスト)
(SU103) プロジェクト完了報告書(ソフトウェア開発)

📄ドキュメント・テンプレート例が用意されています。

■ 実施内容

以下の手順で総合テストの結果に関する共同レビューを実施し、完了報告書を作成する。

- ①ソフトウェア開発に関係するステークホルダ(企画部門担当者、ソフトウェア開発者、ハードウェア開発者、システム評価者、製造担当者など)を集めソフトウェアの最終的な合否判定をする。レビューでは特に下記の点を考慮する。
 - ▶ソフトウェアとして想定される利用者、利用環境下で
 - ・期待される機能要求が確実に保証されていることを確認する
 - ・求められる非機能要求が確実に保証されていることを確認する
 - ▶ソフトウェア開発に関しての残件がないことを確認する。
 - ▶ソフトウェア総合テストにおける不具合検出数、修正数、重大不具合の検出数、修正数などの指標を用いて品質面での問題がないことを確認する。
- ▶ソフトウェア総合テスト以前に実施されたレビューなどにおけるすべての指摘事項について、適切な対応がとられていることを確認する。
関連プロセス：SUP2 品質保証、SUP6 問題解決管理、SUP8 共同レビュー
- ②レビューの記録をもとに、完了報告書を作成する。
完了報告書の作成については「SUP1.4 プロジェクト完了報告書の作成」を参照のこと。
完了報告書は、ソフトウェア開発関係者のみだけでなく、ハードウェア開発関係者やシステム全体のとりまとめ関係者にも配布する。
関連プロセス：SUP1 プロジェクトマネジメント

■ 注意すべき事項

- ▶ レビューでは開発について、定量的側面、定性的側面の両面から開発過程の妥当性とその結果として作成されたプロダクトとしてのソフトウェアの妥当性を確認する。
- ▶ ソフトウェア開発における最終段階でのレビューとなるため、その開発の責任を負う担当者、管理者がレビュー結果の判定を行う。

SAP：セーフティ・エンジニアリング・プロセス

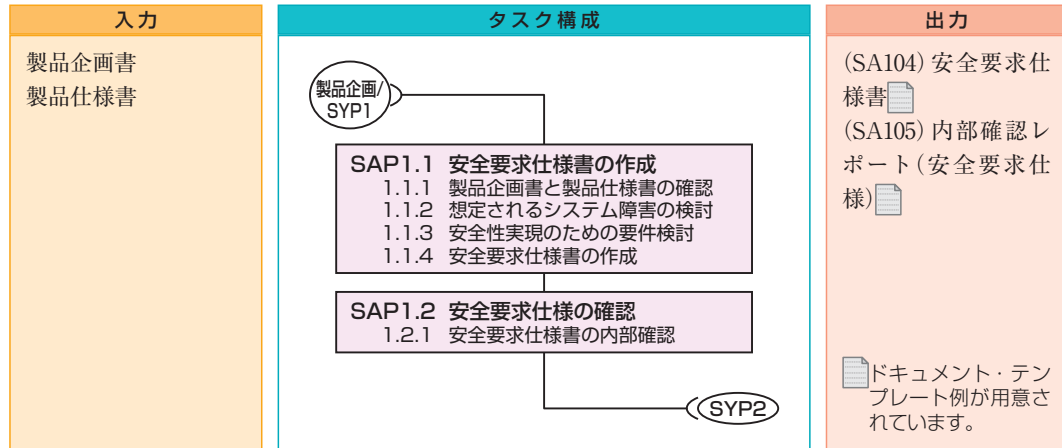
セーフティ・エンジニアリング・プロセスは、製品として想定されるあらゆる利用形態を考慮し、その状況下で製品に求められる安全性を定義し、それが確実に実現されていることを確認するための作業を含んでいます。

このプロセスには、下記のアクティビティが含まれています。

| ID | アクティビティ | アクティビティの概要 | 構成するタスク |
|------|---------|---|--|
| SAP1 | 安全性要求定義 | 当該製品に関して、安全性に関する要求事項を洗い出し、安全要求仕様書に整理する。 | SAP1.1 安全要求仕様書の作成 SAP1.2 安全要求仕様の確認 |
| SAP2 | 安全性テスト | 開発した製品に関して、安全性の視点からのテストを実施する。 | SAP2.1 安全性テストの準備 SAP2.2 安全性テストの実施 SAP2.3 安全性テスト結果の確認 |

SAP1 安全性要求定義

当該製品に求められる安全性に関する要求を明確にする。



■ 解説

このアクティビティは

- (1.1.1) 製品企画書と製品仕様書をもとに、製品として実現・提供する機能やサービスを把握した上で、対象システムが利用される状況やそのユーザなどを明確にし、
- (1.1.2) 想定されるシステム障害を検討する。
- (1.1.3) 対象システムに求められる安全性を実現するためのシステム要件(安全機能)などを検討し、機能ごとに安全水準を決定する。
- (1.1.4) これらの検討結果を整理して、安全要求仕様書を作成する。
- (1.2.1) 作成した安全要求仕様書はあらかじめ確認ポイントを決めて確認を行い、確認した内容を整理して内部確認レポートを作成する。

■ 留意事項

- ▶ 安全性要求定義の開始条件として以下の事項に留意する。
 - ・ 製品企画：製品戦略(エンドユーザニーズなど)が明確になっている
 - ・ 製品システムの利用される状況(コンテキスト)などが明確になっている
- ▶ 当該システムに関する安全性が損なわれたときの影響の大きさや頻度について、おおよそのところを把握しておく。
- ▶ 当該システムに関する安全性欠損による影響範囲や影響期間などについてもおおよそのところを把握しておく。
- ▶ 安全性要求定義では当該システムの安全性実現における責任所在や安全性を担保するための仕組みなどについても検討しておく。

■【参考】手法およびツール

- ▶ ハザード解析技術
 - ・ FMEA (Failure Modes and Effects Analysis : 故障モード影響解析)
 - ・ FTA (Fault Tree Analysis : 故障木解析)

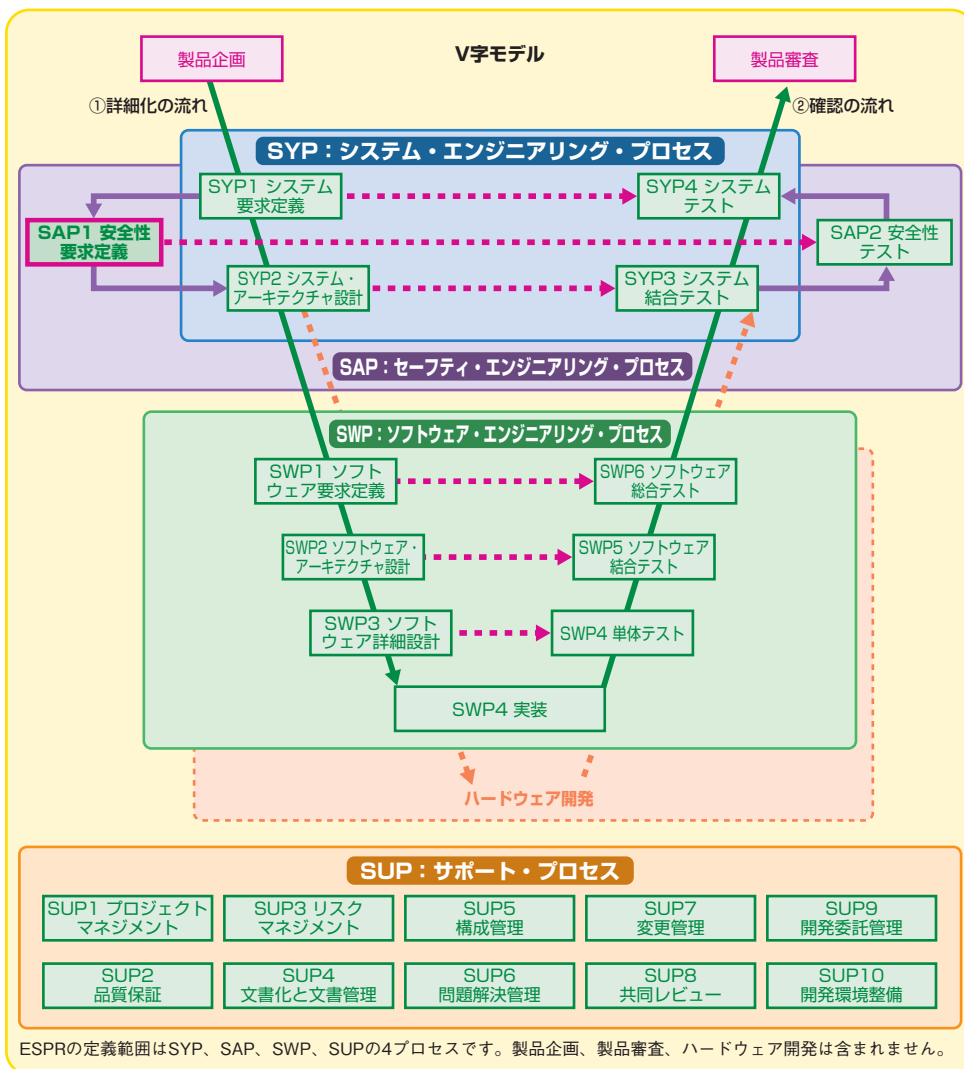


図2.14 V字モデルと開発プロセス (SAP1 安全性要求定義)

SAP1.1 安全要求仕様書の作成

製品企画書および製品仕様書をもとにシステムとして実現が求められる安全性側面からの要求項目を明確にし、安全要求仕様書としてまとめる。

1.1.1 製品企画書と製品仕様書の確認

入力

製品企画書
製品仕様書

概要

製品企画部門などで作成した製品企画書と製品仕様書に記載された内容についてシステム安全性の視点から確認する。

出力

(製品企画書・仕様書確認メモ)

■ 実施内容

- ①製品企画書に記載されている以下の項目について確認する。
 - ▶ 製品の概要や特徴、既存製品からの機能面での差分など
 - ▶ 製品が関係する規格、規約、法律、法規
- ②製品仕様書に記載されている以下の項目について確認する。
 - ▶ 製品のビジョンやコンセプト
 - ▶ 対象ユーザとシステム要求
 - ▶ 製品の利用シーンと利用コンテキスト
 - ▶ 製品実現における制約事項

■ 注意すべき事項

- 製品企画書の確認で注意すべき事項としては以下のようなものがある。
 - ▶ 製品企画書は製品企画部門や営業部門などによって作成される製品に関する概要情報を整理したドキュメントであり、システムや製品の安全性に関する側面について必ずしも十分にあるいは適切に盛り込まれているとは限らないことに注意する。
 - ▶ 製品企画書に記載された事項は、対象ユーザや市場動向などさまざまな要因によって変更される場合もあるため、製品に関する安全要求についても、1つの製品でもさまざまな側面を有することに注意する。
 - 製品仕様書の確認で注意すべき事項としては以下のようなものがある。
 - ▶ 製品のビジョンやコンセプト
 - ・ 製品がどのような機能やサービスを提供するか
 - ・ それらの安全性が損なわれたときに、システムやその周囲にどのような影響が及ぶか
 - ▶ 対象ユーザとシステム要求
 - ・ どのようなユーザがシステムを利用するか。またシステムの一次ユーザや二次三次のユーザとして、どのようなユーザが関係してくるか
 - ・ システムの直接的なユーザではなくても、システムの安全性が損なわれた場合に、どのような影響が及ぶかについても分析しておく
 - ▶ 製品の利用シーンと利用コンテキスト
 - ・ 対象とするユーザが製品を利用する状況
 - 1) 通常の利用に関する状況
 - 2) 想定外の状況での利用
- なども含めて、システム利用の全てのコンテキストに対応した、全ての動作モードを洗い出し、それらに関しての安全性の面からの検討

を加えておく

- ・ 製品の開発段階での安全性に対する責任所在や製品リリース後の安全性に対する責任所在なども明確にしておく
- ▶ 製品実現における制約事項
- ・ 製品を構成する全ての要素(ソフトウェア、ハードウェアなど)を洗い出し、それらの間の関係や制約を整理する
 - ・ システムを利用する上で前提として利用されるシステムやハードウェア(周辺システムやハードウェア)を洗い出し、それらによる動作モードや機能への制約も整理しておく

1.1.2 想定されるシステム障害の検討

入力

製品企画書
製品仕様書

概要

当該製品やシステムに関して発生する可能性のある不具合やトラブルの発生度合いと、それらが発生した場合の影響度合いを検討する。

出力

(SA101) 想定システム障害リスト

■ 実施内容

- ①当該製品やシステムに関して発生する可能性のある不具合やトラブルをリストアップし、それらが発生する度合いを検討する。
- ②上記の不具合やトラブルが発生した場合に、どのような影響がユーザやその周囲に及ぶか、その影響度合いを検討する。

■ 注意すべき事項

- ▶ 通常システムではこれらの不具合やトラブルを未然に防ぐことやその影響を軽減することを目的とする安全機能が考えられているが、それらが、正常に機能しない場合なども考慮しておく。
- ▶ 不具合やトラブルによる安全性への影響としては、人命に関わる事故、傷害事故や経済損、社会的な混乱などさまざまな影響を広範に検討しておく。
- ▶ 不具合やトラブルを引き起こす要因としては、単に組み込みソフトウェアの不具合のみだけでなく、周辺ハードウェアの故障などがきっかけになる場合もあるため、システム構成要素や関連要素をくまなく検討して、不具合発生頻度などを見積もること。
- ▶ システムの運用や利用の段階で安全性が損なわれるケースとしては、システムや製品の利用者側のミスや、それ以外の外部からのさまざまな要因(外乱)などによるケースもあるため、これらへの対処も含めてシステム安全性が阻害される可能性とその影響を考慮しておく。
- ▶ 特にハードウェア起因でソフトウェアに影響を及ぼしシステム動作を不安全にするようなトラブルにおいては、ハードウェア故障自体は確率論的な視点から起きることを考慮しておく。

1.1.3 安全性実現のための要件検討

入力

製品企画書
製品仕様書
(SA101) 想定システム障害リスト

概要

当該製品やシステムに求められる安全性を実現するために必要となる要件(安全機能)を検討し、機能ごとに安全度水準を決定する。

出力

(SA102) システム安全要求リスト
(SA103) システム安全度水準

■ 実施内容

- ① 製品企画書、製品仕様書、想定システム障害リスト(SA101)をもとに、製品に求められる安全性を考慮して、製品の安全性を実現するためにシステムに求められる安全機能の要求を洗い出す。
- ② 安全機能はソフトウェア単独ではなく、関連するハードウェア動作なども考慮してシステム全体としての安全が保たれるように考える。特に、
 - ▶ ソフトウェアの自己監視機能
 - ▶ ハードウェアやセンサ、アクチュエータの監視機能
 などには注意を払って、システムやソフトウェアのアーキテクチャに対する安全要求を整理する。
- ③ システムの動作モード(始動、自動、手動、半自動、定常・非定常など)に対応した安全機能を洗い出し整理する。
- ④ システムの安全メカニズムとしてロジックの二重化やフェールセーフ、フルブーフなどの機構の実現も検討する。
- ⑤ システムに対する外乱や入力データの誤差、あるいは、ユーザの操作ミスなどの可能性も含めて、これらの影響を最小化し、システムの安全性を維持するためのメカニズムを検討する。
- ⑥ システム安全要求については、単純にシステムとして実装する安全機能以外にも、システムの利用や運用の際の方式面(システム異常時のシステム管理者やユーザの対応など)での安全性確保などについても検討しておく。
- ⑦ 安全な製品・システムを開発するために、利用する開発手法やツールなどについても合わせて整理しておく。特に高度な安全性が求められる場合には、開発で使用するコンパイラやツールなどを明らかにしておく。
- ⑧ 不具合やトラブルの発生頻度とそれらによる影響度合いを考慮して、安全機能がどの程度機能することを保証するか検討し、安全機能ごとに安全性の水準(安全度水準)を決定する。

■ 注意すべき事項

- ▶ システムに求められる安全性が達成され得るように、必要な安全機能については十分に詳細にその仕様を検討し整理する。
- ▶ 安全機能に関する要求事項は明確かつ厳密に定義し、機能安全評価や安全性テストの局面で、検証や試験が可能であるように配慮しておく。
- ▶ ソフトウェア、ハードウェア間の安全に関する制約なども全てリストアップしておく。
- ▶ 開発で使用するコンパイラやツールを検討する際には、信頼性や実績を考慮する。
- ▶ システムの安全度の度合い(レベル)については、IEC61508などで提唱されている安全度水準*の考え方なども参考にする。

● 用語解説

* 安全度水準：IEC61508 では、システムに求められる目標とする安全機能が、どの程度の割合で機能不全にいたるかの許容程度を、安全度水準として管理することが求められている。

1.1.4 安全要求仕様書の作成


入力


製品企画書
製品仕様書
(SA101) 想定システム障害リスト
(SA102) システム安全要求リスト
(SA103) システム安全度水準

概要

製品やシステムの安全性に関する要求事項を整理し安全要求仕様書を作成する。

出力

(SA104) 安全要求仕様書 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

① (SA101) ~ (SA103) の事項を考慮し、安全要求仕様書を作成する。

- ▶ 上記検討の段階で幾つかの代替案を並行して検討してきた場合には、最終案を選択決定する必要がある。
- ▶ システムの安全設計指針やシステム利用・運用時の安全性担保の方針も合わせて整理する。

▶ 安全要求仕様書にはソフトウェア安全機能に関する要求を含めて製品に求められる安全性 (安全機能と安全度水準) を明示しておく。

■ 注意すべき事項

- ▶ 安全要求仕様書では
 - ・システムのトラブルなどによって安全性が損なわれた場合の影響
 - ・これらを未然に防ぐためのシステムとしてのメカニズム
 - ・システム運用や利用の際に安全性を担保するための仕組み
 - ・システムに求められる安全度の水準なども明確になっていることが望ましい。
- ▶ 必要に応じてユースケース分析の結果 (ユースケース図、ユースシナリオなど) も加えておくと良い。
- ▶ システムの安全設計指針としては、
 - ・設計に関する安全性の面からの制約条件
 - ・安全な設計を行うための設計手法

- ・システムの安全を確実なものとするためのシステムの動作プラットフォーム
- ・既存システムの再利用による安全性の担保なども記載しておくといよい。
- ▶ 安全要求仕様書の段階で、未確定要因がある場合には、それを明記しておく。
- ▶ 文書化での留意点
 - ・変更履歴を添付し、変更箇所なども明示する
 - ・作成文書に関しての責任所在を明示する
 - ・作成文書は構成管理および変更管理にて確実に管理する

関連プロセス：SUP5 構成管理、SUP7 変更管理

■ SAP1.2 安全要求仕様の確認

定義したシステム安全要求事項が製品としての安全性要求を満たしていることを確認する。

1.2.1 安全要求仕様書の内部確認


入力


(SA104) 安全要求仕様書

概要

安全要求仕様がシステムとして求められる事項を満たしているか確認し、内部確認レポートとしてまとめる。

出力

(SA105) 内部確認レポート (安全要求仕様) 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

① 下記の視点で、安全要求仕様書 (SA104) の内部確認を行う。

- ▶ 当該製品やシステムに関して不具合やトラブルが発生した場合に、どのような影響がユーザーやその周囲に及ぶか、その影響度合いを検討しているか。
- ▶ 不具合やトラブルの発生頻度と、影響度合いを考慮して、システムに求められる安全性の水準 (安全度水準) を決定しているか。
- ▶ システムに対する外乱や入力データの誤差、あるいは、ユーザーの操作ミスなどの可能性も含めて、これらの影響を最小化し、システムの安全性を維持するためのメカニズムを検討しているか。

- ▶ システム安全要求について、単純にシステムとして実装する安全機能以外にも、システムの利用や運用の際の方式面 (システム異常時のシステム管理者やユーザーの対応など) での安全性確保などが検討されているか。
- ▶ 上記を踏まえてシステムで実現すべき安全面の機能要求事項が明確になっているか。

② 確認結果は内部確認レポート (SA105) として整理し、確認作業で指摘された問題およびその対応元を明記した上で、関係者に配布する。

■ 注意すべき事項

- ▶ 安全要求仕様書の確認に際しては、
 - ・ システム開発に関係する開発者・技術者
 - ・ 当該システムの製品仕様書、製品企画書の検討に関わったメンバ
 - ・ 過去に類似のシステム開発に携わった技術者
 なども交えて確認を行うことが望ましい。
- ▶ システムのアーキテクチャ、ハードウェア、安

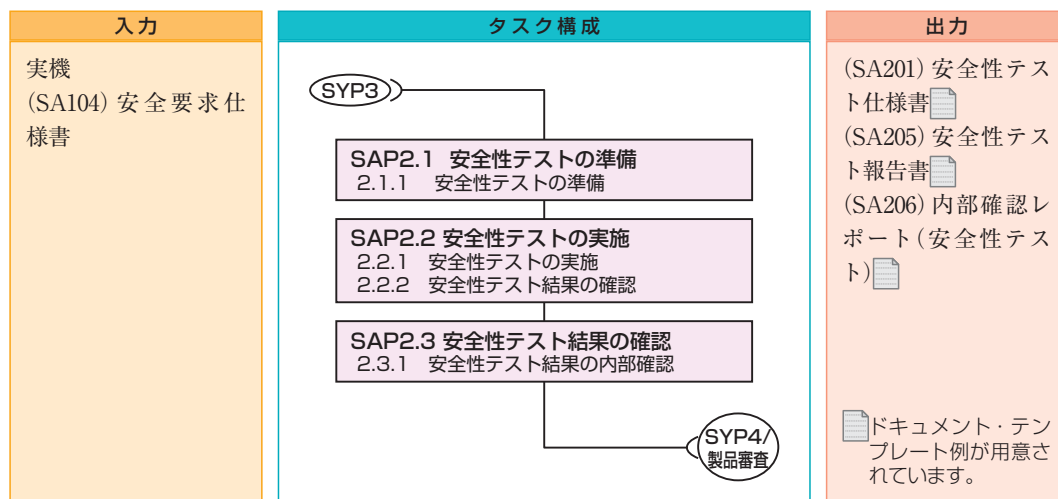
全性、性能効率、使用性などの視点から安全要求仕様書を確認する。

- ▶ システムの安全面に関するトレーサビリティに配慮する。
- ▶ 管理者は内部確認において指摘された問題事項については、解決策やその対応のためのアクションなどもあわせて記載しておくことが望ましい。

- ▶ 要求段階での問題は放置すると開発後半において大きな問題を生むため、この段階で内部確認レポートの情報は、プロジェクトマネージャ / 開発リーダー、製品企画担当者などステークホルダ間で情報共有とコンセンサスを得ておくことが望ましい。

SAP2 安全性テスト

システムの安全性要求事項が実現できているか確認する。



■ 解説

このアクティビティは

- (2.1.1) システムが、安全要求仕様に記載された機能動作を実現しているかどうかを確認するためのテスト仕様を準備し、実際にそのテスト仕様に従い、システムを動作させ、システムとしての安全性のテストを行う。
- (2.2.1) (2.2.2) 安全性テスト結果については、テストの合否判定基準に従って確認を行い、確認した内容を整理して内部確認レポートを作成する。
- (2.3.1) 安全性テストの結果報告書をもとに、製品システム部門としての最終的な合否を判定し、製品審査に必要な情報を整理する。

安全性テストは、システムの開発において、製品の安全性を確認する作業と位置づけることができる。このため、この作業は製品がユーザーに対して危害をもたらすことがないようにさまざまな故障を想定して、漏れがないように実施されなければならない。

■ 留意事項

- ▶ 安全性テストでは、テストの際の動作環境(テスト環境)に注意する。実際にユーザーが製品を利用する環境や故障を想定したテストを実施する。
- ▶ システム開発における製品の安全性を確認するための作業として、製品の正常状態での利

用条件のみだけでなく、正常な利用条件以外や、故障など、テスト項目に漏れがないようにする。

■【参考】手法およびツール

- ▶ システム動作に関するシミュレータ装置など
- ▶ 信頼度成長曲線
- ▶ バグ管理ツール

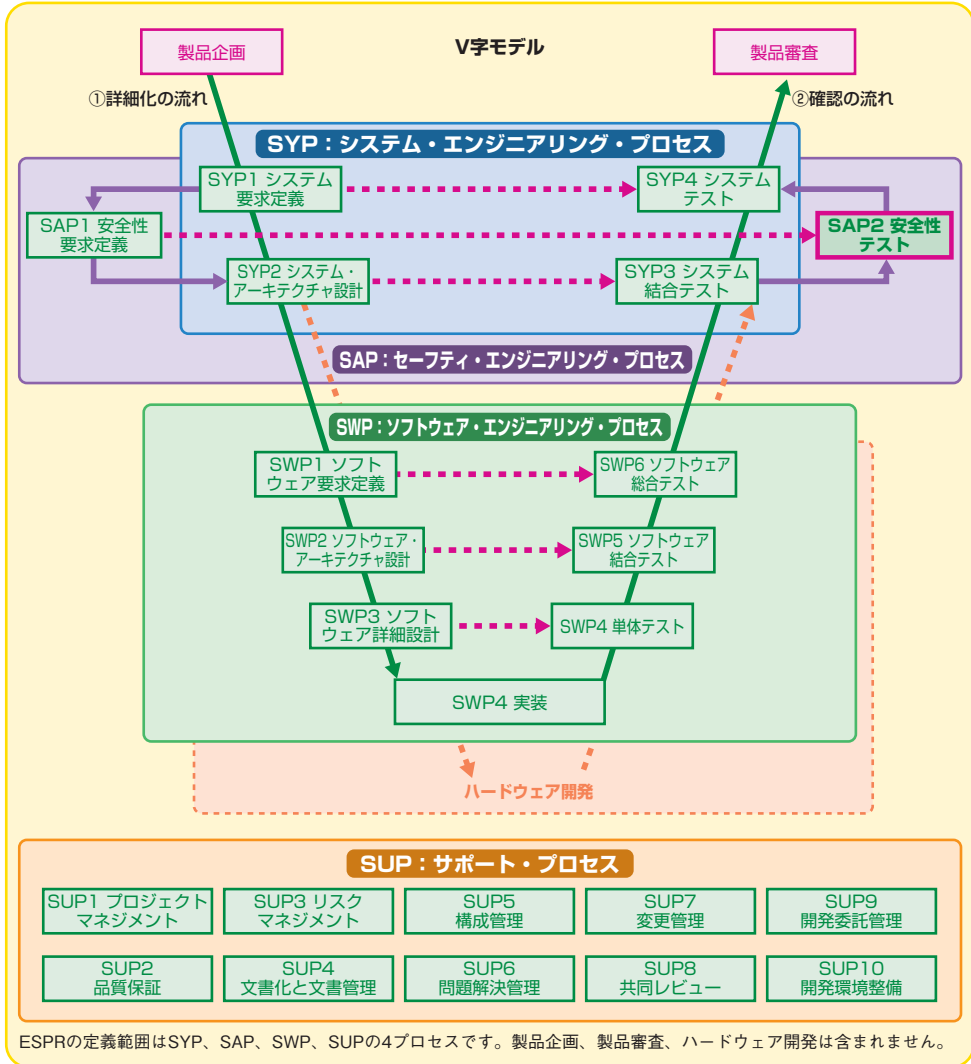


図2.15 V字モデルと開発プロセス (SAP2 安全性テスト)

SAP2.1 安全性テストの準備

安全性テストの準備をする。

2.1.1 安全性テストの準備




入力


(SA104) 安全要求仕様書
(SU601) 不具合管理票
(修正確認の場合)

概要

システムの安全性テストを行うための準備をする。

出力

(SA201) 安全性テスト仕様書 
(SA202) 安全性テストデータ 
(SA203) 内部確認メモ
(安全性テスト仕様) 

ドキュメント・テンプレート例が用意されています。

参照情報

過去の安全性テスト仕様書

■ 実施内容

- ①安全性テストで確認すべき事項をリストアップし、安全性テストのテスト項目を作成する。
 - ②単純にシステムとして実装する安全機能のテスト以外にも、システムの利用や運用の際の方式面(システム異常時のシステム管理者やユーザの対応など)もテスト項目を考える。
 - ③上記のテスト項目を実施するためのテストデータを作成する。
 - ④テスト結果の判定基準や、テスト全体の評価基準や完了基準なども用意しておく。
 - ⑤システムの安全性確認のための計画を検討しておく。
 - ⑥上記のテスト項目、およびテストデータなどを整理し、安全性テスト仕様書を作成する。
- ⑦修正確認テスト項目を準備する(修正確認の場合)。
 - ▶システム結合テストにおいて不具合が検出されて修正を行った場合には、その不具合が解消されたこと、また修正により別の不具合が発生していないことを確認するためのテスト項目を準備する。
 - ▶不具合の内容により、テスト範囲を決定しテスト項目を選択する。

■ 注意すべき事項

- ▶ システムに対する外乱や入力データの誤差、あるいは、ユーザの操作ミスなどの可能性も含めて、テスト項目を検討する。
- ▶ 安全性テスト仕様書は確認しておく。
- ▶ 安全性テストで利用する手法や技法をあらかじめ検討し、選択しておく。
- ▶ 安全性テスト計画では、誰が何時やるのか、およびテスト対象の特定、テスト戦略、テスト環境などを明記しておく。
- ▶ 修正確認テストの準備の考慮点
 - ・ テストの実施範囲は、影響する部分をすべて網羅しているか
 - ・ テストは基本的に既に作成したテスト項目を再利用する。ただし、修正量が多い場合や、影響範囲が広い場合などは、新たなテスト項目を用意することも検討する
- ▶ 文書化での留意点
 - ・ 変更履歴を添付し、変更箇所なども明示する
 - ・ 作成文書に関しての責任所在を明示する
 - ・ 作成文書は構成管理および変更管理にて確実に管理する

関連プロセス：SUP5 構成管理、SUP7 変更管理

■ SAP2.2 安全性テストの実施

安全性テストを実施する。

2.2.1 安全性テストの実施

入力

実機
(SA201) 安全性テスト仕様書
(SA202) 安全性テストデータ
(SU601) 不具合管理票
(修正確認の場合)

概要

安全性テスト仕様書に従って安全性テストを実施する。

出力

(SA204) 安全性テスト結果

■ 実施内容

- ① 安全性テスト仕様書に記載されたテスト項目に従い、順次、安全性に関する動作確認を行う。
 - ▶ 代替項目やデータを用いてテストした場合にも、その理由を明記した上で記録に残す。
 - ▶ 準備したテスト項目が実施できなかった場合には、その理由を明記した上で記録に残す。また、その理由の妥当性について判断する。
- ② 修正確認テストを実施する。
 - ▶ 不具合が解消されたかをテストする。
 - ▶ 不具合の修正により、別の不具合が発生していないかをテストする。

■ 注意すべき事項

- ▶ 安全性確認手段として、テストの補足のためにシミュレーションなどの手法も活用する。
- ▶ 修正確認テストでは修正後の最新版でテストを実施しているか確認する。

2.2.2 安全性テスト結果の確認



入力


(SA201) 安全性テスト仕様書
テスト対象のシステム
(SA204) 安全性テスト結果
(SU601) 不具合管理票
(修正確認の場合)

概要

安全性テストの結果を確認し、合否判定をする。

出力

(SA205) 安全性テスト報告書 
・(SU601) 不具合管理票 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

- ① 安全性テスト仕様書に記載した合否判定基準などを参考に、安全性テストの合否判定を行う。
- ② 安全性テストで検出された不具合については、ソフトウェア起因の不具合、ハードウェア起因の不具合など、不具合の原因箇所の特定を行い、不具合管理票に記載する
- ③ テスト報告書には
 - ▶ テスト方法、テスト環境や利用したツールやデータ
 - ▶ 検出した不具合数や重大不具合、不具合内容の分析
 - ▶ テスト完了の判断と根拠などを明確に記述する。

関連プロセス：SUP6 問題解決管理

■ 注意すべき事項

- ▶ テストにおいて不具合が検出された場合には、まず、その不具合の再現を試み、その際のコンテキストやシステムの状況を明確にする。

SAP2.3 安全性テスト結果の確認

安全性要求定義で定義された要求事項を正しく実現されているかテスト結果を確認する。

2.3.1 安全性テスト結果の内部確認


入力


(SA104) 安全要求仕様書
(SA201) 安全性テスト仕様書
(SA203) 内部確認メモ
(安全性テスト仕様)
(SA205) 安全性テスト報告書
(SU601) 不具合管理票

概要

安全性テスト結果報告書の内容を確認し、未解決な問題の有無や対応について確認し、内部確認レポートとしてまとめる。

出力

(SA206) 内部確認レポート (安全性テスト) 

ドキュメント・テンプレート例が用意されています。

■ 実施内容

①下記の視点で安全性テスト結果を確認する。

- ▶ 未解決となっている問題が確認された場合には、
 - ・ その問題の重要度を評価し、
 - ・ システム全体としての機能への影響やシステムの信頼性・安全性に関わる重大な問題の場合には、
 - a. システム開発(ソフトウェア、ハードウェア)プロセスへの差し戻し
 - b. システム利用条件面などでの制約の付与
 - c. リリース計画の見直しなどを検討し、具体的な対策を実施する。

関連プロセス：SUP8 共同レビュー、SUP1 プロジェクトマネジメント

- ▶ 未実施となっているテスト項目がないか。未実施となっている場合は、未実施となった理由を確認して対応を検討する。

②確認結果は内部確認レポート(SA206)として整理し、確認作業で指摘された問題およびその対応元を明記した上で関係者に配布する。

■ 注意すべき事項

- ▶ 不具合検出数は、品質基準を満たしているか確認する。
- ▶ 修正確認テストの場合は、テスト範囲と環境は適切であったか確認する。
- ▶ 管理者は内部確認において指摘された問題事項については、解決策やそのためのアクション(共同レビュー開催など)などもあわせて記載しておくことが望ましい。

SUP：サポート・プロセス

サポート・プロセスは組込みソフトウェアの開発を組織的に実行するため、開発作業（システム・エンジニアリング・サービス、ソフトウェア・エンジニアリング・プロセス、セーフティ・エンジニアリング・サービス）を横断的に支える管理的側面に関する作業から構成されます。このプロセスには下記のアクティビティが含まれます。

ISO/IEC12207、15288では、さらにこのプロセスを組織/支援などのライフサイクルプロセス群に分割し、詳細に定義していますが、ESPR Ver.2.0では、下図のピンク色で塗り分けられたアクティビティについて整理してあります。


| ID | アクティビティ | アクティビティの概要 | 構成するタスク |
|-------|--------------|---|---|
| SUP1 | プロジェクトマネジメント | 組込みソフトウェアを開発するための開発プロジェクトを定義し、そのプロジェクトの活動を円滑に進めるための作業を規定する。 | SUP1.1 プロジェクト計画書作成 SUP1.2 プロジェクト実施状況の把握 SUP1.3 プロジェクトの制御 SUP1.4 プロジェクト完了報告書の作成 |
| SUP2 | 品質保証 | 開発している組込みソフトウェアの品質が要求や市場ニーズに合致するように、開発過程での品質の作りこみを実現するための作業を規定する。 | SUP2.1 品質目標の設定 SUP2.2 品質保証方式の確定 SUP2.3 品質の可視化に基づく品質制御 |
| SUP3 | リスクマネジメント | 組込みソフトウェアの開発過程で発生しうるリスクを早期に把握し対策を講じる。 | SUP3.1 リスクの洗い出しと把握 SUP3.2 リスクのモニタリング SUP3.2 リスク対策の決定と実行 |
| SUP4 | 文書化と文書管理 | ソフトウェア・エンジニアリング・プロセスで実施した作業結果を文書に整理し、管理する際の作業を規定する。 | SUP4.1 文書の作成と確認 SUP4.2 文書の配布 SUP4.3 文書の保守と管理 |
| SUP5 | 構成管理 | 組込みシステムを構成する個々のユニットやそれらの設計情報などの構成を把握し、管理する。 | SUP5.1 構成管理対象物の把握 SUP5.2 構成管理/変更管理履歴の管理 |
| SUP6 | 問題解決管理 | 開発の過程で生ずるさまざまな問題点や課題を把握し、それらへの対策や解決状況を管理する。 | SUP6.1 問題の記録と原因分析 SUP6.2 影響分析と対策立案 SUP6.3 対策の実行 SUP6.4 対策結果の確認 |
| SUP7 | 変更管理 | 開発着手後に発生する要求や設計の変更とそれらへの対応を管理する。 | SUP7.1 変更要求情報の記録 SUP7.2 変更による影響の分析 SUP7.3 変更計画の立案と実施 SUP7.4 変更結果の確認 |
| SUP8 | 共同レビュー | 開発作業の節目ごとに当該プロセスでの作業結果が適切であったかどうかを、関係者間で技術面、管理面の両面から確認する。 | SUP8.1 レビューの準備 SUP8.2 レビューの実施 SUP8.3 レビュー結果の確認とフォロー |
| SUP9 | 開発委託管理 | 一部のプロセスを外部委託する際に必要となる作業を規定する。 | SUP9.1 発注の準備と契約 SUP9.2 開発委託作業のモニタリング |
| SUP10 | 開発環境整備 | 設計から、実行モジュール作成、テストに到るまでの開発に必要な環境（実装環境、テスト環境など）を整備し管理する。 | SUP10.1 開発環境整備計画の立案 SUP10.2 開発環境の構築 SUP10.3 開発環境の維持 |

SUP1 プロジェクトマネジメント

組込みソフトウェアを開発するための開発プロジェクトを定義し、そのプロジェクトの活動を円滑に進めるための作業を規定する。

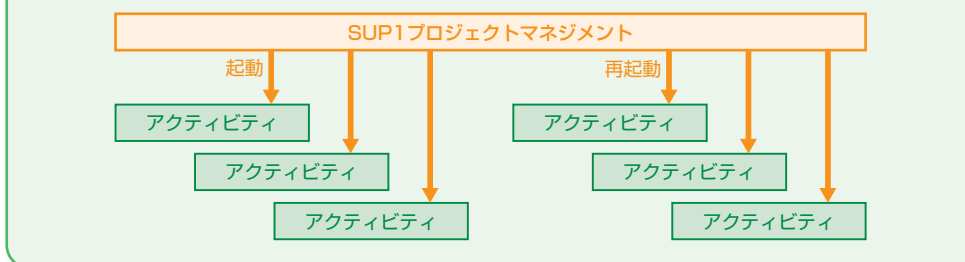
| タスク | 入力 | 主たる作業 | 出力 |
|-------------------------|---|---|--------------------|
| SUP1.1 プロジェクト計画書の作成 | <ul style="list-style-type: none"> システム要求仕様 開発要員などの情報 システムのリリース計画 <p>参照情報</p> <ul style="list-style-type: none"> プロジェクト開発計画書ガイド | <p>以下の作業を実施し、プロジェクト計画書(SU101)を作成する。</p> <ol style="list-style-type: none"> ①開発対象およびそのシステム要求仕様を把握する。 ②開発プロジェクトに関する期間面、リソース面などの制約を把握する。 ③開発プロジェクトの目標を明確にする。 ④組込みソフトウェア開発で実施すべき作業を検討し、WBSを作成するとともに開発プロセスを設計する。 ⑤各作業に携わる技術者などのアサインを検討する。 <p>*開発計画書は開発着手前および開発着手後の情報が具体化した段階など複数段階で徐々に詰めていくことが望ましい。</p> | (SU101) プロジェクト計画書 |
| SUP1.2 プロジェクト実施状況の把握 | (SU101) プロジェクト計画書 | <p>プロジェクト計画書(SU101)に記載された事項どおりにプロジェクトが進んでいるかどうかをモニターする。</p> <p>特に、プロジェクトの範囲、予算とコスト、リソースやスケジュールとの整合性などをチェックする。</p> <p>また組込みソフトウェアの開発の場合、ハードウェア開発との整合性などについても注意する。</p> | (SU102) プロジェクト状況報告 |

| タスク | 入力 | 主たる作業 | 出力 |
|---------------------------------|--|--|--|
| SUP1.3 プロジェクトの制御 | (SU102) プロジェクト状況報告 (SU603) 課題管理票 | <p>プロジェクト状況報告 (SU102) において、当初計画からの遅延や障害が確認できる場合には、その遅延などの原因分析を行い、必要な対策の検討を行い実施する。こうしたプロジェクト遂行上で発生する問題点や課題は課題管理票 (SU603) に記録・更新し、確実に管理する。</p> <p>関連プロセス：SUP6 問題解決管理</p> <p>また、これに対応して、適宜、プロジェクト計画の見直しを行う。</p> | (SU101) プロジェクト計画書 (プロジェクト計画見直しの場合) (SU603) 課題管理票 |
| SUP1.4 プロジェクト完了報告書の作成 | (SW105) ソフトウェア要求仕様書 (SW606) ソフトウェア総合テスト報告書 (SU801) 共同レビュー記録(ソフトウェア総合テスト) (SY106) システム要求仕様書 (SY406) システムテスト報告書 (SU801) 共同レビュー記録(システムテスト) (SA104) 安全要求仕様書 (SA201) 安全性テスト仕様書 (SA206) 内部確認レポート(安全性テスト) (SU601) 不具合管理票 (SU603) 課題管理票 (SU101) プロジェクト計画書 (SU102) プロジェクト状況報告 | <p>ソフトウェア総合テスト、システムテストの結果報告書や共同レビュー結果を元に、ソフトウェア/システム開発の最終的な状況・結果の報告や製品リリース判断の元となる品質情報などを報告書として整理する。</p> <p>また、開発を通して確認できた開発プロセス面、開発技術などの課題を整理し、次回開発における改善項目として位置づける。</p> <p>完了報告をもってソフトウェア/システム開発は開発部門の手を離れることとなる。このため、開発完了報告は製品製造や製品リリースを担当する部門への引渡し資料となることを意識して整理する。</p> | (SU103) プロジェクト完了報告書(ソフトウェア開発) (SU104) プロジェクト完了報告書(システム開発) |

ドキュメント・テンプレート例が用意されています。

プロジェクトマネジメント・アクティビティと他のアクティビティとの関連

プロジェクト計画書(SU101)に従ってアクティビティを起動または再起動し、プロジェクトを制御する。



■ 参考情報

本ガイドでは、プロジェクトマネジメントに関して、プロジェクト計画書の作成、プロジェクトの実施状況の把握および制御に絞って記述している。一般的なプロジェクトマネジメントについては、PMI (Project Management Institute) によって、プロジェクトマネジメントを行う際の基本動作がPMBOK (Project Management Body of Knowledge)として体系的に整理されている。このサブタスクの実施については、これらを参考にすることも有効である。また、SECの小冊子『プロジェクトマネジメントの勧め』なども参考になる。

プロジェクト計画書作成については、SECにより整理された『組込みソフトウェア向け プロジェクトマネジメントガイド [計画書編]』を参考とし、その中に掲載されているプロジェクト計画書テンプレートを活用すると効率的にプロジェクト計画書を作成することができる。

■ [参考] 手法およびツール

- ▶ WBS (Work Breakdown Structure : 作業分割構成)
- ▶ PERT (Program Evaluation and Review Technique : 遂行評価レビュー技法)

SUP2 品質保証

開発している組込みソフトウェアの品質が要求や市場ニーズに合致するように、開発過程での品質の作りこみを実現するための作業を規定する。

| タスク | 入力 | 主たる作業 | 出力 |
|----------------------------|---|---|-----------------|
| SUP2.1 品質目標の設定 | 製品企画書 (SY106) システム要求仕様書 (SW105) ソフトウェア要求仕様書 | <p>品質の作り込み状況のチェックや品質状況を予想するために、対象システムを利用するユーザやコンテキストを考慮して、明確な品質目標値を設定する。</p> <p>品質目標値は、機能性、信頼性、使用性、効率性、保守性、移植性などの観点から検討、整理し、品質保証計画書 (SU201) を作成する。プロダクトの品質目標値の例としては、以下のものがある。</p> <ol style="list-style-type: none"> ①設計工程と製造工程での不具合検出率 (不具合件数/開発規模) ②設計工程と製造工程でのレビュー実施率 (レビュー時間/開発規模) ③テスト密度 (テスト項目数/開発規模) ④テスト工程での障害検出率 (障害件数/開発規模) ⑤テスト工程での障害収束率 ((該工程までの発生累積障害件数/目標障害件数) * 100) <p>品質目標についてはステークホルダ間で合意しておく。</p> | (SU201) 品質保証計画書 |
| SUP2.2 品質保証方式の確定 | (SU201) 品質保証計画書 | <p>品質目標を達成するために、品質保証の観点からどのような組織構成、責任分担で品質保証を進めていくか、また、どのようなルーチンで品質保証を進めていくかを明確にする。目標の達成度合いを測る仕組みとしては、以下の運用ルールを明確にしておく。</p> <ol style="list-style-type: none"> ①品質関連情報収集の仕組み (いつ、誰が、何を、どのような手段で、誰に、報告するのか) ②品質関連情報活用の仕組み (誰が、何を、もとに、何のために、使うか) <p>品質保証活動を進める推進責任者および品質に責任を負う責任者を明確にしておく。</p> | (SU201) 品質保証計画書 |

| タスク | 入力 | 主たる作業 | 出力 |
|--------------------------|---|---|-----------------|
| SUP2.3 品質の可視化に基づく品質制御 | 各種成果物（仕様書、設計書、ソースコード、実機） (SU201) 品質保証計画書 | ①システムの品質保証における主要なイベント（レビューやテスト）において、品質保証計画書に定義した品質目標値に対する成果物の品質の達成度合いを確認する。 ②成果物の品質が目標値と乖離している場合はその対策を立案し、実施する。 対策の実施に関しては、設計担当者やプロジェクトマネージャと連携し、必要な措置を講じる。 | (SU201) 品質保証計画書 |

■ 参考情報

製品として開発されるシステムやソフトウェアでは、どの程度の品質を目標に開発するかを明確に定め、そのためにどのような品質保証の仕組みを動かしていくかあらかじめ決めておく必要がある。まず、対象のシステムやソフトウェアに求められる品質の目標を決める際には、数値などで定量的に品質を押さえることができるようにし、その目標を達成するための品質保証の体制や仕組みを定めておく。また、レビューやテストといった品質目標を達成する上で欠くことのできない主要なイベントの日程や進め方も決めておく。また、システムの安全性要求なども考慮して品質保証活動を推進する。

■ 【参考】手法およびツール

- ▶ 信頼度成長曲線

SUP3 リスクマネジメント

組込みシステムの開発過程で発生しうるリスクを早期に把握し、対策を講じる作業を規定する。

| タスク | 入力 | 主たる作業 | 出力 |
|------------------------------|---|--|----------------------|
| SUP3.1 リスクの洗い出しと把握 | 過去の類似プロジェクトにおけるトラブル事例、リスク例等 製品企画書 製品仕様書 | <p>①プロジェクトのリスクマネジメントに関する基本方針とそれを実行に移すための仕組みを明確にする。</p> <ul style="list-style-type: none"> ・管理方法(リスクの特定、分析、優先順位付け、計画、モニターおよび解決など) ・リスクマネジメントの責任と権限 <p>②プロジェクトにおいて発生が予見されるさまざまなリスクを洗い出す。</p> <p>リスクを抽出する方法としては以下のものがある。</p> <ul style="list-style-type: none"> ・過去の類似プロジェクトにおけるトラブル事例やリスク例をあたる ・仕様書や設計書等のレビューでリスクを検討する ・特性要因図やデシジョン・ツリーを使って洗い出す <p>③抽出したリスクごとに、その発生確率、トリガー、発生した時の影響度(影響範囲、工数等)、対策および優先順位付け、コンティンジェンシープランなどを検討し、リスクテーブルなどで整理し、リスクマネジメント計画書(SU301)を作成する。組込みの場合、ハードウェア関連、動作環境、ビジネス環境などに起因するリスクに特に注意しておく。</p> | (SU301) リスクマネジメント計画書 |
| SUP3.2 リスクのモニタリング | (SU301) リスクマネジメント計画書 各種成果物等(仕様書、設計書、ソースコード、実機) | <p>①システム開発における主要なイベント(レビューやテスト)において、リスクマネジメントの視点からリスク対策が実行されているか、リスクの優先順位などの状況が変わっていないかなどをモニタリングする。</p> <p>②モニタリングにより、新しいリスクの抽出や、除去されたリスクの削除を行ない、リスクマネジメント計画書(SU301)を更新する。</p> <p>リスクモニタリングはプロジェクトの初期のインシヤルリスクのみでなく、プロジェクト進行に合わせて定期的にモニタしていく。</p> | (SU301) リスクマネジメント計画書 |

| タスク | 入力 | 主たる作業 | 出力 |
|-------------------------------|------------------------------|---|------------------------------|
| SUP3.3 リスク対策 の決定と 実行 | (SU301) リスク マネジメン ト計画書 | ①リスクが発生する前にリスクファクターを除去する。 ②リスクが除去できない場合は、リスクの発生確率を 軽減するために、コンティンジェンシープランを実行 する。 ③リスクが発生した場合は、リスクマネジメント計画 書(SU301)に基づき、その対応策を実行する。 対策の実施に関しては、設計担当者やプロジェクトマ ネージャと連携し、必要な措置を講じる。 | (SU301) リスク マネジメント計 画書 |

■ 参考情報

システムやソフトウェア開発の過程では、品質目標に影響を与えるさまざまな事象が発生する。こうしたトラブルの多くは、事前に予見できる場合があり、通常、プロジェクトの着手前や途中で、どのような潜在的なトラブルの芽があるか洗い出しておく必要がある。そして、実際のプロジェクトが動き出してからは、こうしたトラブルが発生しているかどうか、また、それらに対して未然に防ぐための対策が講じられているかどうかなどを適切にマネジメントしていく。

■ 【参考】手法およびツール

- ▶ リスク分析技術
 - ・ FTA (Fault Tree Analysis : 故障木解析)
 - ・ FMEA (Failure Modes and Effects Analysis : 故障モード影響解析)

SUP5 構成管理

SYP、SWPやSAPで実施した作業の結果(設計書やソースコードなどのアクティビティ成果物)を整理し、管理する際の作業を規定する。

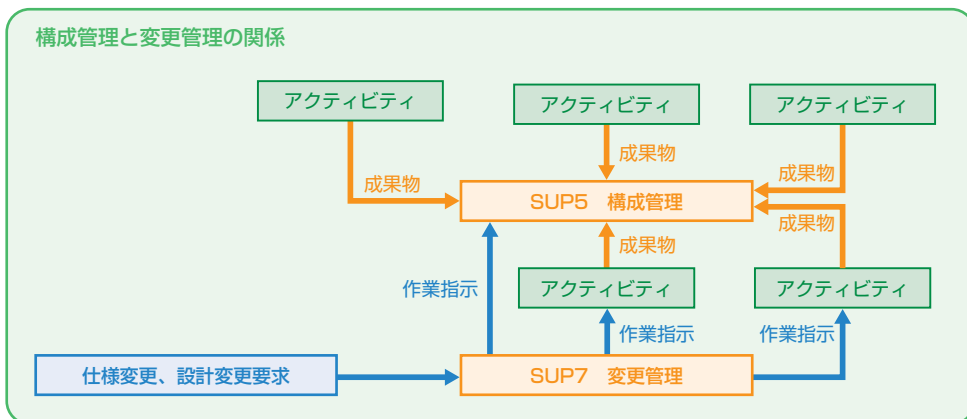
| タスク | 入力 | 主たる作業 | 出力 |
|---------------------------------|---|--|---|
| SUP5.1 構成管理対象物の把握 | 構成管理対象(アクティビティ成果物、開発環境、結合されたシステム等) | ①構成管理の方針と仕組みを明確にする。 ・構成管理の手順とルール 追跡管理できるように表を用いるなどして各成果物間のトレーサビリティを対応付けられるようにする。 ・ベースラインの予定と監査の方針など ②構成管理の対象物を把握し決定する。管理対象になるのは各アクティビティの出力として、以下ものがある。 ・仕様書、設計書、報告書などのドキュメント類 ・ソースコード、結合されたシステムなど ・開発環境 | (SU501) 構成管理表 |
| SUP5.2 構成管理/変更管理履歴の管理 | 構成管理対象(アクティビティ成果物、開発環境、結合されたシステム等) (SU501) 構成管理表 | ①構成管理の対象物に変更が発生した場合にベースラインおよび変更履歴を更新しバージョン管理する。 ②ベースラインへのアクセスを管理する。 ③ベースラインからの成果物の生成を認可する。 ④構成管理情報を関係者に配布する。 組込みの場合、開発途中での仕様変更などが入ることも多いため、確実に管理しておく必要がある(変更内容、理由なども明記する)。 *ベースラインとブランチの扱いに注意する。 | 構成管理対象(アクティビティ成果物、開発環境、結合されたシステム等) (SU501) 構成管理表 |

ベースラインと構成管理の関係



● 用語解説

- * **ベースライン管理**: ベースライン管理とは、ある時点での構成要素をベースライン(出発点)として、以後の変更を管理する手法のこと。ブランチはベースラインから派生したものを指す。



■ 参考情報

最近のシステムやソフトウェアの開発ではシリーズ開発が多いため、さまざまなバージョンが同時に開発される傾向がある。こうした複数のバージョンが同時に進行する開発では、バージョン管理が重要となり、どのバージョンにどのような機能が盛り込まれているかといった情報を適切に管理する必要がある。

このため、一般的には成果物を管理するデータベースなどを用意して成果物の一元管理を行い、派生バージョンの開発の際にはこのデータベースから必要に応じて成果物をチェックイン、チェックアウトして開発作業を行うことが望ましい。派生バージョンの管理を適切に行わないと、製品リリースバージョンにおけるファイルの取り違えなどの事故や不具合発生時の不具合修正対象の把握などの面で深刻な影響が出かねないため、必要に応じて個々のバージョン間の関連を図示するなどして、明確にしておく必要がある。

■ 【参考】手法およびツール

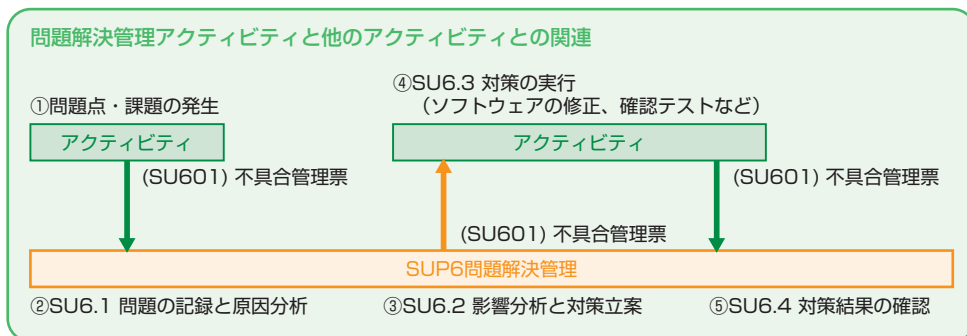
- ▶ 成果物管理データベース

SUP6 問題解決管理

開発の過程で生ずるさまざまな不具合や課題を把握し、それらへの対策や解決状況を管理する。

| タスク | 入力 | 主たる作業 | 出力 |
|---------------------------------|--|--|--|
| SUP6.1 問題の記録と 原因分析 | (SU601) 不具合管理票 (SU603) 課題管理票 | <p>①開発過程で作成される成果物のレビューやテストで確認されたシステムあるいはソフトウェアの不具合を不具合管理票 (SU601) に記録し、各不具合に項番をつけて不具合管理台帳 (SU602) で管理する。</p> <p>②プロジェクト遂行上の課題などを課題管理票 (SU603) に記録し、各課題に項番をつけて課題管理台帳 (SU604) で管理する。</p> <p>関連プロセス：SUP1 プロジェクトマネジメント</p> | (SU602) 不具合管理台帳 (SU604) 課題管理台帳 |
| SUP6.2 影響分析と対策立案 | (SU601) 不具合管理票 (SU602) 不具合管理台帳 (SU603) 課題管理票 (SU604) 課題管理台帳 | <p>①不具合管理票 (SU601) や課題管理票 (SU603) に記載された個々の不具合、課題について、影響範囲を分析・検討する。 また、その結果をもとに問題の重要性や緊急性なども分析し、カテゴライズする。</p> <p>②分析・カテゴライズ結果をもとに、不具合や課題の原因分析を行い、</p> <p>③不具合や課題に対する対策を立案し、不具合管理票あるいは課題管理票 (SU603) に記録する。</p> | (SU601) 不具合管理票 (SU602) 不具合管理台帳 (SU603) 課題管理票 (SU604) 課題管理台帳 |
| SUP6.3 対策の実行 | (SU601) 不具合管理票 (SU602) 不具合管理台帳 (SU603) 課題管理票 (SU604) 課題管理台帳 | 不具合管理票 (SU601) あるいは課題管理票 (SU603) に記載した問題解決策を実行する。 | (SU601) 不具合管理票 (SU602) 不具合管理台帳 (SU603) 課題管理票 (SU604) 課題管理台帳 |

| タスク | 入力 | 主たる作業 | 出力 |
|-------------------|--|--------------------------------|--|
| SUP6.4 対策結果の確認 | (SU601) 不具合管理票 (SU602) 不具合管理台帳 (SU603) 課題管理票 (SU604) 課題管理台帳 | 問題解決策を実行した結果、不具合や課題が解決したか確認する。 | (SU601) 不具合管理票 (SU602) 不具合管理台帳 (SU603) 課題管理票 (SU604) 課題管理台帳 ドキュメント・テンプレート例が用意されています。 |



■ 参考情報

システムあるいはソフトウェア開発の過程で作成されるさまざまな中間成果物や最終成果物は、レビューやテストなどの作業により、その不具合が検出、指摘される。「SUP6問題解決管理」では、これらの不具合を適切に処理していくことを目的としている。また、こうした不具合以外でも、プロジェクト遂行上では、さまざまな問題点・課題点が発生する。

エンジニアリング・プロセスに含まれる個々の作業から、サポート・プロセスに含まれる作業まで、さまざまな課題や問題の発生が想定されるが、どのような問題・課題であっても、それを問題として認識・可視化し、その影響などをもとに問題の大きさや緊急性を判断して、具体的な対策を実行することが重要である。「SUP6 問題解決管理」としては、こうした作業やプロセス上の問題や課題も、その範疇に含めて考える（ただし、この場合には「(SU601) 不具合管理票」は若干の読みかえが必要になることがある）。

対策立案実行については、その対策を「いつまでに」「誰が」実行するかを明確にするとともに、その対策が完了したと判断するための環境基準なども明確にしておくことが望ましい。

■ 【参考】手法およびツール

- ▶ プログラムスライサー
- ▶ 不具合管理ツール

SUP7 変更管理

開発着手後に発生する要求や設計の変更とそれらへの対応を管理する。

| タスク | 入力 | 主たる作業 | 出力 |
|----------------------|--|--|--|
| SUP7.1 変更要求情報の記録 | (外部からの) 変更要求 | 開発着手後に、顧客やハードウェア部門など外部から仕様や設計などの変更が求められる場合には、それらの変更要求をきちんと確認し、変更要求管理票 (SU701) に記入する。 各変更要求は項番をつけて変更要求管理台帳 (SU702) で管理する。 | (SU701) 変更要求管理票 (SU702) 変更要求管理台帳 |
| SUP7.2 変更による影響の分析 | (SU701) 変更要求管理票 (SU702) 変更要求管理台帳 | 変更要求管理票 (SU701) に記載された個々の変更要求項目について、それらを受け入れた場合に、 ・仕様や設計上のどの部分に影響が加わるか ・関連ドキュメントのどの部分に影響が出るか ・他部門の作業などにどのような影響が及ぶかなどを詳細に分析評価する。 | (SU701) 変更要求管理票 (SU702) 変更要求管理台帳 |
| SUP7.3 変更計画の立案と実行 | (SU701) 変更要求管理票 (SU702) 変更要求管理台帳 | 変更要求管理票 (SU701) に基づき、変更範囲や変更時期を判断し関連部門などに変更の連絡をしたうえで、変更を行う。 仕様に影響を及ぼす変更要求については、関係者間で確認・合意され、文書化しておくことに留意する。 安全 | (SU701) 変更要求管理票 (SU702) 変更要求管理台帳 (SU703) 変更済み成果物 |
| SUP7.4 変更結果の確認 | (SU701) 変更要求管理票 (SU702) 変更要求管理台帳 (SU703) 変更済み成果物 | 変更内容が変更要求に合致しているかどうかを確認する。 | (SU701) 変更要求管理票 (SU702) 変更要求管理台帳 |

■ 参考情報

安全：セーフティに関連する作業

組込みソフトウェアの開発では、開発着手後の仕様変更や設計変更などさまざまな変更が入ることが少なくない。これらさまざまな変更要求は、実際に成果物などに手を加える前に、変更による他部分への影響など事前評価しておく必要がある。特に、ソフトウェア内部で独立性が低い、あるいは、周囲との強結合を有する部分などの変更は要注意である。




また、変更を繰り返していくと、ソフトウェアの内部構造が複雑になる場合なども発生するため、ソフトウェアを変更する際には常に、ソフトウェア・アーキテクチャを意識した変更が求められる。

■ 【参考】手法およびツール

- ▶ Intersolv PVCS/VCS (Version Control System) : バージョン管理システム)

SUP8 共同レビュー

開発作業の節目ごとに当該プロセスでの作業結果が適切であったかどうかを、関係者間で技術面、管理面の両面から確認する。

| タスク | 入力 | 主たる作業 | 出力 |
|--------------------------|------------------|---|--|
| SUP8.1 レビューの準備 | レビュー対象成果物 | プロジェクトの状況などを考慮し、レビュー対象(成果物)、レビュースケジュール、レビュー参加者などを決め、レビューの準備をする。 | (レビュー開催通知等) |
| SUP8.2 レビューの実施 | レビュー対象成果物 | レビュー対象成果物について、技術的側面、管理的側面などからレビューを実施する。 レビュー対象物は、あらかじめレビュー会議参加者に事前配布し、内容確認などをしておくことよ。またレビューでは問題検出を中心に進め、レビュー記録を作成する。 | (SU801) 共同レビュー記録  |
| SUP8.3 レビュー結果の確認とフォロー | (SU801) 共同レビュー記録 | レビュー会議での検討内容を確認し整理したレビュー記録を、レビュー参加者や関係者に配布する。 また、レビューでの指摘事項については、このレビュー記録をもとにその対策が検討、実施されたかどうかを継続的にフォローしていく。 | (SU801) 共同レビュー記録   ドキュメント・テンプレート例が用意されています。 |

■ 参考情報

レビューはソフトウェア開発過程での品質作りこみのための重要な作業の1つとして位置づけることができる。エンジニアリング・プロセスで作成したさまざまな成果物を対象に、開発工程の節目で実施することが望ましい。レビューの形式は、開発グループ内でのレビュー（ピアレビュー/グループ内確認）や、本プロセスに示す共同レビューなどがある。

共同レビューでは、成果物の作成を担当した技術者だけでなく、製品開発に関係するステークホルダなども参加し、多視点でのレビューを行う。レビューで確認された課題点や問題点は、その検討担当者と期日などを決めてレビュー記録として残し、その課題解決のフォローにつなげていく。

■ 【参考】手法およびツール

- ▶ CBR (Checklist-based Reading : チェックリストに基づくレビュー)
- ▶ PBR (Perspective-based Reading : ユーザやテスト担当者などの観点別に行うレビュー)

SUP10 開発環境整備

設計から、実行モジュール作成、テストに到るまでの開発に必要となる環境(実装環境、テスト環境など)を整備し管理する。

| タスク | 入力 | 主たる作業 | 出力 |
|-------------------------------|--|--|---------------------|
| SUP10.1 開発環境整備計画の立案 | (SU101) プロジェクト計画書 (SW105) ソフトウェア要求仕様書 (SW205) ソフトウェア・アーキテクチャ設計書 (SW305) ソフトウェア詳細設計書 (SW401) 単体テスト仕様書 (SW501) ソフトウェア結合テスト仕様書 (SW601) ソフトウェア総合テスト仕様書 | ①設計から実行モジュール作成までのソフトウェア開発を進めていく上で必要となる開発環境の計画を立案する。 ②ハードウェア計画、開発期間、予算、要員スキル等を考慮し、単体テストから総合テストに関するデバッグ/テスト環境の構築計画を立案する。 ③管理面(進捗管理、生産物管理、品質保証、仕様管理等)の作業環境の計画を立案する。 | (SU1001) 開発環境整備計画書 |
| SUP10.2 開発環境の構築 | (SU1001) 開発環境整備計画書 | ①計画に従って必要な環境を必要な時期までに構築する。 ②利用時期までに利用者の教育も行っておく。 ③使用可能な状態であることを確認する。 ④提供遅れ、品質不良、数量不足などについて問題を解決する。 | (SU1002) ソフトウェア開発環境 |
| SUP10.3 開発環境の維持 | (SU1002) ソフトウェア開発環境 | ①ソースコードの変更やツールのバージョンアップに合わせて開発環境を更新するなど、開発環境を維持する。 ②更新後は開発環境が使用可能な状態であるか確認する。 | (SU1002) ソフトウェア開発環境 |

■ 参考情報

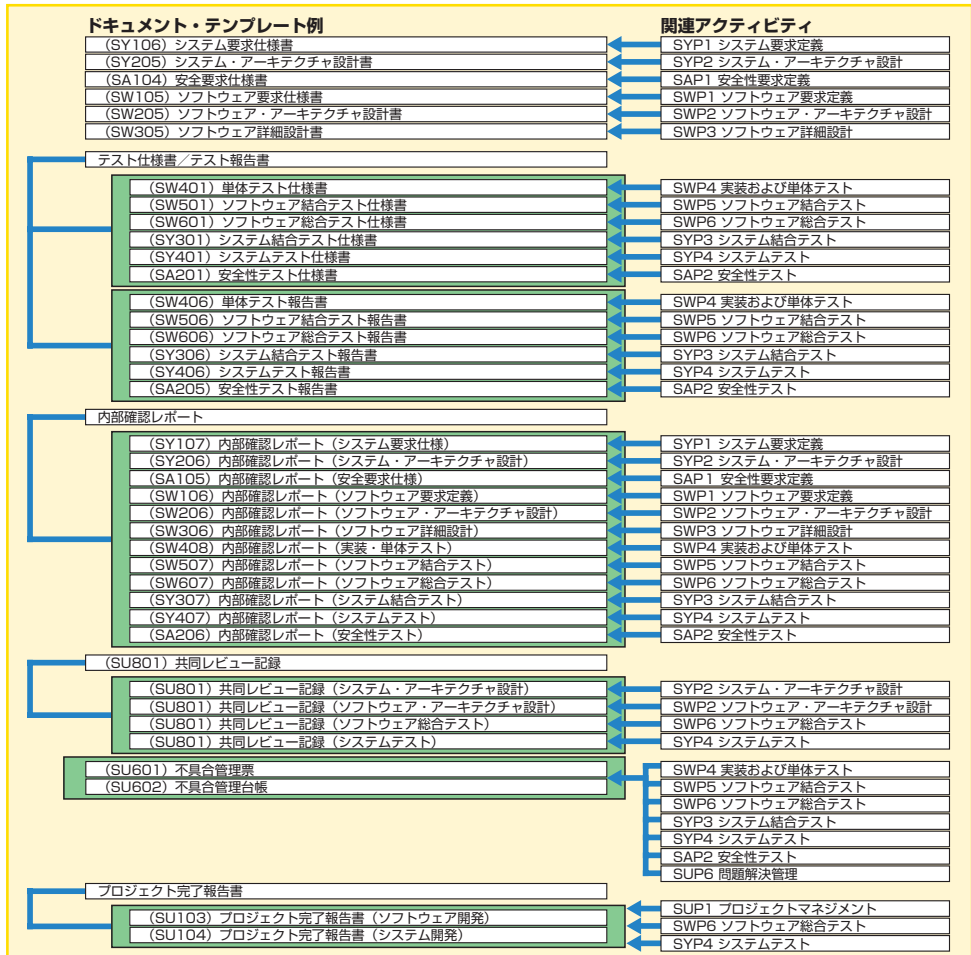
開発環境の整備は、遅れ気味になることが多いため、ソフトウェア開発をスムーズに行うためにも、早めに構築計画を立て、計画に従って構築していく。

開発環境には、ソフトウェア開発環境とデバッグ/テスト環境の他に管理環境(進捗管理、生産物管理、品質保証、仕様管理等)も含まれる。

2.3 ドキュメント・テンプレート例

本ガイドでは、実際の開発作業の一助となるように、「2.2 プロセス定義書」で示した個々のタスクやサブタスクレベルでの作業結果を整理する際のドキュメントのうち、主要なものをドキュメント・テンプレート例として用意してあります(下図参照)。

ドキュメント・テンプレート例には記載する項目、内容および記載例を各ドキュメントごとに参考情報として示しました。これらの情報は、あくまでも参考ですので、特定の利用方法は指定していません。このため、実際に利用する際には個々の開発プロジェクトに合わせて、項目や語句などを調整する必要があります。



(SY106) システム要求仕様書

システム要求仕様書には、システム要求定義の作業で検討した、システムとして実現が求められる機能要求事項、非機能要求事項や制約条件などを記載します。

■ このドキュメントを作成するサブタスク

SYP1 システム要求定義

- 1.1 システム要求仕様書の作成
- 1.1.6 システム要求仕様書の作成

■ ドキュメントをまとめる際に参照する情報

| | |
|-----------------------|-------------------------|
| 製品企画書 | (SY103) システム非機能リスト |
| 製品仕様書 | (SY104) システム動作制約リスト |
| (SY101) システム機能リスト | (SY105) 優先順位付きシステム要求リスト |
| (SY102) システム機能動作マトリクス | |

■ 記載項目例

- | | |
|-----------|----------------------|
| 1. 概要 | 5. ユースケースとユースケースシナリオ |
| 2. システム構成 | 6. 機能詳細 |
| 3. 機能概要 | 7. 性能・品質等非機能要求詳細 |
| 4. 制約条件 | 8. その他 |

■ このドキュメントを利用するサブタスク

SYP1 システム要求定義

- 1.2 システム要求仕様の確認
 - 1.2.1 システム要求仕様書の内部確認

SYP2 システム・アーキテクチャ設計

- 2.1 システム・アーキテクチャ設計書の作成
 - 2.1.1 設計条件の確認
 - 2.1.2 システム構成の設計
 - 2.1.3 システム全体の振る舞いの設計
 - 2.1.4 インタフェースの設計
- 2.2 システム・アーキテクチャ設計の確認
 - 2.2.1 システム・アーキテクチャ設計書の内部確認

2.3 システム・アーキテクチャ設計の共同レビュー

- 2.3.1 システム・アーキテクチャ設計書の共同レビュー

SWP1 ソフトウェア要求定義

- 1.1 ソフトウェア要求仕様書の作成
 - 1.1.1 制約条件の確認
 - 1.1.2 ソフトウェア機能要求事項の明確化
 - 1.1.3 ソフトウェア非機能要求事項の明確化
- 1.2 ソフトウェア要求仕様書の確認
 - 1.2.1 ソフトウェア要求仕様書の内部確認

SYP4 システムテスト

- 4.1 システムテストの準備
 - 4.1.1 システムテスト仕様書の作成
 - 4.1.3 システムテスト仕様書の内部確認
- 4.3 システムテスト結果の確認
 - 4.3.1 システムテスト結果の内部確認
- 4.4 システム開発の完了確認
 - 4.4.1 システム開発の完了確認

SUP1 プロジェクトマネジメント

- 1.4 プロジェクト完了報告書の作成

■ (SY106) システム要求仕様書の例

表紙

文書名

文書番号

| | |
|------|------|
| 承認 | 作成 |
| 承認者名 | 作成者名 |
| 承認日 | 作成日 |

発行日
発行部署

文書名を記載する。

文書の識別情報を記載する。

作成・承認の責任・確認を記載できる欄を設ける。

発行組織名、発行日を記載する。

改訂履歴

文書名

改訂履歴

| 項番 | 日付 | バージョン | 改訂内容 | 備考 |
|----|----|-------|------|----|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| | | ⋮ | | |
| | | ⋮ | | |

⋮

⋮

文書番号 頁番号 発行日・発行部署

改訂情報を記載できる欄を設ける。

目次

文書名

目次

| | |
|---------------------------|-----|
| 1. 概要..... | 頁番号 |
| 2. システム構成..... | 頁番号 |
| 3. 機能概要..... | 頁番号 |
| 4. 制約条件..... | 頁番号 |
| 5. ユースケースとユースケースシナリオ..... | 頁番号 |
| 6. 機能詳細..... | 頁番号 |
| 7. 性能・品質等非機能要求詳細..... | 頁番号 |
| 8. その他..... | 頁番号 |

文書番号 頁番号 発行日・発行部署

1. 概要

- <記載例>
- ・本書の目的
 - ・本書の位置づけ
 - ・対象ユーザ
 - ・記載範囲、記載内容 など
 - ・参照しているドキュメント など
 - ・定義（用語、略語 など）

ドキュメントの目的、位置づけ、記載内容などの本書の概要および参照しているドキュメント名等を記載する。

2. システム構成

- <記載例>
- ・システム全体構成
 - システム構成要素の名称/基本機能
 - ・システムの動作環境および外部環境

ハードウェアおよびソフトウェアを含めたシステム全体の構成と関係/条件を記載する。

3. 機能概要

- <記載例>
- ・システムとして実現・提供する機能のリストと概要

機能概要を箇条書きでまとめる。詳細は6項に記載する。

4. 制約条件

- <記載例>
- ・製品開発に割くことのできる期間や開発コスト
 - ・製品の耐用年数や製品寿命
 - ・製品に期待される品質（信頼性、安全性、使い勝手 など）
 - ・製品を利用する上で前提として利用されるシステムやハードウェア（周辺システムやハードウェア）
 - ・既存製品仕様との継続性に関する制約
 - ・既存システムの再利用に関する制約・セキュリティ、環境問題に関する制約
 - ・システムの利用などに付随する法的な制約、社会慣習面での制約
 - ・他社などの知的財産権、他社技術などとの関係

制約となる条件を漏れなく記載する。

5. ユースケースとユースケースシナリオ

- <記載例>
- ・製品の利用シーンや利用コンテキストをもとにしたユースケースとユースケースシナリオ

システムを構成する機能ごとにユーザとシステムとのやりとりを考慮し、時系列的にその流れを整理して記載する。

6. 機能詳細

- <記載例>
- ・5項で整理したユースケースを実現する機能の詳細

各機能の詳細を機能ごとに記載する。

7. 性能・品質等非機能要求詳細

<記載例>

- ・信頼性要求
 - システムの異常処理方式
 - システムの異常動作モードからの復帰手順や復帰方式
- ・使用性要求
 - ユーザインタフェースの操作性
- ・効率性要求の検討例
 - システムの実行性能（例：処理速度、起動時間、応答時間 など）
 - リアルタイム処理の程度
 - リソース効率（例：メモリ容量、データサイズ）
- ・保守性要求
 - リモートメンテナンスなど保守の方式とその実現方法
- ・移植性要求
 - システム機能の独立性
- ・セキュリティ要求の検討例
 - 例：データ暗号化、ユーザ認証、ウィルス対策 など

機能で表現できない、性能・品質的な事項を記載する。

8. その他

<記載例>

- ・相互運用性（例：通信プロトコル など）
- ・外部インタフェース要求（例：周辺システムとのインタフェース、ユーザインタフェース など）

その他特記しておくべき事項がある場合には記載する。

(SY205) システム・アーキテクチャ設計書

システム・アーキテクチャ設計書には、システム・アーキテクチャ設計の作業で検討した、要求事項の実現方法(ソフトウェア構成、制御方式など)を記載します。

■ このドキュメントを作成するサブタスク

SYP2 システム・アーキテクチャ設計

- 2.1 システム・アーキテクチャ設計書の作成
- 2.1.5 システム・アーキテクチャ設計書の作成

■ ドキュメントをまとめる際に参照する情報

(SY201) システム構成図(機能ブロック図)
(SY202) 共通機能・データ一覧表
(SY203) システム動作設計書

(SY204) システムインタフェース設計書

■ 記載項目例

- | | |
|----------------------|---------------|
| 1. 概要 | 5. 機能ブロック詳細 |
| 2. システム構成 | 6. システムで扱うデータ |
| 3. 機能ブロック概要 | 7. 例外一覧 |
| 4. 制御方式 | 8. その他 |
| 4.1 制御シーケンス | |
| 4.2 ユースケースと機能ブロックの対応 | |
| 4.3 性能見積 | |

■ このドキュメントを利用するサブタスク

SYP2 システム・アーキテクチャ設計

- 2.2 システム・アーキテクチャ設計の確認
 - 2.2.1 システム・アーキテクチャ設計書の内部確認
- 2.3 システム・アーキテクチャ設計の共同レビュー
 - 2.3.1 システム・アーキテクチャ設計書の共同レビュー

SWP1 ソフトウェア要求定義

- 1.1 ソフトウェア要求仕様書の作成
 - 1.1.1 制約条件の確認

- 1.1.2 ソフトウェア機能要求事項の明確化
- 1.1.3 ソフトウェア非機能要求事項の明確化
- 1.2 ソフトウェア要求仕様書の確認
 - 1.2.1 ソフトウェア要求仕様書の内部確認

SWP2 ソフトウェア・アーキテクチャ設計

- 2.1 ソフトウェア・アーキテクチャ設計書の作成
 - 2.1.1 設計条件の確認
 - 2.1.2 ソフトウェア構成の設計

2.1.3 ソフトウェア全体の振る舞いの設計

2.1.4 インタフェースの設計

SYP3 システム結合テスト

3.1 システム結合テストの準備

3.1.1 システム結合の準備

3.1.2 システム結合テストの準備

3.3 システム結合テスト結果の確認

3.3.1 システム結合テスト結果の内部確認

■ (SY205) システム・アーキテクチャ設計書の例

表紙

文書名

文書番号

| | |
|------|------|
| 承認 | 作成 |
| 承認者名 | 作成者名 |
| 承認日 | 作成日 |

発行日
発行部署

文書名を記載する。

文書の識別情報を記載する。

作成・承認の責任・確認を記載できる欄を設ける。

発行組織名、発行日を記載する。

改訂履歴

文書名

改訂履歴

| 項番 | 日付 | バージョン | 改訂内容 | 備考 |
|----|----|-------|------|----|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| | | ⋮ | | |
| | | ⋮ | | |

⋮

⋮

文書番号 頁番号 発行日・発行部署

改訂情報を記載できる欄を設ける。

目次

文書名

目次

| | |
|---------------------------|-----|
| 1. 概要..... | 頁番号 |
| 2. システム構成..... | 頁番号 |
| 3. 機能ブロック概要..... | 頁番号 |
| 4. 制御方式..... | 頁番号 |
| 4.1 動作シーケンス..... | 頁番号 |
| 4.2 ユースケースと機能ブロックの対応..... | 頁番号 |
| 4.3 性能見積..... | 頁番号 |
| 5. 機能ブロック詳細..... | 頁番号 |
| 6. システムで扱うデータ..... | 頁番号 |
| 7. 例外一覧..... | 頁番号 |
| 8. その他..... | 頁番号 |

文書番号 頁番号 発行日・発行部署

1. 概要

<記載例>

- ・本書の目的
- ・本書の位置づけ
- ・対象ユーザ
- ・記載範囲、記載内容 など
- ・参照しているドキュメント など
- ・定義（用語、略語 など）

ドキュメントの目的、位置づけ、記載内容などの本書の概要および参照しているドキュメント名等を記載する。

2. システム構成

<記載例>

- ・システム全体構成
 - システム構成要素の名称/基本機能
- ・システムを構成する主たる機能（要素）とハードウェアとソフトウェアの機能分担

例：ハードウェアプラットフォーム（MPU名称、接続I/O種、ハード割込みレベルとI/Oの対応、メモリー種類とサイズ）、ソフトウェアプラットフォーム（OS、ミドルウェア、入出力データ形式、各種性能）、周辺デバイス（外部媒体種/制御方式）など
体種/制御方式など

ハードウェアとソフトウェアの役割分担を含めたシステム全体の構成と構成要素の名称、基本機能等を記載する。

3. 機能ブロック概要

<記載例>

- ・システムを構成する機能ブロックの名称/基本機能など
- ・機能ブロック間のインタフェース

例：各機能ブロック間の制御、データを静的に記載したもので、必要に応じて、OS、メモリ、外部記憶媒体、ハードウェア機構を併記する。
システムのコア機能を明確にする。

システムとして実現・提供する機能ブロックの概要を記載する。詳細は5項に記載する。

システム要求仕様書(SY106)に記載された要求をシステム全体(ハードウェア、ソフトウェア、オペレータ、外部記憶媒体、メモリなど)を通してどのように実現するかを、機能ブロックを中心に記載する。

4. 制御方式

4.1 動作シーケンス

<記載例>

- ・動作シーケンス図

個々の機能ブロックがどのように連携してシステムの機能サービスを実現していくかを記載する。

4.2 ユースケースと機能ブロックの対応

<記載例>

- ・ユースケース、ユースケースシナリオと機能ブロックとの対応

システム要求分析で検討したユースケースをもとにして、ユースケースと機能ブロックの対応付けを記載する。

4.3 性能見積

<記載例>

- ・必要性能/時間：x x (ms)
- ・性能実現方式：対象条件とその処理シーケンス
- ・各処理時間見積もり：各処理要素の見積もりおよび根拠

システムの非機能要求を念頭に、システムの動作に関するレスポンスタイムなどの時間制約など、ソフトウェア動作、ハードウェア動作の時間的な側面などに関する概算見積りを記載する。

5. 機能ブロック詳細

<記載例>

- ・XX機能ブロック詳細
- ・構成、機能、入出力インタフェース、処理方式、共通領域 など。

各機能ブロックの構成/機能、インタフェース等を詳細化/具体化する。

6. システムで扱うデータ

<記載例>

- ・センサーやアクチュエータなどでやり取りされる共通データ（種類、量、精度など）

共通データなどシステム全体で共通に扱うデータの使用目的、更新/参照元などを記載する。

7. 例外一覧

<記載例>

- ・異常/例外情報一覧
- 異常/例外コード、メッセージ、意味、対策 など

異常/例外時のコード、メッセージ、意味、対策などを一覧表として記載する。

8. その他

<記載例>

- ・共通制御情報領域詳細
- 共通領域内の各情報領域の名称、構成、形式、サイズ、初期値、アクセス制限 など

システム全体を通して、開発チーム全員で情報共有すべき事柄を記載する。

(SA104) 安全要求仕様書

安全要求仕様書には、安全性要求定義の作業で検討した、システムとして実現が求められる安全度水準や安全要求事項などを記載します。

■ このドキュメントを作成するサブタスク

SAP1 安全性要求定義

- 1.1 安全要求仕様書の作成
- 1.1.4 安全要求仕様書の作成

■ ドキュメントをまとめる際に参照する情報

| | |
|-------|---------------------|
| 製品企画書 | (SA101) 想定システム障害テスト |
| 製品仕様書 | (SA102) システム安全要求リスト |
| | (SA103) システム安全度水準 |

■ 記載項目例

- | | |
|--------------|----------------|
| 1. 概要 | 4. システム安全要求リスト |
| 2. システム構成 | 5. その他 |
| 3. システム安全度水準 | |

■ このドキュメントを利用するサブタスク

SAP1 安全性要求定義

- 1.2 安全要求仕様書の確認
- 1.2.1 安全要求仕様書の内部確認

SYP2 システム・アーキテクチャ設計

- 2.1 システム・アーキテクチャ設計書の作成
 - 2.1.1 設計条件の確認
 - 2.1.2 システム構成の設計
 - 2.1.3 システム全体の振る舞いの設計
 - 2.1.4 インタフェースの設計
- 2.2 システム・アーキテクチャ設計の確認
 - 2.2.1 システム・アーキテクチャ設計書の内部確認
- 2.3 システム・アーキテクチャ設計の共同レビュー
 - 2.3.1 システム・アーキテクチャ設計書の共同レビュー

SWP1 ソフトウェア要求定義

- 1.1 ソフトウェア要求仕様書の作成
 - 1.1.1 制約条件の確認
 - 1.1.2 ソフトウェア機能要求事項の明確化
 - 1.1.3 ソフトウェア非機能要求事項の明確化
- 1.2 ソフトウェア要求仕様書の確認
 - 1.2.1 ソフトウェア要求仕様書の内部確認

SAP2 安全性テスト

- 2.1 安全性テストの準備
 - 2.1.1 安全性テストの準備
- 2.3 安全性テストの確認
 - 2.3.1 安全性テスト結果の内部確認

SYP4 システムテスト

- 4.4 システム開発の完了確認
 - 4.4.1 システム開発の完了確認

SUP1 プロジェクトマネジメント

- 1.4 プロジェクト完了報告書の作成

■ (SA104) 安全要求仕様書の例

表紙

文書名

文書名を記載する。

文書番号

文書の識別情報を記載する。

| | |
|------|------|
| 承認 | 作成 |
| 承認者名 | 作成者名 |
| 承認日 | 作成日 |

作成・承認の責任・確認を記載できる欄を設ける。

発行日
発行部署

発行組織名、発行日を記載する。

改訂履歴

文書名

改訂履歴

| 項番 | 日付 | バージョン | 改訂内容 | 備考 |
|----|----|-------|------|----|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| | | | ⋮ | |
| | | | ⋮ | |

改訂情報を記載できる欄を設ける。

文書番号 頁番号 発行日・発行部署

目次

文書名

目次

1. 概要..... 頁番号

2. システム構成..... 頁番号

3. 想定システム障害リスト..... 頁番号

4. システム安全要求リスト..... 頁番号

5. システム安全度水準..... 頁番号

6. その他..... 頁番号

文書番号 頁番号 発行日・発行部署

1. 概要

<記載例>

- ・本書の目的
- ・本書の位置づけ
- ・対象ユーザ
- ・記載範囲、記載内容 など
- ・参照しているドキュメント など
- ・定義（用語、略語 など）

ドキュメントの目的、位置づけ、記載内容などの本書の概要および参照しているドキュメント名等を記載する。

2. システム構成

<記載例>

- ・システム全体構成
- ・システム構成要素の名称/基本機能

ハードウェアを含めたシステム全体の構成とソフトウェアの位置づけ、およびソフトウェアを取り巻く関係/条件を記載する。
システム要求仕様書やハードウェア仕様書等の関連資料より要求、条件などを整理して記載する。

3. 想定システム障害リスト

<記載例>

- ・システムが通常想定している動作環境・動作モードに関する定義
- ・システムで想定される安全逸脱状況とそれが発生する可能性
- ・システムハザード解析（FTA、FMEA）

システムが想定する安全逸脱状況とそのハザード解析結果を記載する。

4. システム安全要求リスト

<記載例>

- ・システムが安全逸脱状況に陥った場合の回避動作（システム系、人間系などに関する定義）
- ・システムの非正常動作に陥った場合の安全機能に関する要求
例：システムの動作モード（始動、自動、手動、半自動、定常・非定常など）に対応した安全機能
- ・システムの故障回避メカニズムに関する要求

安全性を実現するためにシステムおよびシステムの周辺（外部システム、ユーザなど）に求められる要件を記載する。

- 例：
- ・システムの安全メカニズムとしてロジックの二重化やフェールセーフ、フルブールプなどの機構
 - ・システムに対する外乱や入力データの誤差、あるいは、ユーザの操作ミスなどの可能性も含めて、これらの影響を最小化し、システムの安全性を維持するためのメカニズム
 - ・システムの利用や運用の際の方式面（システム異常時のシステム管理者やユーザの対応など）での安全性確保の仕組み

5. システム安全度水準

<記載例>

- ・安全機能ごとにシステムに求められる安全性の水準（安全度水準）

システムに求められる安全性の水準（安全度水準）を記載する。

6. その他

<記載例>

- ・開発過程での安全性チェックのためのメカニズムに関する要求。

その他特記しておくべき事項がある場合には記載する。

(SW105) ソフトウェア要求仕様書

ソフトウェア要求仕様書には、ソフトウェア要求定義の作業で検討した、ソフトウェアとして実現が求められる機能要求事項、非機能要求事項や制約条件などを記載します。

■ このドキュメントを作成するサブタスク

SWP1 ソフトウェア要求定義

- 1.1 ソフトウェア要求仕様書の作成
- 1.1.5 ソフトウェア要求仕様書の作成

■ ドキュメントをまとめる際に参照する情報

(SW101) 制約条件リスト

(SW102) ソフトウェア機能要求リスト

(SW103) ソフトウェア非機能要求リスト

(SW104) 優先順位付けされたソフトウェア要求リスト

■ 記載項目例

- | | |
|----------------------|------------------|
| 1. 概要 | 6. 機能詳細 |
| 2. システム構成 | 7. インタフェース詳細 |
| 3. 機能概要 | 8. 性能・品質等非機能要求詳細 |
| 4. 制約条件 | 9. その他 |
| 5. ユースケースとユースケースシナリオ | |

■ このドキュメントを利用するサブタスク

SWP1 ソフトウェア要求定義

- 1.2 ソフトウェア要求仕様書の確認
 - 1.2.1 ソフトウェア要求仕様書の内部確認

SWP2 ソフトウェア・アーキテクチャ設計

- 2.1 ソフトウェア・アーキテクチャ設計書の作成
 - 2.1.1 設計条件の確認
 - 2.1.2 ソフトウェア構成の設計
 - 2.1.3 ソフトウェア全体の振る舞いの設計
 - 2.1.4 インタフェースの設計
- 2.2 ソフトウェア・アーキテクチャ設計の確認
 - 2.2.1 ソフトウェア・アーキテクチャ設計書の内部確認
- 2.3 ソフトウェア・アーキテクチャ設計の共同レビュー

- 2.3.1 ソフトウェア・アーキテクチャ設計書の共同レビュー

SWP3 ソフトウェア詳細設計

- 3.1 機能ユニット詳細設計書の作成
 - 3.1.1 プログラムユニット分割

SWP6 ソフトウェア総合テスト

- 6.1 ソフトウェア総合テストの準備
 - 6.1.1 ソフトウェア総合テスト仕様書の作成
 - 6.1.2 ソフトウェア総合テストの準備
 - 6.1.3 ソフトウェア総合テスト仕様書の内部確認
- 6.3 ソフトウェア総合テスト結果の確認
 - 6.3.1 ソフトウェア総合テスト結果の内部確認

SUP1 プロジェクトマネジメント

- 1.4 プロジェクト完了報告書の作成

■ (SW105) ソフトウェア要求仕様書の例

表紙

文書名

文書番号

| | |
|------|------|
| 承認 | 作成 |
| 承認者名 | 作成者名 |
| 承認日 | 作成日 |

発行日
発行部署

文書名を記載する。

文書の識別情報を記載する。

作成・承認の責任・確認を記載できる欄を設ける。

発行組織名、発行日を記載する。

改訂履歴

文書名

改訂履歴

| 項番 | 日付 | バージョン | 改訂内容 | 備考 |
|----|----|-------|------|----|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| | | ⋮ | | |
| | | ⋮ | | |

⋮

⋮

文書番号 頁番号 発行日・発行部署

改訂情報を記載できる欄を設ける。

目次

文書名

目次

| | |
|---------------------------|-----|
| 1. 概要..... | 頁番号 |
| 2. システム構成..... | 頁番号 |
| 3. 機能概要..... | 頁番号 |
| 4. 制約条件..... | 頁番号 |
| 5. ユースケースとユースケースシナリオ..... | 頁番号 |
| 6. インタフェース詳細..... | 頁番号 |
| 7. 性能・品質等非機能要求詳細..... | 頁番号 |
| 8. その他..... | 頁番号 |

文書番号 頁番号 発行日・発行部署

1. 概要

- <記載例>
- ・本書の目的
 - ・本書の位置づけ
 - ・対象ユーザ
 - ・記載範囲、記載内容など
 - ・参照しているドキュメントなど
 - ・定義（用語、略語など）

ドキュメントの目的、位置づけ、記載内容などの本書の概要および参照しているドキュメント名等を記載する。

2. システム構成

- <記載例>
- ・システム全体構成
 - システム構成要素の名称/基本機能
 - ソフトウェアに求められる要求仕様

ハードウェアを含めたシステム全体の構成とソフトウェアの位置づけ、およびソフトウェアを取り巻く関係/条件を記載する。システム要求仕様書やハードウェア仕様書等の関連資料より要求、条件などを整理して記載する。

3. 機能概要

- <記載例>
- ・システムとして実現・提供する機能のうち、ソフトウェアで実現する機能のリストと概要

機能概要を箇条書きでまとめる。詳細は6項に記載する。

4. 制約条件

- <記載例>
- ・ハードウェア構成とその制約
 - ・利用するOS、ミドルウェアなどの制約

制約となる条件を漏れなく記載する。

5. ユースケースとユースケースシナリオ

- <記載例>
- ・製品の利用シーンや利用コンテキストをもとにしたユースケースとユースケースシナリオ

システムを構成する機能ブロックごとにユーザとソフトウェアとしてのやりとりを考慮し、時系列的にその流れを整理して記載する。

6. 機能詳細

- <記載例>
- ・5項で整理したユースケースを実現する機能の詳細

各機能の詳細を機能ごとに記載する。

7. インタフェース詳細

- <記載例>
- ・製品が連携動作する周辺のソフトウェア、システムやハードウェアなどとのインタフェース

インタフェース対象ごとインタフェースの詳細を記載する。

8. 性能・品質等非機能要求詳細

<記載例>

- ・信頼性要求
 - システムの異常処理方式
 - システムの異常動作モードからの復帰手順や復帰方式
- ・使用性要求
 - ソフトウェアで実現する部分の操作性
 - ハードウェアで実現する部分との接点
- ・効率性要求の検討例
 - システムの実行性能（例：処理速度、起動時間、応答時間など）
 - リソース効率（例：メモリ容量、データサイズ）
- ・保守性要求
 - リモートメンテナンスなど保守の方式とその実現方法
- ・移植性要求
 - ソフトウェアユニットの独立性
- ・セキュリティ要求の検討例
 - 例：データ暗号化、ユーザ認証、ウィルス対策 など

機能で表現できない、性能・品質的な事項を記載する。

9. その他

<記載例>

- ・セキュリティ要求（例：データ暗号化、ユーザ認証、ウィルス対策など）
- ・相互運用性（例：通信プロトコルなど）
- ・外部インタフェース要求（例：連携ソフトウェアとの関数インタフェース、通信プロトコル、ユーザインタフェースなど）

その他特記しておくべき事項がある場合には記載する。

(SW205) ソフトウェア・アーキテクチャ設計書

ソフトウェア・アーキテクチャ設計書には、ソフトウェア・アーキテクチャ設計の作業で検討した、要求事項の実現方法(ソフトウェア構成、制御方式など)を記載します。

■ このドキュメントを作成するサブタスク

SWP2 ソフトウェア・アーキテクチャ設計

- 2.1 ソフトウェア・アーキテクチャ設計書の作成
- 2.1.6 ソフトウェア・アーキテクチャ設計書の作成

■ ドキュメントをまとめる際に参照する情報

(SW201) ソフトウェア構成設計書
(SW202) 機能ユニット設計書
(SW203) ソフトウェア動作設計書

(SW204) ソフトウェア・インタフェース設計書
・(性能試算資料)
・(メモリ使用試算資料)

■ 記載項目例

- | | |
|------------------|----------------|
| 1. 概要 | 4.2 ソフトウェア制御方式 |
| 2. システム構成 | 4.3 性能見積 |
| 3. ソフトウェア構成 | 5. 機能ユニット詳細 |
| 4. 制御方式 | 6. その他 |
| 4.1 メモリー構成/レイアウト | |

■ このドキュメントを利用するサブタスク

SWP2 ソフトウェア・アーキテクチャ設計

- 2.2 ソフトウェア・アーキテクチャ設計の確認
- 2.2.1 ソフトウェア・アーキテクチャ設計書の内部確認
- 2.3 ソフトウェア・アーキテクチャ設計の共同レビュー
- 2.3.1 ソフトウェア・アーキテクチャ設計書の共同レビュー

SWP3 ソフトウェア詳細設計

- 3.1 機能ユニット詳細設計書の作成
- 3.1.1 プログラムユニット分割
- 3.1.2 プログラムユニット設計
- 3.1.3 インタフェースの詳細化
- 3.1.4 メモリ量の見積もり
- 3.2 ソフトウェア詳細設計の確認

3.2.1 ソフトウェア詳細設計書の内部確認

- 3.3 ハードウェア仕様との整合性の確認
- 3.3.1 ハードウェア仕様との整合性の確認

SWP5 ソフトウェア結合テスト

- 5.1 ソフトウェア結合テストの準備
- 5.1.1 ソフトウェア結合の準備
- 5.1.2 ソフトウェア結合テストの準備
- 5.2 ソフトウェア結合テストの実施
- 5.2.1 ソフトウェア結合
- 5.3 ソフトウェア結合テスト結果の確認
- 5.3.1 ソフトウェア結合テスト結果の内部確認

■ (SW205) ソフトウェア・アーキテクチャ設計書の例

表紙

文書名

文書名を記載する。

文書番号

文書の識別情報を記載する。

| | |
|------|------|
| 承認 | 作成 |
| 承認者名 | 作成者名 |
| 承認日 | 作成日 |

作成・承認の責任・確認を記載できる欄を設ける。

発行日
発行部署

発行組織名、発行日を記載する。

改訂履歴

文書名

改訂履歴

| 項番 | 日付 | バージョン | 改訂内容 | 備考 |
|----|----|-------|------|----|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| | | ⋮ | | |
| | | ⋮ | | |

改訂情報を記載できる欄を設ける。

文書番号
頁番号
発行日・発行部署

目次

文書名

目次

| | |
|-----------------------|-----|
| 1. 概要..... | 頁番号 |
| 2. システム構成..... | 頁番号 |
| 3. ソフトウェア構成..... | 頁番号 |
| 4. 制御方式..... | 頁番号 |
| 4.1 メモリー構成/レイアウト..... | 頁番号 |
| 4.2 ソフトウェア制御方式..... | 頁番号 |
| 4.3 性能見積..... | 頁番号 |
| 5. 機能ユニット詳細..... | 頁番号 |
| 6. システムで扱うデータ..... | 頁番号 |
| 7. 例外一覧..... | 頁番号 |
| 8. その他..... | 頁番号 |

文書番号
頁番号
発行日・発行部署

| 本文① | 文書名 | |
|--|--|----------|
| <h3>1. 概要</h3> <p><記載例></p> <ul style="list-style-type: none"> ・本書の目的 ・本書の位置づけ ・対象ユーザ ・記載範囲、記載内容など ・参照しているドキュメントなど ・定義（用語、略語など） | <p>ドキュメントの目的、位置づけ、記載内容などの本書の概要および参照しているドキュメント名等を記載する。</p> | |
| <h3>2. システム構成</h3> <p><記載例></p> <ul style="list-style-type: none"> ・システム全体構成 <ul style="list-style-type: none"> - システム構成要素の名称/基本機能 - ソフトウェアに関する外部要素の基本仕様 ・システムを構成する主たるソフトウェア機能（要素） <p>例：MPU名称、接続I/O種、ハード割込みレベルとI/Oの対応、メモリー種類とサイズ、通信プロトコル、入出力データ形式、各種性能、外部媒体種/制御方式など</p> | <p>ハードウェアを含めたシステム全体の構成とソフトウェアの位置づけ、およびソフトウェアを取り巻く関係/条件を記載する。 システム・アーキテクチャ設計書、ハードウェア仕様書等の関連資料より要点、条件などを整理して記載する。</p> | |
| <h3>3. ソフトウェア構成</h3> <p><記載例></p> <ul style="list-style-type: none"> ・ソフトウェア全体構成 <ul style="list-style-type: none"> - ソフトウェアを構成する機能ユニットの名称/基本機能/動作形態（動作形態とは、タスク/非タスク、タスクプライオリティ、システム間動作/ユーザ空間動作、常駐/非常駐 共通関数、保守用など） - ソフトウェア機能ユニット間のインタフェース <p>例：各機能ユニット間の制御、データを静的に記載したもので、必要に応じて、OS、メモリ、外部記憶媒体、ハードウェア機構を併記する。</p> | <p>開発ソフトウェアの構成および基本的関係を記載する。</p> | |
| <h3>4. 制御方式</h3> | <p>ソフトウェア要求仕様書（SW105）に記載された要求をシステム全体（ハードウェア、オペレータ、外部記憶媒体、メモリなど）を通してどのように実現するかを、機能ユニットを中心に記載する。</p> | |
| <h4>4.1 メモリー構成/レイアウト</h4> <p><記載例></p> <ul style="list-style-type: none"> ・メモリアドレス空間を基に、ROM/RAMの種類およびSRAM/キャッシュ割り当て域、ソフトウェア用領域、ハードウェアレジスタ領域位置、サイズなど ・ソフトウェア領域の構成（コード域、データ域、OS使用域） ・ソフトウェア共通情報制御領域の構成および、テーブル/バッファのリンク方式など | <p>メモリ構成、種類、サイズ等、システム・アーキテクチャ設計書または、ハードウェア仕様書等から明確にするとともにソフトウェア領域をどのように使用するかを記載する。 （ただし、高機能32ビットMPU、VM機能付きOS、SoC等の場合は、固定化あるいはバーチャル化されており、ソフトウェアでの必要な事項に関して明確化すれば良い。）</p> | |
| <h4>4.2 ソフトウェア制御方式</h4> <p><記載例></p> <ul style="list-style-type: none"> ・リカバリ方式：エラー発生から再立ち上げまでの制御の流れ ・性能実現方式：要求される機器性能を実現する並行処理制御 ・他国語対応：複数の外国語に対応するための変化方式 ・出荷後のソフトウェアアップデート方式：フィールド出荷後のソフトウェア更新方式 ・障害調査データ取得方式：動作トレースなどの格納から取得/解析までの流れなど | <p>機能ユニット間の処理シーケンス図等を利用し、信号、コマンド、データ等の検知、制御、変換、処理、通知、格納等を、ケースごとに記載する。また、要求としては明に記載されないが、開発者の共有事項として重要な事項（たとえば、OS動作、内部状態遷移）についても記載するのが望ましい。</p> | |
| <h4>4.3 性能見積</h4> <p><記載例></p> <ul style="list-style-type: none"> ・必要性能/時間：x x (ms) ・性能実現方式：対象条件とその処理シーケンス ・各処理時間見積もり：各処理要素の見積もりおよび根拠 | <p>定量的に性能を見積もる。</p> | |
| 文書番号 | 頁番号 | 発行日・発行部署 |

注：ここでのソフトウェアとは、開発対象のソフトウェアおよび、利用ソフトウェアを含んでいます。

5. 機能ユニット詳細

<記載例>

- ・XX機能ユニット詳細
- ・構成、機能、入出力インタフェース、処理方式、共通領域など。

詳細設計が可能になるまで、各機能ユニットの構成/機能、インタフェース等を詳細化/具体化する。詳細化の程度は、規模、要員数を考慮して記載する。詳細設計者が詳細化できる情報が網羅されていれば良い。

6. システムで扱うデータ

<記載例>

- ・排他処理が必要な共通データなど

共通データなどシステム全体で共通に扱うデータの使用目的、更新/参照元などを記載する。

7. 例外一覧

<記載例>

- ・異常/例外情報一覧
- 異常/例外コード、メッセージ、意味、対策 など

異常/例外時のコード、メッセージ、意味、対策などを一覧表として記載する。

8. その他

<記載例>

- ・エラー/異常情報一覧
- エラー/異常のコード、メッセージ、意味、対策など
- ・共通制御情報領域詳細
- 共通領域内の各情報域の名称、構成、形式、サイズ、初期値、アクセス制限など

ソフトウェア全体を通して、開発グループ全員で情報共有すべき事柄を記載する。

(SW305) ソフトウェア詳細設計書

ソフトウェア詳細設計書には、ソフトウェア詳細設計の作業で検討した、プログラムユニットの構成、詳細処理およびプログラムユニット間のインタフェースなどを実装可能なレベルで記載します。

■ このドキュメントを作成するサブタスク

SWP3 ソフトウェア詳細設計

- 3.1 機能ユニット詳細設計書の作成
 - 3.1.5 ソフトウェア詳細設計書の作成

■ ドキュメントをまとめる際に参照する情報

(SW301) プログラムユニット機能/構成設計書
(SW302) プログラムユニット設計書
(SW303) プログラムユニット・インタフェース設計書

(SW304) メモリ使用量 (メモ)
(SW309) ハードウェア仕様との整合性確認結果レポート (指摘事項反映の場合)

■ 記載項目例

- 1. 概要
- 2. プログラムユニット機能/構成設計書
 - 2.1 プログラムユニット一覧表
 - 2.2 プログラムユニット構成図
- 3. プログラムユニット設計書
 - 3.1 プログラムユニット詳細処理
 - 3.2 状態管理
 - 3.3 リソース定義
- 3.4 ハードウェア制御方法
- 3.5 システム初期化処理
- 3.6 共通定義
- 4. プログラムユニット・インタフェース設計書
 - 4.1 シーケンス図
 - 4.2 インタフェース詳細
- 5. メモリ使用量

■ このドキュメントを利用するサブタスク

SWP3 ソフトウェア詳細設計

- 3.2 ソフトウェア詳細設計の確認
 - 3.2.1 ソフトウェア詳細設計書の内部確認
- 3.3 ハードウェア仕様との整合性の確認
 - 3.3.1 ハードウェア仕様との整合性の確認

SWP4 実装および単体テスト

- 4.1 実装および単体テストの準備
 - 4.1.2 単体テストの準備
- 4.2 実装および単体テストの実施
 - 4.2.1 プログラムユニットの実装
- 4.3 実装および単体テスト結果の確認
 - 4.3.1 ソースコードの確認
 - 4.3.2 単体テスト結果の内部確認

■ (SW305) ソフトウェア詳細設計書の例

表紙

文書名

文書番号

| | |
|------|------|
| 承認 | 作成 |
| 承認者名 | 作成者名 |
| 承認日 | 作成日 |

発行日
発行部署

文書名

文書の識別情報を記載する。

作成・承認の責任・確認を記載できる欄を設ける。

発行組織名、発行日を記載する。

文書名を記載する。

文書の識別情報を記載する。

作成・承認の責任・確認を記載できる欄を設ける。

発行組織名、発行日を記載する。

改訂履歴

文書名

改訂履歴

| 項番 | 日付 | バージョン | 改訂内容 | 備考 |
|----|----|-------|------|----|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| | | | ⋮ | |
| | | | ⋮ | |
| | | | ⋮ | |

文書番号
頁番号
発行日・発行部署

改訂情報を記載できる欄を設ける。

目次

文書名

目次

| | |
|------------------------------|-----|
| 1. 概要..... | 頁番号 |
| 2. プログラムユニット機能/構成設計書..... | 頁番号 |
| 2.1 プログラムユニット一覧表..... | 頁番号 |
| 2.2 プログラムユニット構成図..... | 頁番号 |
| 3. プログラムユニット設計書..... | 頁番号 |
| 3.1 プログラムユニット詳細処理..... | 頁番号 |
| 3.2 状態管理..... | 頁番号 |
| 3.3 リソース定義..... | 頁番号 |
| 3.4 ハードウェア制御方法..... | 頁番号 |
| 3.5 システム初期化処理..... | 頁番号 |
| 3.6 共通定義..... | 頁番号 |
| 4. プログラムユニット・インタフェース設計書..... | 頁番号 |
| 4.1 シーケンス図..... | 頁番号 |
| 4.2 インタフェース詳細..... | 頁番号 |
| 5. メモリ使用量..... | 頁番号 |

文書番号
頁番号
発行日・発行部署

1. 概要

<記載例>

- ・本書の目的
- ・本書の位置づけ
- ・対象ユーザ
- ・記載範囲、記載内容など
- ・参照しているドキュメントなど
- ・定義（用語、略語など）

ドキュメントの目的、位置づけ、記載内容などの本書の概要および参照しているドキュメント名等を記載する。

2. プログラムユニット機能/構成設計書

2.1 プログラムユニット一覧表

<記載例>

- ・プログラムユニット名
- ・機能概要

全プログラムユニットの名称と機能概要を一覧化する。

2.2 プログラムユニット構成図

<記載例>

- ・プログラムユニットの名称
- ・プログラムユニット間の相互関係（起動方法、インタフェース概要など）

プログラムユニットの構成と相互関係を図示する。

3. プログラムユニット設計書

3.1 プログラムユニット詳細処理

<記載例>

- ・プログラムユニット名
- ・引数
- ・戻り値
- ・処理内容（ハードウェア制御方法、タイミング、エラー処理、利用するOSシステムコールや汎用ライブラリの引数値、高速化が必要な場合はその条件と手法など）
- ・制約事項（処理時間、割り込みなど）
- ・備考（参照する設計書など）

プログラムユニットの処理内容を記載する。不具合を解析する際に利用する機能もプログラムユニットとして処理内容を記載する。実装に必要なすべての情報を記載することが望ましい。

3.2 状態管理

<記載例>

状態遷移表

| | 状態A | 状態B | 状態C | 状態D |
|-------|---------------|-----|-----|-----|
| イベントA | | | | |
| イベントB | 遷移先の状態名を記載する。 | | | |
| イベントC | | | | |

ソフトウェアの状態と、その状態がイベントを起因として遷移する状態を記載する（状態遷移表）。

3.3 リソース定義

<記載例>

- ・リソース名
- ・用途
- ・構造
- ・値とその意味
- ・サイズ
- ・取得・解放

共通に使用するリソース（メモリ、DBなど）の詳細内容を記載する。

3.4 ハードウェア制御方法

<記載例>
 ・ハードウェアへの書き込み/読み出し方法
 ・設定値
 ・アドレス/ポート番号
 ・制約事項（順序、禁止事項など）

個々のハードウェアに対する具体的な制御方法を記載する。

3.5 システム初期化処理

<記載例>
 ・初期化する順序
 ・ハードウェア初期値
 ・ベクターアドレス/割り込み
 ・リソース初期値（メモリ、DB など）
 ・OSの初期化
 ・汎用ライブラリの初期化
 ・外部機器の初期化

ハードウェアおよびソフトウェアの初期化方法を記載する。

3.6 共通定義

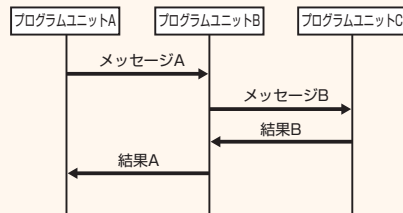
<記載例>
 ・エラー値
 ・コンパイル条件
 ・リソースのサイズ（メモリ、DBなど）

ソフトウェア全体を通して共通に使用する値を記載する。

4. プログラムユニット・インタフェース設計書

4.1 シーケンス図

<記載例>



プログラムユニット間の相互作用を時系列に表す。縦軸は時間を表し下に向かって進む。

4.2 インタフェース詳細

<記載例>
 ・呼び出し方法（値、意味）
 ・結果（値、意味）
 ・データフォーマット（構造、値、意味、サイズ）
 ・制約事項

インタフェースの詳細内容を記載する。

5. メモリ使用量

<記載例>
 ・ROM（各ユニット名称+使用量の一覧）
 ・RAM（リソース名称+使用量の一覧）
 ・スタックエリア（各ユニット名称+使用量の一覧）

メモリ種別ごとに使用量を記載する。使用量が分析できるように、各ユニットやリソースごとに記載することが望ましい。

(SW401、SW501、SW601、SY301、SY401、SA201)テスト仕様書 (SW406、SW506、SW606、SY306、SY406、SA205)テスト報告書

テスト仕様書には、テスト項目、テストデータなどを記載します。

テスト報告書には、テスト結果の評価を記載します。評価には、とらえられた問題と対応(問い合わせ件数とその回答、不具合件数、ソフトウェア変更有無の必要性など)を記載します。

■ このドキュメントを作成するサブタスク

SWP4 実装および単体テスト

- 4.1 実装および単体テストの準備
 - 4.1.2 単体テストの準備
- 4.2 実装および単体テストの実施
 - 4.2.3 単体テスト結果の確認

SWP5 ソフトウェア結合テスト

- 5.1 ソフトウェア結合テストの準備
 - 5.1.2 ソフトウェア結合テストの準備
- 5.2 ソフトウェア結合テストの実施
 - 5.2.3 ソフトウェア結合テスト結果の確認

SWP6 ソフトウェア総合テスト

- 6.1 ソフトウェア総合テストの準備
 - 6.1.1 ソフトウェア総合テスト仕様書の作成
- 6.2 ソフトウェア総合テストの実施
 - 6.2.2 ソフトウェア総合テスト結果の確認

SYP3 システム結合テスト

- 3.1 システム結合テストの準備
 - 3.1.2 システム結合テストの準備
- 3.2 システム結合テストの実施
 - 3.2.2 システム結合テスト結果の確認

SYP4 システムテスト

- 4.1 システムテストの準備
 - 4.1.1 システムテスト仕様書の作成
- 4.2 システムテストの実施
 - 4.2.2 システムテスト結果の確認

SAP2 安全性テスト

- 2.1 安全性テストの準備
 - 2.1.1 安全性テストの準備
- 2.2 安全性テストの実施
 - 2.2.2 安全性テスト結果の確認

■ ドキュメントをまとめる際に参照する情報

単体テスト

- (SW305)ソフトウェア詳細設計書
- (SW405)単体テスト結果(メモ)
- (SU1002)ソフトウェア開発環境
- (SU601)不具合管理票(修正確認の場合)

ソフトウェア結合テスト

- (SW205)ソフトウェア・アーキテクチャ設計書
- (SU1002)ソフトウェア開発環境
- (SW505)ソフトウェア結合テスト結果
- (SU601)不具合管理票(修正確認の場合)

ソフトウェア総合テスト

- (SW105)ソフトウェア要求仕様書
- テスト仕様書作成に必要な情報(成果物)
- 参照情報
 - 過去のソフトウェア総合テスト仕様書
 - (SW504)実行可能形式化された最終のソースコード
 - (SW605)ソフトウェア総合テスト結果
 - (SU601)不具合管理票(修正確認の場合)

システム結合テスト

- (SY205) システム・アーキテクチャ設計書
- (SY305) システム結合テスト結果
- (SU601) 不具合管理票 (修正確認の場合)

システムテスト

- (SY106) システム要求仕様書
- 製品マニュアル
- テスト仕様書作成に必要な情報 (成果物)
- 参照情報
- 過去のシステムテスト仕様書

テスト対象のシステム

- (SY405) システムテスト結果
- (SU601) 不具合管理票 (修正確認の場合)

安全性テスト

- (SA104) 安全要求仕様書
- 参照情報
- 過去の安全性テスト仕様書

テスト対象のシステム

- (SA204) 安全性テスト結果
- (SU601) 不具合管理票 (修正確認の場合)

■ 記載項目例

- 概要
 - テスト概要
 - 2.1 テスト方針
 - 2.2 開発フェーズおよびテスト期間
 - 2.3 対象範囲
 - 評価設備・環境
 - 3.1 システム構成
 - 3.2 テスト環境・設備
 - テスト項目
 - テストデータ
 - テスト手順
 - 評価結果*
 - 7.1 テスト結果概要*
 - 7.2 テスト結果詳細*
- * テスト報告書の場合に記載する項目

■ このドキュメントを利用するサブタスク

- | | |
|--------------------------|---------------------------|
| SWP4 実装および単体テスト | 6.1.2 ソフトウェア総合テストの準備 |
| 4.2 実装および単体テストの実施 | 6.1.3 ソフトウェア総合テスト仕様書の内部確認 |
| 4.2.2 単体テストの実施 | |
| 4.2.3 単体テスト結果の確認 | |
| 4.3 実装および単体テスト結果の確認 | 6.2 ソフトウェア総合テストの実施 |
| 4.3.2 単体テスト結果の内部確認 | 6.2.1 ソフトウェア総合テストの実施 |
| SWP5 ソフトウェア結合テスト | 6.2.2 ソフトウェア総合テスト結果の確認 |
| 5.2 ソフトウェア結合テストの実施 | 6.3 ソフトウェア総合テスト結果の確認 |
| 5.2.2 ソフトウェア結合テストの実施 | 6.3.1 ソフトウェア総合テスト結果の内部確認 |
| 5.2.3 ソフトウェア結合テスト結果の確認 | 6.4 ソフトウェア開発の完了確認 |
| 5.3 ソフトウェア結合テスト結果の確認 | 6.4.1 ソフトウェア開発の完了確認 |
| 5.3.1 ソフトウェア結合テスト結果の内部確認 | |
| SWP6 ソフトウェア総合テスト | SYP3 システム結合テスト |
| 6.1 ソフトウェア総合テストの準備 | 3.2 システム結合テストの実施 |

- 3.2.1 システム結合テストの実施
- 3.2.2 システム結合テスト結果の
確認
- 3.3 システム結合テスト結果の確認
 - 3.3.1 システム結合テスト結果の
内部確認
- SYP4 システムテスト**
 - 4.1 システムテストの準備
 - 4.1.2 システムテストの準備
 - 4.1.3 システムテスト仕様書の内
部確認
 - 4.2 システムテストの実施
 - 4.2.1 システムテストの実施
 - 4.2.2 システムテスト結果の確認
 - 4.3 システムテスト結果の確認
- 4.3.1 システムテスト結果の内部
確認
- 4.4 システム開発の完了確認
 - 4.4.1 システム開発の完了確認
- SAP2 安全性テスト**
 - 2.2 安全性テストの実施
 - 2.2.1 安全性テストの実施
 - 2.2.2 安全性テスト結果の確認
 - 2.3 安全性テストの確認
 - 2.3.1 安全性テスト結果の内部確
認
- SUP1 プロジェクトマネジメント**
 - 1.4 プロジェクト完了報告書の作成

■ (SW401、SW501、SW601、SW406、SW506、SW606、SY301、SY401、SY306、SY406、SA201、SA205) テスト仕様書/テスト報告書

表紙

文書名

文書番号

| | |
|------|------|
| 承認 | 作成 |
| 承認者名 | 作成者名 |
| 承認日 | 作成日 |

発行日
発行部署

文書名を記載する。

文書の識別情報を記載する。

作成・承認の責任・確認を記載できる欄を設ける。

発行組織名、発行日を記載する。

改訂履歴

文書名

改訂履歴

| 項番 | 日付 | バージョン | 改訂内容 | 備考 |
|----|----|-------|------|----|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| | | | ⋮ | |
| | | | ⋮ | |
| | | | ⋮ | |

文書番号
頁番号
発行日・発行部署

改訂情報を記載できる欄を設ける。

目次

文書名

目次

| | |
|-------------------------|-----|
| 1. 概要..... | 頁番号 |
| 2. テスト概要..... | 頁番号 |
| 2.1 テスト方針..... | 頁番号 |
| 2.2 開発フェーズおよびテスト期間..... | 頁番号 |
| 2.3 対象範囲..... | 頁番号 |
| 3. 評価設備・環境..... | 頁番号 |
| 3.1 システム構成..... | 頁番号 |
| 3.2 テスト環境・設備..... | 頁番号 |
| 4. テスト概要・結果概要..... | 頁番号 |
| 5. 添付資料..... | 頁番号 |
| 5.1 テスト項目詳細..... | 頁番号 |
| 5.2 バグカーブ..... | 頁番号 |
| 5.3 その他..... | 頁番号 |

文書番号
頁番号
発行日・発行部署

1. 概要

<記載例>

- ・本書の目的
- ・本書の位置づけ
- ・対象ユーザ
- ・記載範囲、記載内容など
- ・参照しているドキュメントなど
- ・定義（用語、略語など）

ドキュメントの目的、位置づけ、記載内容などの本書の概要および参照しているドキュメント名等を記載する。

2. テスト概要

2.1 テスト方針

テストに対する基本的な考え方を記載する。

2.2 開発フェーズおよびテスト期間

<記載例>

| 開発フェーズ | SWバージョン | HWバージョン | 検査開始日 | 検査終了日 | 備考 |
|--------|---------|---------|-------|-------|----|
| | | | | | |

テストの対象となるソフトウェアの開発フェーズ、ソフトウェアバージョン、ハードウェアバージョンおよびテスト期間を記載する。

2.3 対象範囲

<記載例>

| ソフトウェア要求仕様書 | ソフトウェア・アーキテクチャ設計書 | 検査機能 | 検査状況 |
|-------------|-------------------|------|------|
| | | | |
| | | | |

本仕様書にて実施されるソフトウェアのテスト対象範囲（機能）を記載する。（ソフトウェア要求仕様書やソフトウェア・アーキテクチャ設計書とテスト項目との対応づけをする）

⋮

3. 評価設備・環境

3.1 システム構成

3.2 テスト環境・設備

<記載例>

| 設備 | メーカ | 型番 | シリアル番号 | 台数 | 用途 |
|----|-----|----|--------|----|----|
| | | | | | |
| | | | | | |

⋮

4. テスト概要／結果概要

<記載例>

| 分類ID | 分類 | テスト項目数 | 検査開始日 | 検査終了日 | 未消化項目数 | 不具合数 | 未対応不具合数 | 備考 |
|------|----|--------|-------|-------|--------|------|---------|----|
| | | | | | | | | |
| | | | | | | | | |

⋮

5. 添付資料

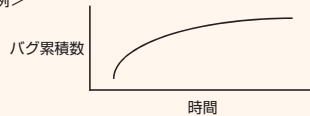
5.1 テスト項目詳細

<記載例>

(次ページに掲載)

5.2 バグカーブ

<記載例>



5.3 その他

<記載例>

- ・テスト項目の分類基準（正常値入力、異常値入力、過去不具合、負荷テスト、耐久テストなど）
- ・テスト結果判定基準（OK、NG、OB、No Check）
- ・テスト結果を客観的に証明する添付資料（波形データなど）

文書番号

頁番号

発行日・発行部署

システム構成とテスト対象の（ソフトウェア）関係を図等を使用し分かりやすく記載する。

テスト実施時に使用するテスト環境（ソフトウェア総合テストの場合にはハードウェア環境など）を図等を使用して分かりやすく記載する。
（テストにて使用するツールとの接続方法も含む）
ソフトウェア総合テストの場合には、実施時に使用する設備の情報（バージョン情報など）と目的を記載する。

テスト内容と期待する結果を記載する。各テスト項目には管理しやすいように番号を振り、テストの実施日やテスト結果を記載する。

テスト項目ごとにテストデータ、期待する出力、結果などを記載する。

テスト工程におけるバグ収束状況を確認するために、縦軸にバグ累積数、横軸に時間をとってその関係をグラフで記載する。
<関連情報>Related Techniques (P73)

ソフトウェア全体を通して、開発グループ全員で情報共有すべき事柄やテスト証跡などを記載する。

(SY107、SY206、SA105、SW106、SW206、SW306、SW408、SW507、SW607、SY307、SY404、SY407、SA206)内部確認レポート

内部確認レポートには、仕様書、設計書、テスト結果の内部確認作業で発見された問題点の対策、対応担当者および修正確認などを記載します。

■ このドキュメントを作成するサブタスク

SYP1 システム要求定義

- 1.2 システム要求仕様の確認
 - 1.2.1 システム要求仕様書の内部確認

SYP2 システム・アーキテクチャ設計

- 2.2 システム・アーキテクチャ設計の確認
 - 2.2.1 システム・アーキテクチャ設計書の内部確認

SAP1 安全性要求定義

- 1.2 安全要求仕様の確認
 - 1.2.1 安全要求仕様書の内部確認

SWP1 ソフトウェア要求定義

- 1.2 ソフトウェア要求仕様書の確認
 - 1.2.1 ソフトウェア要求仕様書の内部確認

SWP2 ソフトウェア・アーキテクチャ設計

- 2.2 ソフトウェア・アーキテクチャ設計の確認
 - 2.2.1 ソフトウェア・アーキテクチャ設計書の内部確認

SWP3 ソフトウェア詳細設計

- 3.2 ソフトウェア詳細設計の確認
 - 3.2.1 ソフトウェア詳細設計書の内部確認

SWP4 実装および単体テスト

- 4.3 実装および単体テスト結果の確認
 - 4.3.2 単体テスト結果の内部確認

SWP5 ソフトウェア結合テスト

- 5.3 ソフトウェア結合テスト結果の確認
 - 5.3.1 ソフトウェア結合テスト結果の内部確認

SWP6 ソフトウェア総合テスト

- 6.1 ソフトウェア総合テストの準備
 - 6.1.3 ソフトウェア総合テスト仕様書の内部確認
- 6.3 ソフトウェア総合テスト結果の確認
 - 6.3.1 ソフトウェア総合テスト結果の内部確認

SYP3 システム結合テスト

- 3.3 システム結合テスト結果の確認
 - 3.3.1 システム結合テスト結果の内部確認

SYP4 システムテスト

- 4.1 システムテストの準備
 - 4.1.3 システムテスト仕様書の内部確認
- 4.3 システムテスト結果の確認
 - 4.3.1 システムテスト結果の内部確認

SAP2 安全性テスト

- 2.3 安全性テスト結果の確認
 - 2.3.1 安全性テスト結果の内部確認

■ ドキュメントをまとめる際に参照する情報

システム要求定義

(SY106) システム要求仕様書

システム・アーキテクチャ設計

(SY106) システム要求仕様書

(SA104) 安全要求仕様書

(SY205) システム・アーキテクチャ設計書

(SY201) システム構成図(機能ブロック図)

(SY203) システム動作設計書

(SY204) システムインタフェース設計書

ソフトウェア要求定義

製品企画書

(SY106) システム要求仕様書

(SY205) システム・アーキテクチャ設計書

(SA104) 安全要求仕様書

(SW105) ソフトウェア要求仕様書

ソフトウェア・アーキテクチャ設計

(SW105) ソフトウェア要求仕様書

(SW205) ソフトウェア・アーキテクチャ設計書

(SW201) ソフトウェア構成設計書

(SW202) 機能ユニット設計書

(SW203) ソフトウェア動作設計書

(SW204) ソフトウェア・インタフェース設計書

ソフトウェア詳細設計

(SW205) ソフトウェア・アーキテクチャ設計書

(SW305) ソフトウェア詳細設計書

実装および単体テスト

(SW401) 単体テスト仕様書

(SW403) 内部確認メモ(単体テスト仕様)

(SW406) 単体テスト報告書

(SW407) 内部確認メモ(ソースコード)

(SU601) 不具合管理票

ソフトウェア結合テスト

(SW501) ソフトウェア結合テスト仕様書

(SW503) 内部確認メモ(ソフトウェア結合テスト仕様)

(SW506) ソフトウェア結合テスト報告書

(SU601) 不具合管理票

ソフトウェア総合テスト

(SW601) ソフトウェア総合テスト仕様書

(SW604) 内部確認メモ(ソフトウェア総合テスト仕様)

(SW606) ソフトウェア総合テスト報告書

(SU601) 不具合管理票

システム結合テスト

(SY301) システム結合テスト仕様書

(SY303) 内部確認メモ(システム結合テスト仕様)

(SY306) システム結合テスト報告書

(SU601) 不具合管理票

システムテスト

(SY401) システムテスト仕様書

(SY404) 内部確認メモ(システムテスト仕様)

(SY406) システムテスト報告書

(SU601) 不具合管理票

安全性テスト

(SA201) 安全性テスト仕様書

(SA203) 内部確認メモ(安全性テスト仕様)

(SA205) 安全性テスト報告書

(SU601) 不具合管理票

■ 記載項目例

- ▶ プロジェクト名
- ▶ 評価対象
- ▶ 日時、場所
- ▶ 出席者
- ▶ 評価資料
- ▶ 問題点、対策、担当、期限、修正確認

■ このドキュメントを利用するサブタスク

SYP2 システム・アーキテクチャ設計

- 2.3 システム・アーキテクチャ設計の共同レビュー
 - 2.3.1 システム・アーキテクチャ設計書の共同レビュー

SWP2 ソフトウェア・アーキテクチャ設計

- 2.3 ソフトウェア・アーキテクチャ設計の共同レビュー
 - 2.3.1 ソフトウェア・アーキテクチャ設計書の共同レビュー

SWP6 ソフトウェア総合テスト

- 6.4 ソフトウェア開発の完了確認
 - 6.4.1 ソフトウェア開発の完了確認

SYP4 システムテスト

- 4.4 システム開発の完了確認
 - 4.4.1 システム開発の完了確認

SUP1 プロジェクトマネジメント

- 1.4 プロジェクト完了報告書の作成

■(SY107、SY206、SA104、SW106、SW206、SW306、SW408、SW507、SW607、SY307、SY404、SY407、SA206) 内部確認レポート

| 内部確認レポート | | | | | |
|----------|-----|----|----------|------|-----|
| プロジェクト名 | | | | 管理No | |
| 評価対象 | | | | 部署 | |
| 日時 | 年 | 月 | 日 () | 確認 | 確認 |
| 場所 | | | | | 作成 |
| 出席者 | | | | / / | / / |
| 確認資料 | | | | | / / |
| No | 問題点 | 対策 | 担当 期限 | 修正確認 | |
| | | | / / | / / | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

プロジェクト名、日時、場所、出席者、管理情報を記載する。

作成・承認の責任・確認を記載できる欄を設ける。

内部確認の対象資料を記載する。

問題点とその対策、対策を実施する担当者と期限、修正確認を記載する。

Related Standards

ISO/IEC 15288 Systems engineering – System life cycle processes JIS X 0170 システムライフサイクルプロセス

成立・改訂年： ISO/IEC 2002、JIS 2004

目的： (IT)システムのライフサイクルプロセス群の概念の標準化

対象分野： (IT)システム開発プロセス

URL： <http://www.iso.org/>

国内入手先： 日本規格協会

概要：

複数のソフトウェアやハードウェアなどが関連して動作するシステムを対象として、その開発プロセスを定義した規格です。「ISO/IEC 12207 ソフトウェアライフサイクルプロセス」をシステムレベルにまで拡大したものと位置づけられます。

この規格は、システムの開発を構想する段階から、そのシステム利用を停止する段階に至るまでのライフサイクルに従って整理され、プロセスの集合およびそのプロセスに関連する用語を定義しています。

また、この規格ではプロセスを「目的」「成果」および「それを達成するために必要な行動」に分けて記述しています。

プロセス記述

－プロセス名称

－プロセスの目的

－プロセスの成果

－それを達成するために必要な行動

(SU801) 共同レビュー記録

共同レビュー記録には、レビューでの指摘事項、対策、対応担当者および修正確認などを記載します。

■ このドキュメントを作成するサブタスク

- SYP2 システム・アーキテクチャ設計
 - 2.3 システム・アーキテクチャ設計の共同レビュー
 - 2.3.1 システム・アーキテクチャ設計書の共同レビュー
- SWP2 ソフトウェア・アーキテクチャ設計
 - 2.3 ソフトウェア・アーキテクチャ設計の共同レビュー
 - 2.3.1 ソフトウェア・アーキテクチャ設計書の共同レビュー
- SWP6 ソフトウェア総合テスト
 - 6.4 ソフトウェア開発の完了確認
 - 6.4.1 ソフトウェア開発の完了確認
- SYP4 システムテスト
 - 4.4 システム開発の完了確認
 - 4.4.1 システム開発の完了確認
- SUP8 共同レビュー
 - 8.2 レビューの実施
 - 8.3 レビュー結果の確認とフォロー

■ ドキュメントをまとめる際に参照する情報

- | | |
|------------------------------------|-------------------------------|
| システム・アーキテクチャ設計 | (SW607) 内部確認レポート(ソフトウェア総合テスト) |
| (SY205) システム・アーキテクチャ設計書 | (SU101) プロジェクト計画書 |
| (SY106) システム要求仕様書 | (SU601) 不具合管理票 |
| (SA104) 安全要求仕様書 | |
| ソフトウェア・アーキテクチャ設計 | システムテスト |
| (SW205) ソフトウェア・アーキテクチャ設計書 | (SY106) システム要求仕様書 |
| (SW206) 内部確認レポート(ソフトウェア・アーキテクチャ設計) | (SY401) システムテスト仕様書 |
| 参照情報 | (SY406) システムテスト報告書 |
| (SY106) システム要求仕様書 | (SY407) 内部確認レポート(システムテスト) |
| (SY205) システム・アーキテクチャ設計書 | (SA104) 安全要求仕様書 |
| (SW105) ソフトウェア要求仕様書 | (SA201) 安全性テスト仕様書 |
| ソフトウェア総合テスト | (SA206) 内部確認レポート(安全性テスト) |
| (SW106) ソフトウェア要求仕様書 | (SU101) プロジェクト計画書 |
| (SW601) ソフトウェア総合テスト仕様書 | (SU601) 不具合管理票 |
| (SW606) ソフトウェア総合テスト報告書 | |

■ 記載項目例

- ▶ プロジェクト名
- ▶ 評価対象
- ▶ 日時、場所
- ▶ 出席者
- ▶ レビュー資料
- ▶ 指摘事項、対策、担当、期限、修正確認

■ このドキュメントを利用するサブタスク

- SUP1 プロジェクトマネジメント
 - 1.4 プロジェクト完了報告書の作成
- SUP8 共同レビュー
 - 8.3 レビュー結果の確認とフォロー

(SU601) 不具合管理票

不具合管理票には、テストで発見された不具合の状況、原因・影響の分析結果および対応内容や方針を記載します。

各不具合管理票は不具合管理台帳 (SU602) で管理します。

■ このドキュメントを作成するサブタスク

- SWP4 実装および単体テスト**
 - 4.2 実装および単体テストの実施
 - 4.2.3 単体テスト結果の確認
- SWP5 ソフトウェア結合テスト**
 - 5.2 ソフトウェア結合テストの実施
 - 5.2.3 ソフトウェア結合テスト結果の確認
- SWP6 ソフトウェア総合テスト**
 - 6.2 ソフトウェア総合テストの実施
 - 6.2.2 ソフトウェア総合テスト結果の確認
- SAP2 安全性テスト**
 - 2.2 安全性テストの実施
 - 2.2.2 安全性テスト結果の確認
- SYP3 システム結合テスト**
 - 3.2 システム結合テストの実施
 - 3.2.2 システム結合テスト結果の確認
- SYP4 システムテスト**
 - 4.2 システムテストの実施
 - 4.2.2 システムテスト結果の確認
- SUP6 問題解決管理**
 - 6.1 問題の記録と原因分析
 - 6.2 影響分析と対策立案
 - 6.3 対策の実行
 - 6.4 対策結果の確認

■ ドキュメントをまとめる際に参照する情報

- | | |
|------------------------|------------------------|
| 単体テスト | (SW601) ソフトウェア総合テスト仕様書 |
| (SW401) 単体テスト仕様書 | (SW605) ソフトウェア総合テスト結果 |
| (SW405) 単体テスト結果 | 安全性テスト |
| ソフトウェア結合テスト | (SA201) 安全性テスト仕様書 |
| (SW501) ソフトウェア結合テスト仕様書 | (SA204) 安全性テスト結果 |
| (SW505) ソフトウェア結合テスト結果 | システム結合テスト |
| ソフトウェア総合テスト | (SY301) システム結合テスト仕様書 |

(SY305) システム結合テスト結果
システムテスト
(SY401) システムテスト仕様書

(SY405) システムテスト結果

■ 記載項目例

- ▶ プロジェクト名
- ▶ 管理番号
- ▶ 状況 (発見日時、不具合箇所、発見工程、現象など)
- ▶ 原因・影響
- ▶ 対応内容
- ▶ 対応方針 (優先度、期限など)

■ このドキュメントを利用するサブタスク

- | | |
|---|--|
| <p>SWP4 実装および単体テスト</p> <ul style="list-style-type: none">4.1 実装および単体テストの準備4.1.2 単体テストの準備4.2 実装および単体テストの実施4.2.1 プログラムユニットの実装4.2.2 単体テストの実施4.2.3 単体テスト結果の確認4.3 実装および単体テスト結果の確認4.3.2 単体テスト結果の内部確認 <p>SWP5 ソフトウェア結合テスト</p> <ul style="list-style-type: none">5.1 ソフトウェア結合テストの準備5.1.2 ソフトウェア結合テストの準備5.2 ソフトウェア結合テストの実施5.2.2 ソフトウェア結合テストの実施5.2.3 ソフトウェア結合テスト結果の確認5.3 ソフトウェア結合テスト結果の確認5.3.1 ソフトウェア結合テスト結果の内部確認 <p>SWP6 ソフトウェア総合テスト</p> <ul style="list-style-type: none">6.1 ソフトウェア総合テストの準備6.1.2 ソフトウェア総合テストの準備6.2 ソフトウェア総合テストの実施6.2.2 ソフトウェア総合テスト結果の確認 | <ul style="list-style-type: none">6.3 ソフトウェア総合テスト結果の確認6.3.1 ソフトウェア総合テスト結果の内部確認6.4 ソフトウェア開発の完了確認6.4.1 ソフトウェア開発の完了確認 <p>SAP2 安全性テスト</p> <ul style="list-style-type: none">2.1 安全性テストの準備2.1.1 安全性テストの準備2.2 安全性テストの実施2.2.1 安全性テストの実施2.2.2 安全性テスト結果の確認2.3 安全性テスト結果の確認2.3.1 安全性テスト結果の内部確認 <p>SYP3 システム結合テスト</p> <ul style="list-style-type: none">3.1 システム結合テストの準備3.1.2 システム結合テストの準備3.2 システム結合テストの実施3.2.1 システム結合テストの実施3.2.2 システム結合テスト結果の確認3.3 システム結合テスト結果の確認3.3.1 システム結合テスト結果の内部確認 <p>SYP4 システムテスト</p> <ul style="list-style-type: none">4.1 システムテストの準備4.1.2 システムテストの準備 |
|---|--|

- 4.2 システムテストの実施
 - 4.2.2 システムテスト結果の確認
- 4.3 システムテスト結果の確認
 - 4.3.1 システムテスト結果の内部確認
- 4.4 システム開発の完了確認
 - 4.4.1 システム開発の完了確認

- SUP1** プロジェクトマネジメント
 - 1.4 プロジェクト完了報告書の作成
- SUP6** 問題解決管理
 - 6.1 問題の記録と原因分析
 - 6.2 影響分析と対策立案
 - 6.3 対策の実行
 - 6.4 対策結果の確認

■ (SU601) 不具合管理票の例

| 不具合管理票 | | | | | | |
|---|---|-------------------------------|------|-----------|-------------|----|
| プロジェクト名 | | | 管理No | | | |
| 状況 | 発見日時 | 年 月 日 時 | 発見者 | 不具合箇所・Ver | | |
| | 標題 | | | | | |
| | 発見工程 添付： <input type="checkbox"/> 有 <input type="checkbox"/> 無 | | | | | |
| 原因・影響 | 調査日 | 年 月 日 | チーム | 担当 | 主な原因 | |
| | 原因 添付： <input type="checkbox"/> 有 <input type="checkbox"/> 無 | | | | | |
| | 影響 | 【影響範囲／対応しない場合の影響など（必要に応じて記載）】 | | | 不具合を作り込んだ工程 | |
| 添付： <input type="checkbox"/> 有 <input type="checkbox"/> 無 | | | | | | |
| 対応内容 | 検討日 | 年 月 日 | チーム | 担当 | 見積 | 工数 |
| | 完了日 | 年 月 日 | チーム | 担当 | | |
| | 対応内容 添付： <input type="checkbox"/> 有 <input type="checkbox"/> 無 | | | | | |
| 対応方針 | 対応する／しない | 優先度 | 期限 | 備考 | | |
| ※確認欄 | | | | | | |
| 対応承認 | | 検証完了 | | | | |
| / / | | / / | | | | |

プロジェクト名、管理情報を記載する。

不具合の発見日時、状況等を記載する。

不具合箇所や発見工程についても記載する。

不具合の原因と影響範囲を記載する。

不具合を作り込んだ工程についても記載する。

対応内容とその対応を実施するチーム/担当者と対応にかかる工数等を記載する。

対応を実施するか否か、対応作業の優先度、作業期限を記載する。

対応確認・検証完了を記載できる欄を設ける。

Related Standards

**ISO 9241 Ergonomic requirements for office work with visual display terminals (VDTs) – Part 10: Dialogue principles
Part 11: Guidance on usability**

**JIS Z 8520 人間工学 – 視覚表示装置を用いるオフィス作業
– 対話の原則**

**JIS Z 8521 人間工学 – 視覚表示装置を用いるオフィス作業
– 使用性についての手引き**

成立・改訂年： ISO 1996 (8241-10)、1998 (9241-11)、JIS 1999 (JIS Z 8520、8521)

目的： 視覚表示装置 (VDT) を利用したオフィスシステムにおける利用者の満足度の度合の
規定

対象分野： オフィス作業に使用する視覚表示装置 (VDT) を設計・製造および評価する組織・団体

URL： <http://www.iso.org/>

国内入手先： 日本規格協会

概要：

この規格は、視覚表示装置を利用したオフィス作業における事柄を人間工学的な視点から整理しています。ISO 9241 は、ISO 9241-1～17 の全 17 部で構成されていますが、そのうち、ISO 9241-10 と ISO 9241-11 がユーザビリティに関連する規格です。

「ISO 9241-10 : Dialogue principles (対話の原則)」では、ソフトウェアを設計・評価するときに、ユーザとコンピュータの対話として望ましい形を考えるための視点を提示しています。より具体的にユーザビリティについて定義しているのは「ISO 9241-11 : Guidance on usability (使用性についての手引き)」で、ユーザビリティを規定または評価する場合に、考慮すべき情報を提示しています。ほかに、ソフトウェア品質におけるユーザビリティに関連する規格として ISO 9126 があります。

ユーザビリティ (Usability)：

一般にソフトウェアやハードウェアの使いやすさを示す度合いを表す言葉です。ユーザビリティに対応する日本語として、使い勝手、使いやすさ、利用性、使用性、可用性、利用品質などがあてられますが、組込みシステムにとって、ユーザビリティとは利用者側からの品質 (利用品質) の 1 つの特性として位置づけられるようになってきました。

(SU602) 不具合管理台帳

不具合管理台帳には、不具合の一覧表として不具合管理票(SU601)の抜粋を記載します。不具合管理台帳をもとに不具合の管理を行います。

■ このドキュメントを作成するサブタスク

SUP6 問題解決管理

- 6.1 問題の記録と原因分析
- 6.2 影響分析と対策立案
- 6.3 対策の実行
- 6.4 対策結果の確認

■ ドキュメントをまとめる際に参照する情報

(SU601) 不具合管理票

■ 記載項目例

(SU601) 不具合管理票の抜粋を一覧表としてまとめる。

- ▶ プロジェクト名
- ▶ 管理番号

- ▶ 状況
- ▶ 原因・影響調査完了日
- ▶ 対応
- ▶ 検証完了日

■ このドキュメントを利用するサブタスク

SUP6 問題解決管理

- 6.2 影響分析と対策立案
- 6.3 対策の実行
- 6.4 対策結果の確認

(SU103、SU104) プロジェクト完了報告書

プロジェクト完了報告書には、システムテストの結果報告書や共同レビュー結果をもとに、システム開発の最終的な状況・結果の報告や製品リリース判断の元となる品質情報、次回開発における改善項目などを記載する。

■ このドキュメントを作成するサブタスク

SUP1 プロジェクトマネジメント

- 1.4 プロジェクト完了報告書の作成

■ ドキュメントをまとめる際に参照する情報

| | |
|-------------------------------|--------------------------|
| (SW105) ソフトウェア要求仕様書 | (SA104) 安全要求仕様書 |
| (SW606) ソフトウェア総合テスト報告書 | (SA201) 安全性テスト仕様書 |
| (SU801) 共同レビュー記録(ソフトウェア総合テスト) | (SA206) 内部確認レポート(安全性テスト) |
| (SY106) システム要求仕様書 | (SU601) 不具合管理票 |
| (SY406) システムテスト報告書 | (SU101) プロジェクト計画書 |
| (SU801) 共同レビュー記録(システムテスト) | (SU102) プロジェクト状況報告 |

■ 記載項目例

1. 概要
2. プロジェクト概要
 - 2.1 プロジェクト・プロファイル
 - 2.2 システム構成
3. 分析とフィードバック
 - 3.1 エンジニアリング全般
 - 3.2 工数比と工期比の実績
 - 3.3 品質の実績
 - 3.4 生産性の実績
 - 3.5 見積りと実績工数との差異
4. システムの運用・利用における制約事項
5. 課題の整理
6. 添付資料

■ このドキュメントを利用するサブタスク

(製品審査、類似の次回開発プロジェクトなど)

■ (SU103、SU104) プロジェクト完了報告書の例

表紙

文書名

● 文書名を記載する。

文書番号 ● 文書の識別情報を記載する。

| | |
|------|------|
| 承認 | 作成 |
| 承認者名 | 作成者名 |
| 承認日 | 作成日 |

● 作成・承認の責任・確認を記載できる欄を設ける。

発行日 ● 発行組織名、発行日を記載する。
 発行部署

改訂履歴

文書名

改訂履歴

| 項番 | 日付 | バージョン | 改訂内容 | 備考 |
|----|----|-------|--------------------|----|
| 1 | | | ● 改訂情報を記載できる欄を設ける。 | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| | | ⋮ | | |
| | | ⋮ | | |
| | | ⋮ | | |

文書番号
頁番号
発行日・発行部署

目次

文書名

目次

| | |
|----------------------------|-----|
| 1. 概要..... | 頁番号 |
| 2. プロジェクト概要..... | 頁番号 |
| 2.1 プロジェクト・プロフィール..... | 頁番号 |
| 2.2 システム構成..... | 頁番号 |
| 3. 分析とフィードバック..... | 頁番号 |
| 3.1 エンジニアリング全般..... | 頁番号 |
| 3.2 工数比と工期比の実績..... | 頁番号 |
| 3.3 品質の実績..... | 頁番号 |
| 3.4 生産性の実績..... | 頁番号 |
| 3.5 見積りと実績工数との差異..... | 頁番号 |
| 4. システムの運用・利用における制約事項..... | 頁番号 |
| 5. 課題の整理..... | 頁番号 |
| 6. 添付資料..... | 頁番号 |

文書番号
頁番号
発行日・発行部署

1. 概要

- <記載例>
- ・本書の目的
 - ・本書の位置づけ
 - ・対象ユーザ
 - ・記載範囲、記載内容 など
 - ・参照しているドキュメント など
 - ・定義（用語、略語 など）

ドキュメントの目的、位置づけ、記載内容などの本書の概要および参照しているドキュメント名等を記載する

2. プロジェクト概要

2.1 プロジェクトプロフィール

- <記載例>
- ・開発期間、実績工数、制約条件など

プロジェクトの基本事項に関して記載する。

2.2 システム構成

- <記載例>
- ・システム全体構成
 - システム構成要素の名称/基本機能
 - ・システムの動作環境および外部環境

ハードウェアおよびソフトウェアを含めたシステム全体の構成を記載する。

3. 分析とフィードバック

3.1 エンジニアリング全般

<記載例>

- ・製品ビジョン、コンセプトに対する施策
- ・システム要求に対する施策
- ・製品実権における制約事項（納期、品質、コスト、再利用、開発メンバの知識・経験など）に対する施策

製品ビジョン、コンセプトや制約条件に対して、システム開発を行う上で、どのような工夫・施策を実施してきたか、またその効果や達成状況を簡潔に記載する。

3.2 工数比と工期比の実績

<記載例>

- ・工程ごとの工数比（計画）
- ・工程ごとの工数比（実績）
- ・工数比の評価
- ・今後の課題

工程（アクティビティレベル）の計画工数と実績工数を比較し、達成状況を評価し記載する。
計画と実績に乖離がある場合や過去の類似プロジェクトの実績と乖離している場合はその原因を分析し、今後の課題として整理する。

3.3 品質の実績

<記載例>

- ・品質目標値
- ・品質実績値
- ・品質保証の仕組み
- ・品質の評価
- ・今後の課題

品質目標と実績を比較し、達成状況を評価し記載する。
計画と実績に乖離がある場合はその原因を分析し、今後の課題として整理する。

<品質目標値の例>

- ・機能性、信頼性、使用性、効率性、保守性、移植性
- ・要求定義、設計、実装工程での不具合検出率、レビュー実施率
- ・テスト工程でのテスト密度、障害検出率、障害収束率

3.4 生産性の実績

<記載例>

- ・生産性目標値
- ・生産性実績値
- ・生産性の評価
- ・今後の課題

生産性目標と実績を比較し、達成状況を評価し記載する。
計画と実績に乖離がある場合や過去の類似プロジェクトの実績と乖離している場合はその原因を分析し、今後の課題として整理する。

3.5 見積りと実績工数の差異

<記載例>

- ・見積り予実差異
- ・コスト予実績差異
- ・予実の評価
- ・今後の課題

コストと工数の見積りと実績を比較し、達成状況を評価し記載する。
見積りと実績に乖離がある場合はその原因を分析し、今後の課題として整理する。

4. システムの利用・運用における制約事項

システムの不具合対応の結果生ずるシステムの利用や運用の際の制約事項などを記載する。

5. 課題の整理

開発を通して確認できた開発プロセス面、開発技術などの課題を整理し、次回開発における改善項目として記載する。

6. 添付資料

<記載例>
・成果物一覧表

開発を通して確認できた開発プロセス面、開発技術などの課題を整理し、次回開発における改善項目として記載する。

Part 3

活用編

- 3.1 活用の手順 218
- 3.2 組織 / 部門の開発プロセスの整備 221
- 3.3 開発プロジェクトの工程設計 222
- 3.4 開発プロジェクトの作業計画 (工程設計の詳細化) ... 227

3.1 活用の手順

■ プロセスの整備にむけて

本ガイドのPart 2で示したプロセス定義を利用することで、個々の組織やプロジェクトに最適な開発プロセスを容易に整備することができます。

本ガイドのPart 2で示したプロセス定義は、組込みソフトウェア全般を対象とした標準的な開発作業を整理したものです。しかし、実際の製品開発を効率的に進めるためには、それぞれの製品の特徴やその開発を担う組織などの特性を考慮した最適な開発プロセスを整備する必要があります。

Part 3では、本ガイドを利用した個々の製品開発における開発プロセスの活用方法について説明します。

なお、以下に示す活用方法は、一例であり、本ガイドを利用する上で、参考にさせていただくことを想定しています。

■ 開発プロセスの活用

本ガイドのPart 2では標準的な開発プロセスを整理してあります。ここでいう開発プロセスとは、開発を進めていく上で必要となる作業をいくつかの塊に整理したものです。

しかし、実際の製品開発に関しては、対象とする製品の特性によって、開発の中で実施すべき作業が異なったり、作業の重点が異なったりすることは容易に想像できます。

このため、製品開発に関する開発プロセスを活用する上では、

- ①組織/部門の開発プロセス整備
- ②開発プロジェクトの工程設計
- ③開発プロジェクトの作業計画

という3つの作業を行う必要があります(図3.1)。

①組織/部門の開発プロセス整備

組織/部門の開発プロセスの整備では、対象とする製品の特性や開発組織の特性に合わせて、開発プロセスをチューニング(調整)します。

②開発プロジェクトの工程設計

一方、実際の製品開発の段階では、各開発プロジェクトに課せられる納期、品質、コスト等の制約条件を考慮しなくてはなりません。

このため、整備した開発プロセスを利用して、アクティビティ(あるいはタスク、サブタスク)レベルで、各作業にかかるウェイトや作業内容を検討する必要があります。これらを考慮し、実際の時間軸上に開発作業を割り付ける作業を工程設計と呼びます。また、実際に割り付けたものを開発工程と呼びます。

③開発プロジェクトの作業計画(工程設計の詳細化)

さらに、開発をいつスタートするか、そしていつ開発を終わらせるか、また、個々の作業をどの部門に委ねるか、あるいは、実際の作業者を誰にするかといった作業の進め方の詳細(工程設計の詳細化)を決める必要があります。これらを考慮して、作業をスケジューリングし、リソース配分します。

この作業を作業計画と呼びます。また、実際に計画したものを開発計画と呼びます。

なお、作業計画は、プロジェクトマネジメントの範疇になりますので、本ガイドでは概略について記述します。

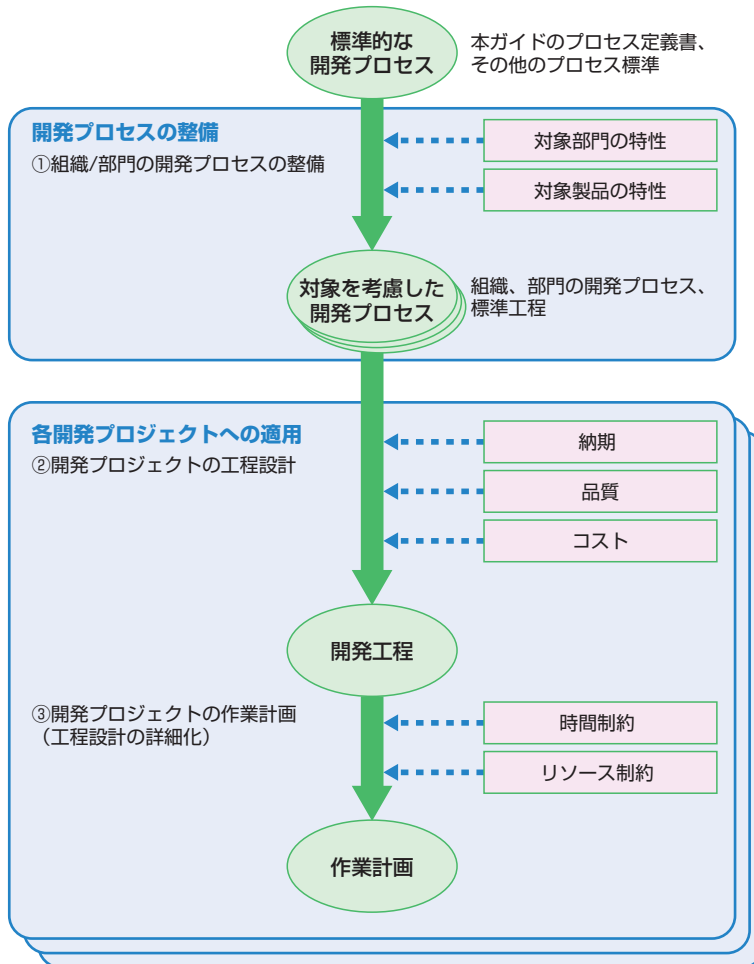


図3.1 開発プロセスの活用例

3.2 組織 / 部門の開発プロセスの整備

組織 / 部門の特性に合った開発プロセスの整備

本ガイドのPart 2では標準的な開発プロセスを整理してあります。

しかし、組込みソフトウェアが搭載される製品はさまざまであり、実際のソフトウェア開発を行っていくには、対象とする製品の特性を考慮し、その開発を担う組織 / 部門に合わせて、開発プロセスを調整(チューニング)する必要があります。

開発プロセスの調整

開発プロセスの調整では、本ガイドのPart 2を参考に、プロセス、アクティビティ、タスク、サブタスクや成果物の名称を組織 / 部門で使っているものに修正したり、サブタスクの実施内容や注意事項を製品の特性に合わせて修正・追加します。

標準工程の整備

また、組織 / 部門における標準的な開発工程(標準工程)を整備し、実際の開発プロジェクトで利用できるようにします。

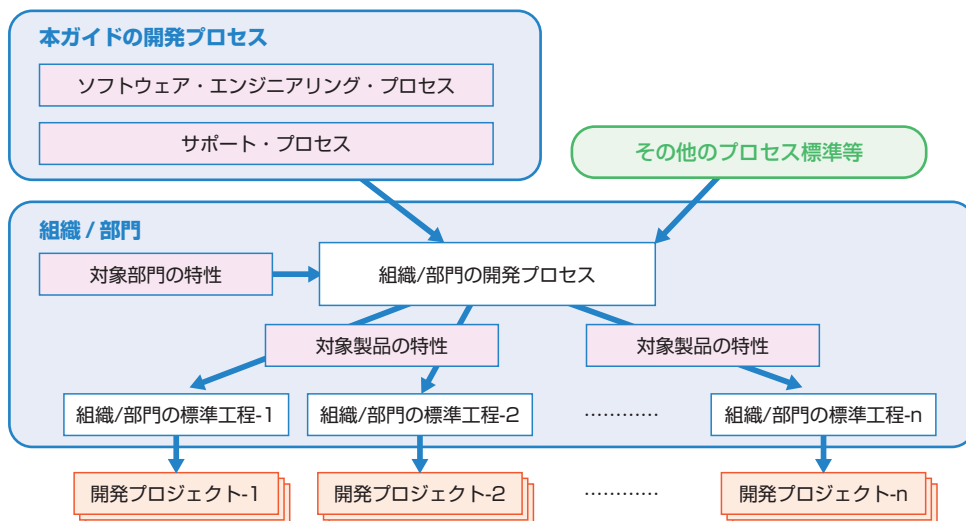


図3.2 組織 / 部門の開発プロセスの整備

3.3 開発プロジェクトの工程設計

組込みソフトウェアにおける開発プロジェクトの工程設計

開発プロジェクトの工程設計の目的は、製品として求められる要求（機能要求、非機能要求）と、その開発を担う組織やプロジェクトのレベルを考慮して、開発に必要な作業を洗い出し、その実施レベルを検討し、決定することにあります。

留意すべき事項

組込みソフトウェアにおける開発プロジェクトの工程設計を行う場合には特に、下記の事項に注意する必要があります。

- ①ハードウェアの開発作業との関係を意識し、整理しておく。また、それぞれの作業の間でのインタフェースを明確にしておく。
- ②各プロセス（作業）の入出力とそれぞれの作業内容（IPO：Input/Procedure/Output）を明確にする。
- ③各プロセス（作業）の開始条件/終了条件を明確にする。
- ④各プロセス（作業）の成果物を明確にしておく。
- ⑤各プロセス（作業）で参照する情報の依存関係を明確にしておく。
- ⑥ハンドリング可能な単位に作業をグループ分けしておく。
- ⑦作業内容などを確認するためのマイルストーンの考え方も整理しておく。

工程設計における本ガイドの利用方法

工程設計については、本ガイドで示すプロセス定義書を利用して、下記の手順で進めることができます。

- Step 1：**対象ソフトウェアおよび対象組織の特徴を把握する。
- Step 2：**標準プロセスの各プロセス、アクティビティレベルでの大まかな必要性や重要性を検討する。
- Step 3：**より具体的な作業レベル（タスク、サブタスク）についてPart 2の「2.2 プロセス定義書」を参考に、作業の実施内容を検討する。この際、作業にかけるウェイト

なども合わせて検討する。

Step 4: 作業結果として、どのような成果物を整理していくかを検討し、必要に応じてPart 2の「2.3 ドキュメント・テンプレート例」なども参考にする。

本ガイドでは、開発の直接作業に相当するエンジニアリング・プロセス (SYP、SWP) とそれを支えるサポート・プロセス (SUP)、セーフティ・エンジニアリング・プロセス (SAP) を規定しています。上記の開発作業の選定の際にも、それぞれエンジニアリング・プロセスとして実施すべき事項、サポート・プロセス、セーフティプロセスとして実施すべき事項を整理し、それぞれをどのように組み合わせていくかについても検討する必要があります。

全体工程設計の検討

工程設計の段階では、組織/部門の開発プロセスをベースに、アクティビティレベルで全体のスケジュールを検討し立案していきます。本ガイドでは、全体スケジュールを作成する作業を**全体工程設計**と呼びます。

全体工程設計を行う目的は、開発作業のマイルストーン、作業間のインタフェースや出力 (成果物) を明確にすることにあります (図3.4)。これらを明確にし、開発プロジェクトに関わる全員が情報を共有することは、計画どおりに開発を進めていく上で重要になります。

なお、ここでは組織/部門で整備した開発プロセス、組織/部門の標準工程からプロセス、アクティビティ、タスク、サブタスクを選定し、修正、並び替え、あるいは再帰的に繰り返すことを想定しています。

明確にすべき事項

全体工程設計では、下記の事項を明確にします。

- ①開発プロジェクトの開発工程
- ②出力 (成果物)
- ③対外インタフェース (外部機関との調整、やり取り)

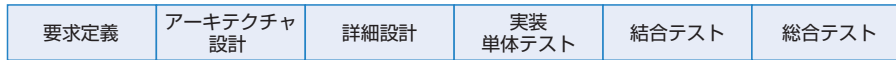
開発プロジェクトの開発工程

開発工程とは、順序性を持った (時間軸上に開発作業を割り付けた) アクティビティ群を指します。

開発モデル (ウォーターフォール開発、スパイラル開発等)、作業スコープ (開発の責任を

持つアクティビティの範囲)、新規開発/追加開発などを考慮して、実施するアクティビティを選定し、その順序を明確にします(図3.3)。

①小規模な開発のケース



②大規模な開発で複数の機能を並行して開発するケース

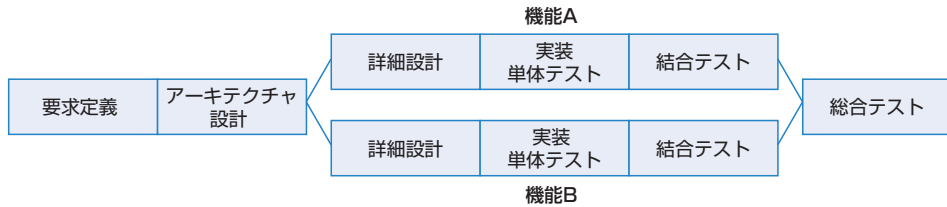


図3.3 開発工程例

出力(成果物)

出力とは、各アクティビティでの主要な成果物を指します。

対外的な関係からの成果物の提供時期やアクティビティ間の入出力を考慮して出力(成果物)を明確にします。

対外インタフェース(外部機関との調整、やり取り)

対外インタフェースとは、外部からの入力、外部への出力および共同作業等を指します。

ハードウェアとのコンカレント開発などを考慮して、開発作業のマイルストーンを明確にします。

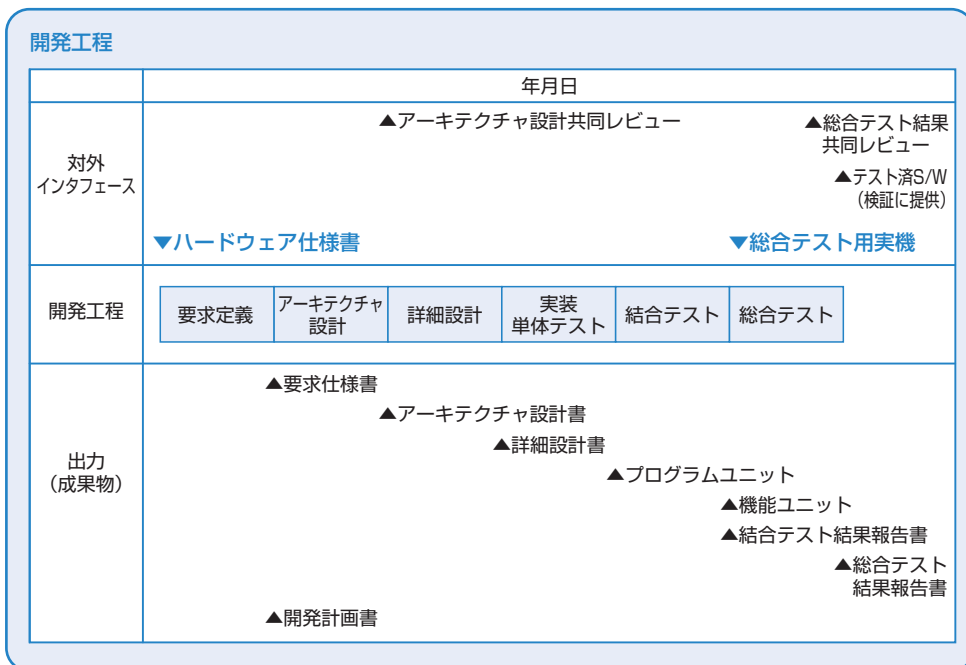


図3.4 全体工程設計例

開発工程の事例

ここでは、開発工程のいくつかの事例について紹介します。

テスト優先(先取りテスト)の事例

ユーザインタフェースが重要な開発などでは、使ってみて、すなわちテストしてみて分かる問題が多いため、要求定義や設計段階など、早い時期にテスト仕様を作成するタスクを入れるなどの考慮をします(図3.5)。

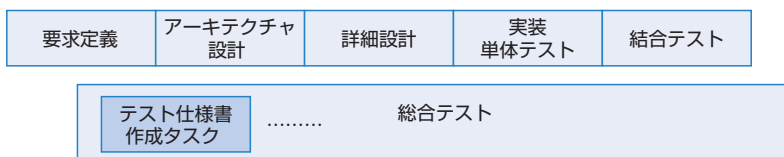


図3.5 開発工程例(テスト優先)

コンカレント開発の事例

新規のハードウェアとのコンカレント開発では、開発途中での共同レビューや、ハードウェア仕様書の改版・提供、テスト用の実機の品質確保などが重要になります。このような場合には、対外インタフェースとして、

- ①ハードウェアとの共同作業（共同レビュー、共同テスト、共同トラブル解析等）
- ②ハードウェアへの提供（ハードデバッグ用S/W、使用手引書等）
- ③ハードウェアからの受領（改訂版ハードウェア仕様書、デバッグ/テスト用実機等）

などの時期を明確にするなどの考慮をします（図3.6）。

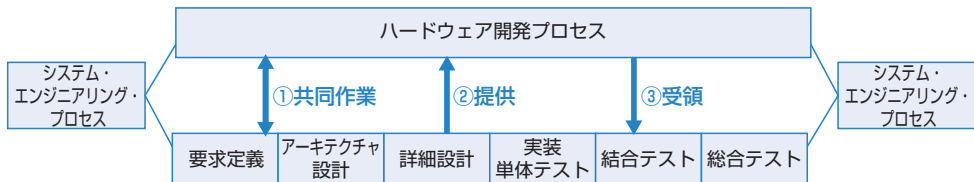


図3.6 開発工程例（コンカレント開発）

外部発注の事例

外部発注を伴う開発の場合は、発注作業、検収作業、進捗管理、品質保証、納品物管理等、開発委託管理のアクティビティとタスクをどのようなタイミングで組み込んでいくかがきわめて重要なポイントになります。

特に外部委託（発注）先のプロセスとの同期などは、工程設計の段階で十分に調整しておく必要があります（図3.7）。

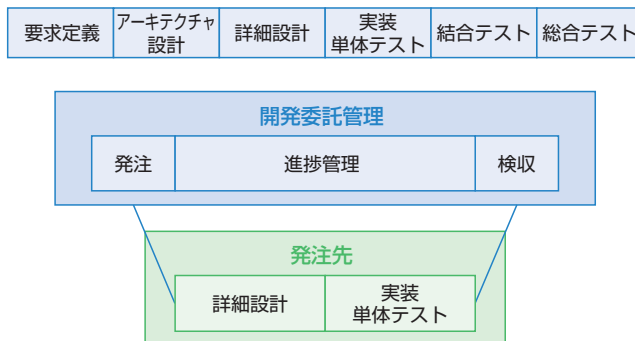


図3.7 開発工程例（外部発注）

3.4 開発プロジェクトの作業計画 (工程設計の詳細化)

■ 組込みソフトウェア開発における作業計画（工程設計の詳細化）

組込みソフトウェア開発では、基本的にソフトウェア開発だけでなくシステムを構成するハードウェア開発も意識して**作業計画**を考える必要があります。ここでの作業計画とは、先に全体工程設計によって検討した必要な作業をさらに詳細化して、開発の実時間上のせ、それぞれの作業の担当などを実際の組織や技術者に割り付けたものを言います。また、それらを割り付ける作業を作業計画と呼びます。

なお、作業計画は、プロジェクトマネジメントの範疇になりますので、本ガイドでは参考として概略について記述しています。作業計画の詳細については、『組込みソフトウェア向けプロジェクトマネジメントガイド[計画書編]』を参照してください。

組込みソフトウェアの作業計画で留意すべき事項

組込みソフトウェアの作業計画を考える上で特に注意が必要な事項としては下記のようなものが挙げられます。

- ①ハードウェア開発プロセスなどとの整合性や連携点の織り込み。
- ②作業上の順序関係などを考慮した作業の並行化などの工夫。
- ③個々の技術者のスキルなどを意識した作業の分担・割り振り。

作業計画における基本手順

組込みソフトウェアの作業の計画をする場合の基本手順は下記のとおりです(図3.8)。

Step 1：個々の作業の作業順序を決める

- ①WBS (Work Breakdown Structure)：個々の作業の作業量を見積もる。
- ②PERT (Program Evaluation and Review Technique)：作業間の関連を整理し、実施順序を決める。

③作業マイルストーンを設定する。

Step 2：作業分担を決める

①各作業の責任部門、責任者などを決める。

②投入する開発者などリソース制約を考慮し作業分担を決める。

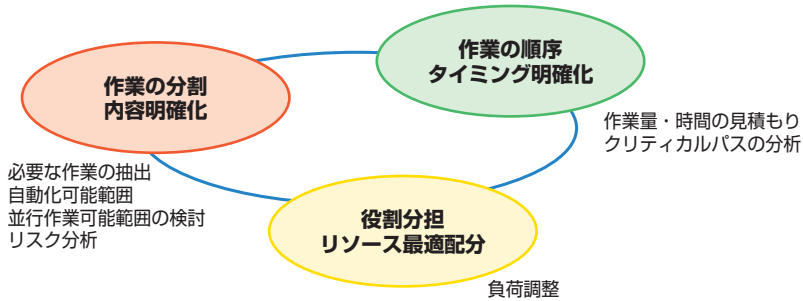


図3.8 開発プロジェクトの作業計画の留意点

作業順序の決定

全体工程設計の段階では、個々の作業の実施レベルや論理的な順序関係などの整理を中心としていますが、作業計画の段階では、具体的な開発の時間の中でこれらの作業を実時間上にはめ込んでいきます。

作業量の見積もり

開発の作業計画で最も基礎となる情報は、個々の作業に関する作業量の見積もり値です。個々の作業を実施する際に、どの程度の手数が掛かるか、あるいは、どの程度のボリュームのアウトプットを作るかなどを考慮して、個々の作業量を見積もります。この作業量の見積もりについては、過去の類似システム開発でそれぞれの作業にどの程度の作業量をかけたかなどのデータを参考にすることも有効です。

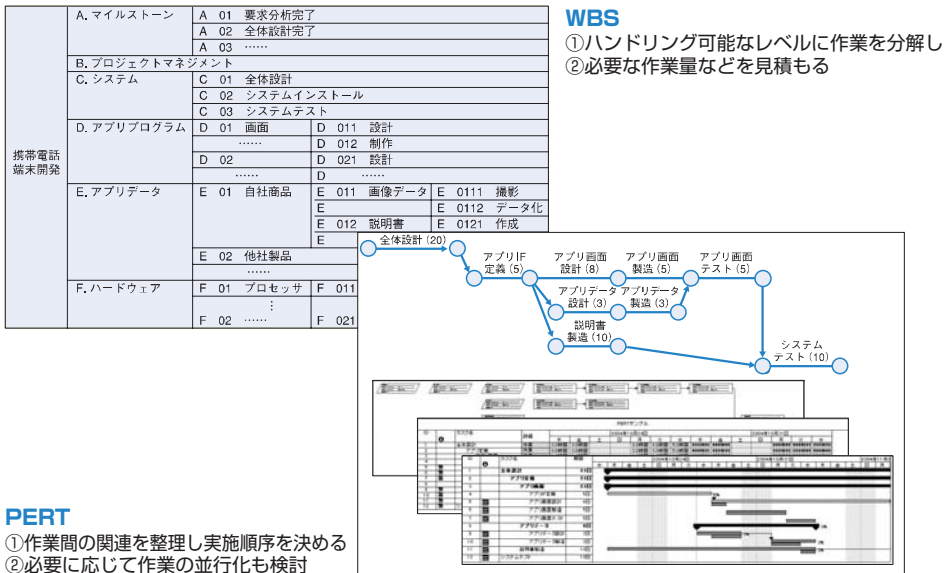
このようにして見積もった作業量はWBSの形で整理すると分かりやすくなります(図3.9)。

作業順序の決定

個々の作業の作業量、および、各作業の論理的な順序関係をもとに、実際の作業の実施順序を決定します。この際のポイントは、

- ①決められた納期を意識し、その中にすべての作業が納まりきるかどうかを考慮する。
- ②可能な範囲で複数部門や複数作業による作業を想定し、作業の並行化を考える。
- ③ハードウェアや周辺機器などの開発工程とのすりあわせに注意する。
- ④作業上のクリティカルパスなども分析評価し、作業上のボトルネックを把握しておく。

といった事項を考える必要があります。この際、PERTなどは有効な道具の1つになります(図3.9)。



PERT

①作業間の関連を整理し実施順序を決める
②必要に応じて作業の並行化も検討

図3.9 WBSとPERT

作業分担の決定

個々の作業をどの部門に委ねるか、あるいは誰に任せるかといった実際の作業の割付については、組織内外の技術者のスキルや作業量などを考慮することが求められます。

作業担当部門などの割付

まず、それぞれの作業を担当技術者に割り付ける前に、個々の作業をどの部門に委ねるか、また、それぞれの作業の責任者を誰にするかなどを決める必要があります。この場合、それぞれの作業内容を考慮し、部門や責任者のミスマッチが生じないように配慮します。

リソースの配分

実際に規模の大きな開発では、複数人の開発者を効率的にアサインする必要があります。また、開発工程上のどの段階でどの程度のスキルの人材を、どれくらいの人数を投入するかといった事項を検討していきます。特に、開発ピーク時のリソースや開発初期段階のリソース配分などに注意をする必要があります。

また、一般的に、多くの技術者は既に担当業務を持っており、その業務の目処が立った時点で、当該業務にアサインするといったことが多くあります。この場合、既担当業務の状況なども十分勘案したリソース充当計画を立てる必要があります(図3.10)。

| | 10月 | 11月 | 12月 | 1月 | 2月 | 3月 | 4月 | 5月 | 6月 | 7月 | |
|----------------|------|-----------|------|----|-------|-------|-------|----|----|----|-------|
| ベース工程計画 | 要求定義 | アーキテクチャ設計 | 詳細設計 | 実装 | 単体テスト | 結合テスト | 総合テスト | | | | |
| 投入工数 | | | | | | | | | | | |
| 仕様設計(人) | | 4 | 4 | 8 | 8 | 2 | 2 | 2 | 2 | 2 | 34人月 |
| ソフト設計(人) | 5 | 13 | 17 | 22 | 29 | 80 | 78 | 54 | 39 | 20 | 358人月 |
| (価評)人 | | | | 5 | 5 | 44 | 44 | 54 | 54 | 24 | 230人月 |

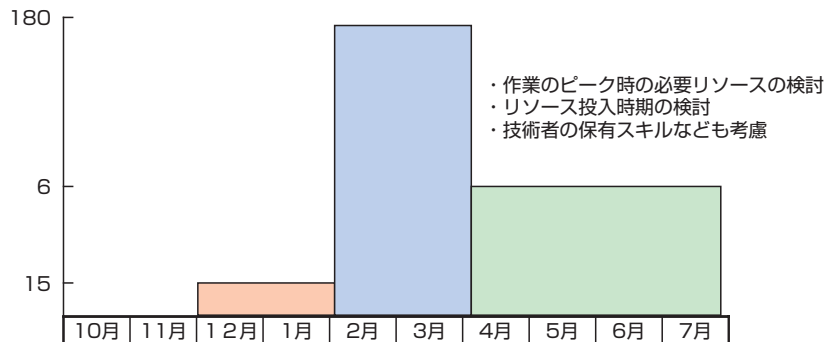


図3.10 作業分担例

■ 開発プロジェクトの作業計画の考え方

ここでは簡単な例をもとに開発プロジェクトの作業計画の考え方を紹介します。

例1：小規模開発（新規）

新規に光学式マウスなどを開発する場合

ソフトウェアとしての特徴：

| | |
|-------------|-----------|
| ソフトウェア規模： | 数百行程度（新規） |
| ユーザインタフェース： | ボタン3つ程度 |
| 開発担当技術者数： | 1名ないし2名 |

この場合、新規開発と開発規模が小さいことを考慮すると、たとえば、

- **SWP1.2（1.2.2 内部確認レポートの作成）**
オフィシャルに実施するよりも、開発グループ内での簡易的な確認にする。
- **SWP4.1（4.1.1 実装の準備）**
再利用するプログラムのユニットは存在しないため作業として考慮する必要はない。

といった項目について、作業としては軽めにするなどの工夫をすることも可能です。

また、一方で、マウスという機器の特性上、レスポンスなど性能面が比較的重視される傾向があるため、

- **SWP2.1（2.1.1 設計条件の確認）**
動作環境や性能をはじめとする非機能的側面の要求反映方法について特に重点的に検討する。
- **SWP3.3（3.3.1 ハードウェア仕様との整合性の確認）**
マウスの中心的なデバイスである光学系の性能などとのすりあわせを重点的に行う。

といったプロセスごとの重要度を考慮した重み付けを行い、実施程度や内容を決めていきます。

例2：大規模開発（追加）

既存のカーナビシステムをベースに新規機能を追加した新システムを開発する場合

ソフトウェアとしての特徴：

- ソフトウェア規模： 大規模（数百万行）
- ユーザインタフェース： ディスプレイを中心に複雑なインタフェースを有する。
- 開発担当技術者数： 100名以上

この場合には、極めて大規模開発であるため、

- ・ SUP1 プロジェクトマネジメント
- ・ SUP8 共同レビュー

などのプロセスを重点的に実施する必要があります。また、大規模ソフトウェアであることと、既存ソフトウェアの流用であることを考慮すると、

- ・ **SWP2.1 (2.1.2 ソフトウェア構成の設計)**

システム全体の構成の整合性や、コア部分の分離などを工夫し、次の再利用にも耐えられるような構造にしておく。

さらに、ユーザインタフェースの複雑性や操作性などへの配慮から、

- ・ **SWP1.1 (1.1.2 ソフトウェア機能要件の明確化)**
(1.1.3 ソフトウェア非機能要件の明確化)

システムの使用性や人間中心設計プロセスなどを考慮しておく。

などの工夫を加える必要があります。

付録

| | | |
|-----|-------------------------|-----|
| 付録1 | 用語 | 234 |
| 付録2 | 規格対応表 (本ガイド- X0160) | 238 |
| 付録3 | アクティビティ / タスク / サブタスク一覧 | 239 |
| 付録4 | ESPR Ver.1.0 からの変更点 | 242 |

付録 1 用語

■ P

PERT (Program Evaluation and Review Technique)

PERTは、WBSで洗い出された作業を組み合わせ、プロジェクトを最短で終了させるための計画を立案する手法です。PERTでは、作業の順番(並行して行うことのできる作業か、順に行う必要のある作業か)からスケジュール計画を立案していきます。

PERTを作成することによって、プロジェクト計画上のクリティカルパスを明確にすることができます。クリティカルパスとは、PERT上でプロジェクトの達成期間を決めている一連の作業です。クリティカルパス上の作業が1つでも遅れることによって、プロジェクト全体の期間が延びることになります。

このクリティカルパスを明確にして、クリティカルパス上の作業が遅れないように管理することが、プロジェクトマネジメントにおいて重要なポイントになります。

■ W

WBS (Work Breakdown Structure)

WBSは、プロジェクト全体を詳細な作業に分割する手法です。プロジェクトのゴール(開発すべき製品)を、細かな成果物に階層的に詳細化し、それぞれの成果物を開発するために必要な作業を配置していきます。

WBSを作成するときには、特にプロジェクトの重要な要件について、より具体的に、成果がイメージできるまで詳細化していきます。

WBSを作成することによって、プロジェクトの目標を達成するための詳細な作業を洗い出すことができます。加えて、作成したWBSをプロジェクトに関わる人々(スポンサー、マーケット担当、プロジェクトメンバなど)によって事前に確認することで、プロジェクトの範囲(範囲)を明確にでき、計画漏れによる後戻りを防ぐことができます。

また、プロジェクトを通じてWBSを管理することによって、要件の変更によるプロジェクトへの影響を管理することが可能になります。

■ あ

アクティビティ

アクティビティはプロセスを実施する際のより具体的な作業をまとめた作業群です。

本ガイドでは、たとえばソフトウェア・エンジニアリング・プロセスを構成するアクティビティには以下のものがあります。

- ・ソフトウェア要求定義
- ・ソフトウェア・アーキテクチャ設計
- ・ソフトウェア詳細設計

- ・実装および単体テスト
- ・ソフトウェア結合およびソフトウェア結合テスト
- ・ソフトウェア総合テスト

■ か

機能要求

機能要求とは「xxxというデータを入力できること」、「メールを送信できること」といった、製品や利用者のニーズをもとにソフトウェアに求められる機能のことです。

関連用語 非機能要求

共同レビュー

開発作業の節目ごとに各プロセスでの作業結果が適切であったかどうかを、関係者間で技術面、管理面の両面から確認するレビューです。共同レビューでは、成果物の作成を担当した技術者だけでなく、製品開発に関係するステークホルダなども参加し、多視点でのレビューを行います。

関連用語 内部レビュー

工程設計

工程設計は、開発プロジェクトで実施する作業を、開発時間軸上に割り付ける作業です。また、実際に割り付けたものを開発工程と呼びます。

工程設計では、実際の作業の順序関係、並行作業、作業担当を検討して決めていきます。

■ さ

サポート・プロセス (SUP : SUpport Process)

サポート・プロセスは、組込みソフトウェアの開発を組織的に実施するため、開発作業(ソフトウェア・エンジニアリング・プロセス)を横断的に支える管理的側面に関する作業を整理したプロセスです。

ISO/IEC12207、15288では、さらにこのプロセスを組織/支援などのライフサイクルプロセス群に分割し、詳細に定義しています。Ver.2.0では、SUP1、SUP6、SUP7、SUP8、SUP10について整理してあります。このプロセスには下記のアクティビティが含まれます。

| | |
|--------------------------|---------------------|
| SUP1 プロジェクトマネジメント | SUP6 問題解決管理 |
| SUP2 品質保障 | SUP7 変更管理 |
| SUP3 リスクマネジメント | SUP8 共同レビュー |
| SUP4 文書化と文書管理 | SUP9 開発委託管理 |
| SUP5 構成管理 | SUP10 開発環境整備 |

システム・エンジニアリング・プロセス (SYP : SYstem engineering Process)

システム・エンジニアリング・プロセスは、組込みソフトウェアが組み込まれて動作する組込みシステムとしてとらえた場合のシステム要求やシステムとしての動作検証などの作業を整理したプロセスです。

このプロセスには、下記のアクティビティが含まれています。

| | |
|----------------------------|-----------------------|
| SYP1 システム要求定義 | SYP3 システム結合テスト |
| SYP2 システム・アーキテクチャ設計 | SYP4 システムテスト |

スタブ

スタブ(stub)とは、テスト対象部分(プログラムユニット)から呼び出される下位モジュールの代替品のことです。

関連用語 テストドライバ

ステークホルダ

ステークホルダは、企業内からエンドユーザまでを含め、製品に利害関係のある人のことです。

セーフティ・エンジニアリング・プロセス(SAP : SAfty engineering Process)

セーフティ・エンジニアリング・プロセスは、安全・安心な組込みシステムを作り上げるために実施すべき作業を整理したプロセスです。

このプロセスには、下記のアクティビティが含まれています。

SAP1 安全性要求定義

SAP2 安全性テスト

ソフトウェア・エンジニアリング・プロセス(SWP : SoftWare engineering Process)

ソフトウェア・エンジニアリング・プロセスは、組込みソフトウェア開発に関する作業の中でも、直接ものづくりに関係するソフトウェアとしての要求定義からソフトウェア総合テストまでの作業を整理したプロセスです。

このプロセスには、下記のアクティビティが含まれています。

SWP1 ソフトウェア要求定義

SWP5 ソフトウェア結合テスト

SWP2 ソフトウェア・アーキテクチャ設計

SWP6 ソフトウェア総合テスト

SWP3 ソフトウェア詳細設計

SWP4 実装および単体テスト

■ た

タスク

タスクは個々のアクティビティを実施し、達成する上で実施が求められる作業をいくつかのグループに分けて整理したものです。

本ガイドでは、たとえば「ソフトウェア要求定義」アクティビティを構成するタスクには以下のものがあります。

- ・ソフトウェア要求仕様書の作成
- ・ソフトウェア要求仕様の確認

テストドライバ

テストドライバとは、テスト対象部分(プログラムユニット)にテストデータを引き渡す上位モジュールの代替品のことです。

関連用語 スタブ

■ な

内部レビュー

開発作業の結果が適切であったかどうかを、成果物の作成を担当した技術者を含む開発グループなどで確認するレビューです。内部レビューでは、製品開発に関係するステークホルダは参加せず、技術面に重点を置いたレビューを行います。

関連用語 共同レビュー

■ は

非機能要求

非機能要求は「nn秒以内に処理を終えること」、「利用者がマニュアルを見ないで操作できること」、「再利用できること」など、ソフトウェアに求められる効率性や使用性、移植性などのことです。

関連用語 機能要求

プロセス(Process)

一般的にどのような業務でも、その業務の目的や目標を達成する上では、その途中でさまざまな作業を実施することが求められます。プロセスは、「どのような作業を実施していくか」を整理したものです。

本ガイドで扱うプロセスは組込みソフトウェア開発を進める上で、実施することが求められる作業をいくつかの塊(作業群)でとらえたもので、ソフトウェアという製品を作り上げる上で、実施すべき作業を整理したものをソフトウェア開発プロセスと呼んでいます。ソフトウェア開発プロセスは、以下の4つのプロセスから構成されます。

①システム・エンジニアリング・プロセス

組込みソフトウェアがベースとなって構築される組込みシステムを取りまとめる作業を中心に整理したもの

②ソフトウェア・エンジニアリング・プロセス

実際のソフトウェア開発に関係する主たる作業をまとめたもの

③セーフティ・エンジニアリング・プロセス

安全・安心な組込みシステムを作り上げるために実施すべき作業をまとめたもの

④サポート・プロセス

ドキュメンテーションなど開発に付随して発生するさまざまな支援作業をまとめたもの

付録2 規格対応表 (本ガイド — X0160)

| ESPR(本ガイド) | X 0160-1996 (ISO/IEC12207:1995) ソフトウェアライフサイクルプロセス |
|--------------------------|--|
| — | 取得プロセス |
| — | 供給プロセス |
| — | 開発プロセス |
| — | プロセス開始の準備 |
| SYP: システム・エンジニアリングプロセス | |
| SYP1 システム要求定義 | システム要求分析 |
| SYP2 システム・アーキテクチャ設計 | システム方式設計 |
| SWP: ソフトウェア・エンジニアリングプロセス | |
| SWP1 ソフトウェア要求定義 | ソフトウェア要求分析 |
| SWP2 ソフトウェア・アーキテクチャ設計 | ソフトウェア方式設計 |
| SWP3 ソフトウェア詳細設計 | ソフトウェア詳細設計 |
| SWP4 実装および単体テスト | ソフトウェアコード作成とテスト |
| SWP5 ソフトウェア結合テスト | ソフトウェア結合 |
| SWP6 ソフトウェア総合テスト | ソフトウェア適格性確認テスト |
| SYP: システム・エンジニアリングプロセス | |
| SYP3 システム結合テスト | システム結合 |
| SYP4 システムテスト | システム適格性確認テスト |
| — | ソフトウェア導入 |
| — | ソフトウェア受入れと支援 |
| — | 運用プロセス |
| — | 保守プロセス |
| SUP: サポート・プロセス | |
| SUP4 文書化と文書管理 | 文書化プロセス |
| SUP: サポート・プロセス | |
| SUP5 構成管理 | 構成管理プロセス |
| SUP: サポート・プロセス | |
| SUP7 変更管理 | — |
| SUP: サポート・プロセス | |
| SUP2 品質保証 | 品質保証プロセス |
| — | 検証プロセス |
| — | 妥当性確認プロセス |
| SUP: サポート・プロセス | |
| SUP8 共同レビュー | 共同レビュープロセス |
| — | 監査プロセス |
| SUP: サポート・プロセス | |
| SUP6 問題解決管理 | 問題解決プロセス |
| SUP: サポート・プロセス | |
| SUP1 プロジェクトマネジメント | 管理プロセス |
| SUP: サポート・プロセス | |
| SUP3 リスクマネジメント | — |
| SUP: サポート・プロセス | |
| SUP10 開発環境整備 | 環境整備プロセス |
| — | 改善プロセス |
| — | 教育訓練プロセス |
| — | 修正プロセス |
| SUP: サポート・プロセス | |
| SUP9 開発委託管理 | — |
| SAP: セーフティ・エンジニアリング・プロセス | |
| SAP1 安全性要求定義 | |
| SAP2 安全性テスト | — |

□ : 本ガイドで規定しているプロセス

付録3 アクティビティ/タスク/ サブタスク一覧

■ SYP : システム・エンジニアリング・プロセス

| | | | |
|-------------|-------------------------------|-------------|--------------------------|
| SYP1 | システム要求定義 ……………22 | SYP3 | システム結合テスト ……………43 |
| 1.1 | システム要求仕様書の作成 | 3.1 | システム結合テストの準備 |
| 1.1.1 | 製品企画書と製品仕様書の確認 | 3.1.1 | システム結合の準備 |
| 1.1.2 | システム機能要求の分析と整理 | 3.1.2 | システム結合テストの準備 |
| 1.1.3 | システム非機能要求の分析と整理 | 3.2 | システム結合テストの実施 |
| 1.1.4 | システム動作制約の明確化 | 3.2.1 | システム結合テストの実施 |
| 1.1.5 | システム要求の優先順位付け | 3.2.2 | システム結合テスト結果の確認 |
| 1.1.6 | システム要求仕様書の作成 | 3.3 | システム結合テスト結果の確認 |
| 1.2 | システム要求仕様書の確認 | 3.3.1 | システム結合テスト結果の内部確認 |
| 1.2.1 | システム要求仕様書の内部確認 | | |
| SYP2 | システム・アーキテクチャ設計 ……………32 | SYP4 | システムテスト ……………51 |
| 2.1 | システム・アーキテクチャ設計書の作成 | 4.1 | システムテストの準備 |
| 2.1.1 | 設計条件の確認 | 4.1.1 | システムテスト仕様書の作成 |
| 2.1.2 | システム構成の設計 | 4.1.2 | システムテストの準備 |
| 2.1.3 | システム全体の振る舞いの設計 | 4.1.3 | システムテスト仕様書の内部確認 |
| 2.1.4 | インタフェースの設計 | | |
| 2.1.5 | システム・アーキテクチャ設計書の作成 | 4.2 | システムテストの実施 |
| 2.2 | システム・アーキテクチャ設計の確認 | 4.2.1 | システムテストの実施 |
| 2.2.1 | システム・アーキテクチャ設計書の内部確認 | 4.2.2 | システムテスト結果の確認 |
| 2.3 | システム・アーキテクチャ設計書の共同レビュー | 4.3 | システムテスト結果の確認 |
| 2.3.1 | システム・アーキテクチャ設計書の共同レビュー | 4.3.1 | システムテスト結果の内部確認 |
| | | 4.4 | システム開発の完了確認 |
| | | 4.4.1 | システム開発の完了確認 |

■ SWP : ソフトウェア・エンジニアリング・プロセス

| | | | |
|-------------|--------------------------------|-------------|------------------------------|
| SWP1 | ソフトウェア要求定義63 | SWP4 | 実装および単体テスト99 |
| 1.1 | ソフトウェア要求仕様書の作成 | 4.1 | 実装および単体テストの準備 |
| 1.1.1 | 制約条件の確認 | 4.1.1 | 実装の準備 |
| 1.1.2 | ソフトウェア機能要求事項の明確化 | 4.1.2 | 単体テストの準備 |
| 1.1.3 | ソフトウェア非機能要求事項の明確化 | 4.2 | 実装および単体テストの実施 |
| 1.1.4 | 要求の優先順位付け | 4.2.1 | プログラムユニットの実装 |
| 1.1.5 | ソフトウェア要求仕様書の作成 | 4.2.2 | 単体テストの実施 |
| 1.2 | ソフトウェア要求仕様の確認 | 4.2.3 | 単体テスト結果の確認 |
| 1.2.1 | ソフトウェア要求仕様書の内部確認 | 4.3 | 実装および単体テスト結果の確認 |
| | | 4.3.1 | ソースコードの確認 |
| | | 4.3.2 | 単体テスト結果の内部確認 |
| SWP2 | ソフトウェア・アーキテクチャ設計76 | SWP5 | ソフトウェア結合テスト 109 |
| 2.1 | ソフトウェア・アーキテクチャ設計書の作成 | 5.1 | ソフトウェア結合テストの準備 |
| 2.1.1 | 設計条件の確認 | 5.1.1 | ソフトウェア結合の準備 |
| 2.1.2 | ソフトウェア構成の設計 | 5.1.2 | ソフトウェア結合テストの準備 |
| 2.1.3 | ソフトウェア全体の振る舞いの設計 | 5.2 | ソフトウェア結合テストの実施 |
| 2.1.4 | インタフェースの設計 | 5.2.1 | ソフトウェア結合 |
| 2.1.5 | 性能/メモリ使用量の見積もり | 5.2.2 | ソフトウェア結合テストの実施 |
| 2.1.6 | ソフトウェア・アーキテクチャ設計書の作成 | 5.2.3 | ソフトウェア結合テスト結果の確認 |
| 2.2 | ソフトウェア・アーキテクチャ設計の確認 | 5.3 | ソフトウェア結合テスト結果の確認 |
| 2.2.1 | ソフトウェア・アーキテクチャ設計書の内部確認 | 5.3.1 | ソフトウェア結合テスト結果の内部確認 |
| 2.3 | ソフトウェア・アーキテクチャ設計の共同レビュー | SWP6 | ソフトウェア総合テスト 120 |
| 2.3.1 | ソフトウェア・アーキテクチャ設計書の共同レビュー | 6.1 | ソフトウェア総合テストの準備 |
| | | 6.1.1 | ソフトウェア総合テスト仕様書の作成 |
| | | 6.1.2 | ソフトウェア総合テストの準備 |
| | | 6.1.3 | ソフトウェア総合テスト仕様書の内部確認 |
| SWP3 | ソフトウェア詳細設計89 | 6.2 | ソフトウェア総合テストの実施 |
| 3.1 | 機能ユニット詳細設計書の作成 | 6.2.1 | ソフトウェア総合テストの実施 |
| 3.1.1 | プログラムユニット分割 | 6.2.2 | ソフトウェア総合テスト結果の確認 |
| 3.1.2 | プログラムユニット設計 | 6.3 | ソフトウェア総合テスト結果の確認 |
| 3.1.3 | インタフェースの詳細化 | 6.3.1 | ソフトウェア総合テスト結果の内部確認 |
| 3.1.4 | メモリ量の見積もり | 6.4 | ソフトウェア開発の完了確認 |
| 3.1.5 | ソフトウェア詳細設計書の作成 | 6.4.1 | ソフトウェア開発の完了確認 |
| 3.2 | ソフトウェア詳細設計の確認 | | |
| 3.2.1 | ソフトウェア詳細設計書の内部確認 | | |
| 3.3 | ハードウェア仕様との整合性の確認 | | |
| 3.3.1 | ハードウェア仕様との整合性の確認 | | |


■ SAP : セーフティ・エンジニアリング・プロセス

| | | | |
|-------------|--------------------------|-------------|------------------------|
| SAP1 | 安全性要求定義 132 | SAP2 | 安全性テスト141 |
| 1.1 | 安全性要求仕様書の作成 | 2.1 | 安全性テストの準備 |
| 1.1.1 | 製品企画書と製品仕様書の確認 | 2.1.1 | 安全性テストの準備 |
| 1.1.2 | 想定されるシステム障害の検討 | 2.2 | 安全性テストの実施 |
| 1.1.3 | 安全性実現のための要件検討 | 2.2.1 | 安全性テストの実施 |
| 1.1.4 | 安全性要求仕様書の作成 | 2.2.2 | 安全性テスト結果の確認 |
| 1.2 | 安全性要求仕様書の確認 | 2.3 | 安全性テスト結果の確認 |
| 1.2.1 | 安全性要求仕様書の内部確認 | 2.3.1 | 安全性テスト結果の内部確認 |

■ SUP : サポート・プロセス

| | | | |
|-------------|-------------------------------|--------------|-------------------------|
| SUP1 | プロジェクトマネジメント 149 | SUP6 | 問題解決管理 158 |
| 1.1 | プロジェクト計画書の作成 | 6.1 | 問題の記録と原因分析 |
| 1.2 | プロジェクト実施状況の把握 | 6.2 | 影響分析と対策立案 |
| 1.3 | プロジェクトの制御 | 6.3 | 対策の実行 |
| 1.4 | プロジェクト完了報告書の作成 | 6.4 | 対策結果の確認 |
| SUP2 | 品質保証 152 | SUP7 | 変更管理 160 |
| 2.1 | 品質目標の設定 | 7.1 | 変更要求情報の記録 |
| 2.2 | 品質管理方法の確定 | 7.2 | 変更による影響の分析 |
| 2.3 | 品質の可視化に基づく品質制御 | 7.3 | 変更計画の立案と実施 |
| SUP3 | リスクマネジメント 154 | 7.4 | 変更結果の確認 |
| 3.1 | リスクの洗い出しと把握 | SUP8 | 共同レビュー161 |
| 3.2 | リスクのモニタリング | 8.1 | レビューの準備 |
| 3.3 | リスク対策の決定と実行 | 8.2 | レビューの実施 |
| SUP4 | 文書化と文書管理 | 8.3 | レビュー結果の確認とフォロー |
| 4.1 | 文書の作成と確認 | SUP9 | 開発委託管理 |
| 4.2 | 文書の配布 | 9.1 | 発注の準備と契約 |
| 4.3 | 文書の保守と管理 | 9.2 | 開発委託作業のモニタリング |
| SUP5 | 構成管理156 | SUP10 | 開発環境整備 162 |
| 5.1 | 構成管理対象物の把握 | 10.1 | 開発環境整備計画の立案 |
| 5.2 | 構成管理/変更管理履歴の管理 | 10.2 | 開発環境の構築 |
| | | 10.3 | 開発環境の維持 |

付録4 ESR Ver.1.0からの変更点

| 項番 | ESPR V2.0の主な追加・変更点 | 追加/変更 | 該当箇所 |
|----|---|-------|-----------|
| 1 | システム・エンジニアリング・プロセス(SYP)の定義を追加しました。 | 追加 | Part2 技術編 |
| 2 | セーフティ・エンジニアリング・プロセス(SAP)の定義を追加しました。また、SYP、SWPにおいてセーフティに関連する作業を追加し、  マークを付けました。 | 追加 | Part2 技術編 |
| 3 | サポート・プロセス(SUP)に以下のアクティビティの定義を追加しました。 ・「SUP2 品質保証」 ・「SUP3 リスクマネジメント」 ・「SUP5 構成管理」 | 追加 | Part2 技術編 |
| 4 | 「SUP1 プロジェクトマネジメント」アクティビティに以下のタスクの定義を追加しました。 ・「SUP1.4 プロジェクト完了報告書の作成」 | 追加 | Part2 技術編 |
| 5 | ドキュメント・テンプレート例に以下のテンプレートを追加しました。 ・システム要求定義書 ・システム・アーキテクチャ設計書 ・安全要求仕様書 ・プロジェクト完了報告書 | 追加 | Part2 技術編 |
| 6 | ESPR V1.0 発行後にSECのホームページで公開したドキュメント・テンプレート例(電子版)との差分を反映しました。 | 変更 | Part2 技術編 |
| 7 | 「SWP2 ソフトウェア・アーキテクチャ設計」アクティビティのタスク/サブタスクを一部統合しました。 - 「SWP2.3.2 共同レビュー報告書の作成」の内容を「SWP2.3.1 ソフトウェア・アーキテクチャ設計の共同レビュー」に統合 | 変更 | Part2 技術編 |
| 8 | 「SWP5 ソフトウェア結合テスト」アクティビティのタスクの名称を変更しました。 - V1.0「SWP5 ソフトウェア結合および結合テスト」⇒ V2.0「SWP5 ソフトウェア結合テスト」に名称変更 - V1.0「SWP5.1 ソフトウェア結合およびソフトウェア結合テストの準備」⇒ V2.0「SWP5.1 ソフトウェア結合テストの準備」に名称変更 - V1.0「SWP5.2 ソフトウェア結合およびソフトウェア結合テストの実施」⇒ V2.0「SWP5.1 ソフトウェア結合テストの実施」に名称変更 | 変更 | Part2 技術編 |
| 9 | 「SWP6 ソフトウェア総合テスト」アクティビティのタスク/サブタスクの一部統合および名称を変更しました。 ・名称変更 - V1.0「SWP6.4 ソフトウェア総合テスト結果の共同レビュー」⇒ V2.0「SWP6.4 ソフトウェア開発の完了確認」に名称変更 - V1.0「SWP6.4.1 ソフトウェア総合テスト結果の共同レビュー」⇒ V2.0「SWP6.4.1 ソフトウェア開発の完了確認」に名称変更 ・統合 - 「SWP6.4.2 共同レビュー報告書の作成」の内容を「SWP6.4.1 ソフトウェア開発の完了確認」に統合 | 変更 | Part2 技術編 |

| 項番 | ESPR V2.0の主な追加・変更点 | 追加 / 変更 | 該当箇所 |
|----|---|---------------|-----------|
| 10 | <p>「x.x.x 内部確認レポートの作成」の内容を「x.x.x.の内部確認」サブタスクに統合しました。</p> <ul style="list-style-type: none"> - 「SWP1.2.2 内部確認レポートの作成」の内容を「SWP1.2.1 ソフトウェア要求仕様書の内部確認」に統合 - 「SWP2.2.2 内部確認レポートの作成」の内容を「SWP2.2.1 ソフトウェア・アーキテクチャ設計書の内部確認」に統合 - 「SWP3.2.2 内部確認レポートの作成」の内容を「SWP3.2.1 ソフトウェア詳細設計書の内部確認」に統合 - 「SWP4.3.3 内部確認レポートの作成」の内容を「SWP4.3.2 単体テスト結果の内部確認」に統合 - 「SWP5.3.2 内部確認レポートの作成」の内容を「SWP5.3.1 ソフトウェア結合テスト結果の内部確認」に統合 - 「SWP6.3.2 内部確認レポートの作成」の内容を「SWP6.3.1 ソフトウェア総合テスト結果の内部確認」に統合 | 変更 | Part2 技術編 |

あとがき

ソフトウェアというモノを1つの製品として完成させるためには、さまざまな作業を積み重ねていくことが必要となる。このソフトウェア、とりわけ組込みソフトウェアという製品を作り上げる上で実施すべき作業は何か。ESPR Ver.1.0は、経済産業省 組込みソフトウェア開発力強化推進委員会 組込みソフトウェアエンジニアリング領域 開発プロセス技術部会が、2年の歳月を費やして検討を重ね、組込みソフトウェア開発における実施すべき作業を『組込みソフトウェア向け 開発プロセスガイド』としてまとめ上げたものである(2006年秋)。

このESPR Ver.1.0では、まず、ソフトウェアを作る際の直接作業(ソフトウェア・エンジニアリング・プロセス)を整理することに重点を置き、組込みソフトウェアを開発する上で必要となる作業の実施内容や注意すべき事項を現場の技術者の方にも分かる言葉で体系的に整理した。

しかし、ソフトウェアを作るには、直接作業だけでなく、管理系・支援系の作業も大切なことは言うをまたない。今回、2006年秋に発行したESPR Ver.1.0をより充実したものとするべく、管理系・支援系のプロセスとして、システム・エンジニアリング・プロセス、サポート・プロセス、セーフティ・エンジニアリング・プロセスを新たに改訂、追記し、ESPR Ver. 2.0として発行することになった。ESPR Ver.1.0に引き続き、多くの方々に利用していただければと考えている。

最後に、巻末に本書の執筆に協力してくださった方々を記して深甚な謝意を表す。

2007年11月

組込みソフトウェア開発力強化推進委員会

執筆にあたりご協力いただいた方（敬称略）

| | |
|-------|---|
| 浅井真生雄 | 社団法人日本電気制御機器工業会 |
| 阿部功二 | 株式会社CSK システムズ |
| 猪狩秀夫 | IPA/SEC(横河デジタルコンピュータ株式会社) |
| 岩橋正実 | 三菱電機メカトロニクスソフトウェア株式会社 |
| 太田孝史 | 日本電気株式会社 |
| 大野克巳 | トヨタテクニカルディベロップメント株式会社 |
| 大金田光範 | 東芝システムテクノロジー株式会社 |
| 兼本茂 | 会津大学 |
| 佐藤吉信 | 国立大学法人 東京海洋大学 |
| 杉山英俊 | キャノン株式会社 |
| 鈴木利彦 | 株式会社ベリサーブ |
| 砂塚利彦 | 砂塚コンサルティングサービス株式会社 |
| 高野たい子 | 株式会社 日立製作所 |
| 龍野由紀男 | 株式会社 沖情報システムズ |
| 田邊安雄 | 株式会社 日本機能安全 |
| 田丸喜一郎 | IPA/SEC 組込み系プロジェクト サブリーダー (株式会社東芝) |
| 蔡光治 | ビジネスキューブ・アンド・パートナーズ株式会社 |
| 程子学 | 会津大学 |
| 友部雅章 | 横河電機株式会社 |
| 中川雅通 | 松下電器産業株式会社 |
| 長友優治 | 株式会社 ベリサーブ |
| 野中誠 | IPA/SEC(東洋大学) |
| 平尾裕司 | 長岡技術科学大学 |
| 平山雅之 | IPA/SEC 組込み系プロジェクト エンジニアリング領域幹事(株式会社東芝) |
| 藤村博司 | 日本電気通信システム株式会社 |
| 三浦邦彦 | 矢崎総業株式会社 |
| 水口大知 | 独立行政法人 産業技術総合研究所 |
| 向殿政男 | 明治大学 |
| 村松昭男 | 富士通株式会社 |
| 室修治 | IPA/SEC(横河デジタルコンピュータ株式会社) |
| 山崎太郎 | IPA/SEC(日本ユニシス株式会社) |
| 吉岡律夫 | 日本システム安全研究所 有限会社 |
| 渡辺雅人 | 株式会社CSK システムズ |

(50音順)

監修

組込みソフトウェア開発力強化推進委員会

編集・著作

独立行政法人 情報処理推進機構
ソフトウェア・エンジニアリング・センター

翔泳社 ecoProject のご案内

株式会社 翔泳社では地球にやさしい本づくりを目指します。

制作工程において以下の基準を定め、このうち4項目以上を満たしたものをエコロジー製品と位置づけ、シンボルマークをつけています。



| 資材 | 基準 | 期待される効果 | 本書採用 |
|--------|---------------------------------|---|------|
| 装丁用紙 | 無塩素漂白パルプ使用紙 あるいは 再生循環資源を利用した紙 | 有毒な有機塩素化合物発生の軽減（無塩素漂白パルプ） 資源の再生循環促進（再生循環資源紙） | ○ |
| 本文用紙 | 材料の一部に無塩素漂白パルプ あるいは 古紙を利用 | 有毒な有機塩素化合物発生の軽減（無塩素漂白パルプ） ごみ減量・資源の有効活用（再生紙） | ○ |
| 製版 | CTP（フィルムを介さずデータから直接プレートを作製する方法） | 枯渇資源（原油）の保護、産業廃棄物排出量の減少 | ○ |
| 印刷インキ* | 大豆インキ（大豆油を20%以上含んだインキ） | 枯渇資源（原油）の保護、生産可能な農業資源の有効利用 | ○ |
| 製本メルト | 難細裂化ホットメルト | 細裂化しないために再生紙生産時に不純物としての回収が容易 | ○ |
| 装丁加工 | 植物性樹脂フィルムを使用した加工 あるいは フィルム無使用加工 | 枯渇資源（原油）の保護、生産可能な農業資源の有効利用 | ○ |

*：パール、メタリック、蛍光インキを除く

【改訂版】組込みソフトウェア向け 開発プロセスガイド

2007年11月19日 初版第1刷発行

編者 独立行政法人 情報処理推進機構
ソフトウェア・エンジニアリング・センター
(<http://sec.ipa.go.jp/>)

発行人 佐々木幹夫

発行所 株式会社翔泳社 (<http://www.shoeisha.co.jp/>)

印刷・製本 大日本印刷株式会社

© 2007 IPA

本書は著作権法上の保護を受けています。本書の一部または全部について（ソフトウェアおよびプログラムを含む）、株式会社翔泳社から文書による許諾を得ず、いかなる方法においても無断で複写、複製することは禁じられています。

本書へのお問い合わせについては、ii ページに記載の内容をお読みください。

落丁・乱丁はお取り替えいたします。03-5362-3705 までご連絡ください。

ISBN978-4-7981-1563-4

Printed in Japan