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］

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

SE
SHOEISHA

本書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@shoeisha.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、記述個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

はじめに

上梓によせて

今を遡ること20年近く前、まだ組込みソフトウェアという分野が確立していない時代がありました。そのころのマイコンソフトウェアの開発は規模も小さく、比較的少人数での開発が主流であり、プロジェクトマネジメントが多少お粗末でもきちんとしたものづくりができていました。しかし、この20年の間に組込みソフトウェアの規模は増大し、多人数を投入するプロジェクト型の開発が主流になってきました。そして、こうした規模の大きな組込みソフトウェア開発でプロジェクトの運営の失敗などが散見されるようになってきています。一般的に物事に携わるメンバーがある一定人数以上になると、それを組織的に動かすための術（プロジェクトマネジメント）を施す必要があります。しかし、組込みソフトウェア開発の場合、それを円滑に進めるマネジメント術の整備と適用が規模拡大に追いついていないように見えます。

本書はこうした背景をもとに、組込みソフトウェア開発プロジェクトを円滑に進めるためのマネジメント術の手始めとして、プロジェクト運営のベースになるプロジェクト計画書作成の雛形を提供するものです。どのようなプロジェクトでも、きちんとした計画があって初めて円滑に進むものです。この『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］』は経済産業省と独立行政法人 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（所長：鶴保征城）の協力による組込みソフトウェア開発力強化推進委員会（委員長：門田浩、副委員長：田丸喜一郎）の活動の一環として設立された、プロジェクトマネジメント技術部会での議論を整理したものです。

本ガイドは、この部会に参加いただいた委員の方々から、実際の開発現場での豊富なプロジェクトマネジメント経験やノウハウを紹介いただき整理したもので、これからプロジェクトマネジメントを実践される方々にとっても大いに参考にしていただけると思います。本書の編纂に際し、委員会での議論や編集に協力いただいた方々には改めて感謝したいと思います。

ぜひ、組込みソフトウェア開発をされる際に一度は眼をとおし、円滑なプロジェクト運営の助けにしていただき、多くのプロジェクトの成功へのきっかけの一つにしていただければ関係者一同、望外の喜びです。

2006年秋
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、室修治、山崎太郎

プロジェクトマネジメントガイド［計画書編］発刊にあたって

近来、組込みソフトウェアは急速な発展を遂げ、組込みソフトウェア開発の成否が製品開発全体の成否を決めているといっても過言ではありません。このため、組込みソフトウェア開発のプロジェクトマネジメントに大きな注目が集まっています。

このプロジェクトマネジメントは、近来になってその知識体系が整備され、エンジニアリングとしての研究が盛んに行われるようになってきました。しかしながら、組込みソフトウェア開発へのプロジェクトマネジメントの適応は、その他の領域に比べて、必ずしも進んでいるとはいえない状況です。

プロジェクトマネジメントの第一歩は、きちんとした計画書を作るところから始まります。大規模化したプロジェクトであれば、誰が何を担当し、いつまでに何を作って誰に渡すのか、また、短期なプロジェクトであれば、どのように急いで開発を進めるのか、開発メンバーだけでなく、プロジェクトの関係者全員で十分に共有してからプロジェクトを着手することが重要です。これにより、クリティカルパスが明確になり、プロジェクト実行中に起こりえるさまざまな問題に対して、効果的な対策を行うことができるようになります。結果として、プロジェクトの後戻り作業は最小化され、プロジェクトを成功に導くことが可能となります。

このたび、経済産業省　組込みソフトウェア開発力強化推進委員会では、産官学連携の取り組みの一環として、『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］』を編纂することにしました。編纂にあたっては、国際標準などのプロジェクト開発計画書を元に、成功した組込みソフトウェア開発プロジェクトから、ベストプラクティスを抽出し、組込みソフトウェア開発に適した内容を盛り込みました。また、できる限りそのままの形で実務にて使用することができる帳票を使用し、かつサンプルや作成テクニックなどを含めることで、忙しい開発現場であっても、わかりやすいガイドの作成を目指しました。

本書により、プロジェクトマネジメントの組込みソフトウェア開発適応に関して、必要な適応レベルや、適応方法、適応による効果などを示すことができたと自負しております。これにより、組込みソフトウェア開発においても、プロジェクトマネジメント導入が進み、プロジェクトの成功に寄与できれば、この上ない喜びです。また、本書を機として、組込みソフトウェア開発プロジェクトに関する、さまざまな実務慣行やデータ取得が行われ、組込みソフトウェア開発プロジェクトに関する技術や知識がますます蓄積されていくことを切に願っております。

2006年11月
組込みソフトウェア開発力強化推進委員会

まえがき

近年、組込みソフトウェアで実現する機能の増大に伴い、ソフトウェアの規模が急激に増大しています。この結果、これらのソフトウェアを開発する開発プロジェクトも肥大化の傾向が続いており、プロジェクトの円滑な運営が滞ることも少なくありません。こうした課題を解決し、開発プロジェクトを適切な方向に導く技術としてプロジェクトマネジメント技術があります。プロジェクトマネジメント技術では、プロジェクトの着手時点から完了時点を通して、製品として提供するソフトウェアの品質、コスト、納期を適切に制御し目標値に近づけていくための技術です。このプロジェクトマネジメントの中でも特に重要となるのが、そもそもの目標値をきちんと決めるためのプロジェクトマネジメントのベースとなるプロジェクト計画を作成する作業です。本ガイドはこのプロジェクト計画を立てるための基本的な考え方や参考となる計画書の雛形を、**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]（ESMR Ver 1.0：Embedded System development Management 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

本ガイドは組込みソフトウェア開発に従事する開発リーダ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計画書を作成する際の参考にしていただくことを目的としています。

本ガイドでは組込みソフトウェア開発プロジェクト計画書の作成を目的として、以下に示す4つのパートから構成されています。

Part 1：解説編
Part 2：技術編
Part 3：事例編
Part 4：参考情報

SEC 関連プロジェクトとの関係

SECでは組込みソフトウェア開発を円滑に進めることを目標に「組込みソフトウェア向け開発プロセスガイド（ESPR）」を策定・公開しています。本ガイドは、このプロセスガイドで定めている支援プロセスカテゴリ（SUP）の「SUP1 プロジェクトマネジメント」の作業に深く関わるものです。ぜひ、開発プロセスガイドもご一読ください。

備考

本ガイドは、経済産業省 組込みソフトウェア開発力強化推進委員会 エンジニアリング領域 プロジェクトマネジメント技術部会で検討を重ね整理したものです。

目次

Part 1
解説編

ここでは、本書を読むにあたっての導入として、組込みソフトウェア開発におけるプロジェクトマネジメントや開発計画書の役割、本書のねらいや構成、想定読者などを説明します。

1.1　開発計画書に基づくプロジェクトマネジメントとは

組込みソフトウェア開発プロジェクト

現在、多くの企業でさまざまな組込みソフトウェア開発が行われています。組込みソフトウェア開発プロジェクトとはどのようなものか、簡単にその特徴を整理してみます。

組込みソフトウェア開発プロジェクトの特徴

近年、ソフトウェア開発では、その規模の増大により、開発を主目的としたプロジェクトを組織して開発にあたることが多くなってきています。

開発プロジェクトという視点で最近の組込みソフトウェア開発をみると、いくつかの傾向があります。

①メンバーの多様性

プロジェクトに多様で数多くのメンバーが関係している。

②短期集中型のプロジェクト

プロジェクトの期間やスケジュールが極めてタイトなものが多く、短い期間に複雑な組込みソフトウェアの開発が求められている。

③制約の多様性

プロジェクトを運営していく上で多様な制約条件があり、これらへの適切な対処が求められる。

メンバーの多様性

組込みソフトウェア開発では、その関係者は単純にソフトウェア開発グループ内にとどまることは少なく、関連するハードウェア開発グループや製品やセット全体を取りまとめる部署などさまざまなメンバーが関係してきます。

このように多様なメンバーが開発プロジェクトに関係するということは、プロジェクトを運営していく上で、それらの多様なメンバー間の調整などが必要になることを意味しており、これらを考慮した適切なプロジェクトマネジメント・システムの構築と実施が必須要件になります。

短期集中型

組込みソフトウェアの多くは、いわゆるコンシューマプロダクトという形態で製品に組み込まれて出荷されます。最近の先端技術を駆使した製品の場合、製品のマーケット投入タイミングはそのビ

ジネス面において極めて重要な意味合いを持っています。このため、多くの組込みソフトウェア開発プロジェクトでは、プロジェクトの着手から完了までの期間が市場動向に応じて短くなる傾向にあり、かつ、製品の納期を意識した開発が求められています。このように短期の開発期間でかつ製品納期を確実に守るためには、開発プロジェクトの最適化を図り適切なマネジメントを実践する必要があります。

またこうしたコンシューマプロダクト以外で利用される組込みソフトウェアでも、その開発は短納期化が進行している場合も多く、同様に、適切な開発プロジェクトマネジメントが重要となってきています。

制約の多様性

上記以外にも組込みソフトウェア開発の場合、開発プロジェクトにさまざまな制約が加わる場合があります。たとえば、多くの製品では現在、シリーズ化された製品開発スタイルや多品種少量開発などの方式が採用されつつありますが、こうした開発方式を採用する場合の開発メンバーのアサインや開発スケジュール調整なども重要になります。

また、一部の製品では、その内部をいくつかに分割し、複数の企業に分けて開発したり、その一部を海外発注するといったスタイルもあります。このような場合には、製品開発全体のマネジメントはよりいっそう難しくなります。

開発プロジェクトマネジメントの必要性

このように最近の組込みソフトウェア開発プロジェクトはその運営が極めて難しくなってきています。こうしたプロジェクトを円滑に運営する術が「プロジェクトマネジメント」技術です。

開発プロジェクトマネジメントのポイント

実際のプロジェクトのマネジメントでは、以下のような点を特に意識して進める必要があります。

①対象製品やプロジェクトの特性に合致したマネジメントの方式を採用する。
②プロジェクトマネジメントではプロジェクト着手時のプロジェクト計画をベースとして、プロジェクトの実際をそれに近づけていく活動を行う。
③対象プロジェクトの状況が常に見えるように、プロジェクト状況の可視化の工夫などを織り込んでいく。
④対象プロジェクトの完了時点で、プロジェクトで実施したマネジメント方式や対策の効果などを分析し次のプロジェクトへの参考知識として整理する。

開発プロジェクトマネジメントにおけるPDCAサイクル

組込みソフトウェア開発プロジェクトに限らず、どのようなプロジェクトであっても、それを成功裏に終わらせるためにはPDCAサイクルをきちんと回すことが重要になります。

PDCAサイクルとは、Plan（計画立案）、Do（計画実行）、Check（実行状況の確認）、Action（是正対策の実施）といった形で、一連の動作を適切に実行するための考え方です。この考え方は、システム開発プロジェクトにおいても同様で、プロジェクト開始時点できちんとプロジェクトの計画を立て、それに基づいた開発作業や管理作業を進めていくことが基本となります。このような開発プロジェクトのPDCAサイクルを適切にまわすためには、まず、出発点となるプロジェクト計画書がきちんと作成されている必要があります。

プロジェクト計画書の必要性

このように組込みソフトウェア開発プロジェクトを適切に運営していく上で、プロジェクトの計画書はマネジメント活動のベースになるものです。

すなわち、プロジェクトの計画段階で、そのプロジェクトをどのように運営していくかについて、きちんと検討し決めておかないと、そのプロジェクトが計画通りに進んでいるのか、あるいは、プロジェクトの進行上で問題があるのかなどを判断することができなくなります。この点において、プロジェクトを円滑に進めるためにはプロジェクト計画書は必須のものといえます。

プロジェクト計画書の目的

ソフトウェア開発を進めるにあたっては、その規模の大小によらず、多くの場合、チームやプロジェクトとしての開発が中心になります。こうした開発を進める上では、それぞれの作業のベースとしてのプロジェクト計画が策定され、計画書として文書化（見える形に整理）されている必要があります。その上で、プロジェクト計画書は下記のような目的に利用されます。

①プロジェクトの初期段階で、プロジェクトの実現可能性を判断する際の判断材料として利用できます。
②プロジェクト内外の利害関係者間での共通理解のベースとして利用でき、また、それぞれのコミットメントを確立する際にも利用できます。
③プロジェクトのコントロールをするための基本計画として利用できます。
④プロジェクトの予実の差分を把握し、是正措置をとる際の参考とすることができます。

プロジェクトはその進行とともに、さまざまな事象が発生します。このため、計画書を作成し、また必要と判断する場合には随時見直しを行うことで、当該プロジェクトを成功に導く道しるべとしての役割を持っています。

1.2　本ガイドの目的

プロジェクト計画書作成上の問題点

プロジェクト計画書を書くことは、プロジェクトとしてシステム開発を進める上での出発点となる重要な作業です。しかし、この「プロジェクト計画書を書く」という作業に直面したことのある方は、いろいろな点で悩んだのではないかと思います。「プロジェクト計画書を書く」にあたっては、以下のような、さまざまな問題があります。

- 計画書にどのような項目を書いていいかわからない。
- 部門として共通化された計画書のフォーマットなどが整備されていない。
- 複数部門との協業でプロジェクトを進める場合に、部門や組織によって計画書の書き方が異なっている。
- 計画書に書くべき項目はわかったとしても、具体的にどのように項目を埋めていくかがわからない。
- そもそも計画書をいつ、誰が書くかが決まっていない。

などなど挙げれば際限なくいろいろな問題がでてきます。結果として、実際の開発現場、特に組込みソフトウェア開発プロジェクトでは、開発計画（書）があいまいなままに、開発プロジェクトが進められる場合が少なくありません。

組込みソフトウェアプロジェクトで観察される状況

特に組込みソフトウェアの開発プロジェクトでは、開発スケジュールなどが極めてタイトであったり、開発途中で要求仕様や開発の制約条件が変わったりする場合が少なくありません。こうした事象が頻発するプロジェクトや部門では、

- 「計画書など作成しても意味がない」
- 「書いてもすぐに変更になるので面倒」

といったことを理由にプロジェクトの開発計画書が作成されない場合が少なくありません。しかしながら、こうした場合、多くのプロジェクトでは計画不在の行き当たりばったりのプロジェクトとなってしまい、プロジェクトはより悪い方向へ暴走してしまいます。プロジェクト計画書は、このようなプロジェクトこそきちんと立ち止まって考え、状況に応じて見直しをかけていくことで、プロジェクトを成功に導くための有効なツールであるといえます。

本書の目的

こうしたプロジェクト計画書作成に関するさまざまな問題を解決し、システム開発プロジェクトにかかわる方々が、きちんとしたプロジェクト計画書を作成し、それにのっとったプロジェクト運営を進めていただくことを目的として、本書を作成しました。本ガイドの特徴は以下に示すとおりです。

本書の特徴

①国際標準への準拠

国際標準などに準拠した形で開発計画書の目次構成と記載項目を定めています。近年、組込みソフトウェア開発でもきちんとした仕組みでプロジェクトマネジメントが実施されることが求められる傾向にあり、一部では、そのために国際標準などへの準拠を求められるケースもあります。本ガイドを参照利用することで、こうした国際標準に準拠した、言わば、国際的にも通用するプロジェクト計画書を作成することが可能となります。

②個別項目の記載内容に関する詳細を定義

計画書に盛り込むべき個別の項目について、そこで記載すべき事項や記載の際に注意すべき事項などを整理して掲載してあります。

③計画書の標準記述フォームを掲載

開発の現場ですぐに利用していただけるように、計画書の標準記述フォームを掲載しています。なお、ここでの標準とは、上記の国際標準や国内の平均的な企業などの計画書を参考に、計画書記載項目を策定しており、これを標準記述フォームとして位置づけています。

④開発計画書の事例を掲載

上記の記述フォームを活用した実際の開発計画書の事例を掲載してあります。この事例を参考に、プロジェクトの開発計画書の具体的な形を理解することが可能です。

⑤開発プロセスと連動

SEC で整理した組込みソフトウェア向け開発プロセスガイド（ESPR）で定めた開発プロセスとリンクする形で、個々の作業フェーズの計画を立てられるように工夫されています。詳細はESPR の SUP1（プロジェクトマネジメント）をご覧ください。

なお、本ガイドに掲載した計画書テンプレートは別途、SEC ホームページよりダウンロードが可能となっています。

1.3　想定する利用者・利用方法と得られる効果

想定する開発プロジェクト

本ガイドが想定する組込みソフトウェア開発のプロジェクトは以下のとおりです。

組込みソフトウェア開発プロジェクト

本ガイドは組込みソフトウェアの開発を対象としています。本書でいう組込みソフトウェアとは「製品に組み込まれて動作するソフトウェア」全般を指します。これらの組込みソフトウェアでは、さまざまな機器やデバイスの制御を司る制御的な側面や、これらの機器で利用されるさまざまな情報を処理したりする情報処理的な側面などを併せ持つ場合がありますが、これらも含めた形での組込みソフトウェアを対象としています。

なお、本ガイドではこうした組込みソフトウェア開発における特徴や注意すべき事項などを考慮して開発計画書のテンプレートを策定しています。

想定するビジネスの形

組込みシステムには、家電製品や自動車などのいわゆるコンシューマプロダクトとして一般ユーザに提供される製品で利用されるものと、さまざまな社会インフラや産業機器などに搭載され、特定の専門オペレータが利用する場合があります。本ガイドは対象とする組込みソフトウェアに関して、特に、これらの視点は区別していません。このため本ガイドの利用に際しては、個々の開発や製品のビジネス特性を考慮し、プロジェクト計画書の内容を吟味していただく必要があります。

想定する開発形態や体制

組込みソフトウェアの開発では、その開発規模やソフトウェアの特性によってさまざまな開発形態や体制で開発が進められています。本ガイドは特定の開発形態や開発体制を意識したものとはなっていません。組込みソフトウェアを開発するプロジェクトやグループで、開発に先立って検討するプロジェクト計画書に盛り込むべき事項を整理したものです。このため、個々の開発において、本ガイドで示す計画書記載事項を参考に、それぞれに適したプロジェクト計画書を作成していただく必要があります。

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開発に関係する下記のような方々を主たる利用者として想定しています。

①組織内の開発プロジェクトのプロジェクト計画書の標準形を整備し統一する役割を担ったソフトウェア開発支援グループのメンバー
②自らが担当する開発プロジェクトの特性に合わせて、開発プロジェクトの開発計画を立案する役割を担ったプロジェクトマネージャやリーダ

なお、小規模な組織の場合、こうした役割分担が明確になっていない場合もありますが、その場合にも、これらの役割を担う技術者の方々に参考にしていただければと考えています。

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関して、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計画を策定する際に利用することを想定しています。

①プロジェクト計画書の書式や書き方が未整備な部門や組織において、部門や組織の標準的なプロジェクト計画書作成方法を決める場合
②既に部門や組織の標準的なプロジェクト計画書の書式などがあるが、実際のプロジェクト計画書とのずれやギャップがあり、見直しが必要な場合
③複数部門や組織が関係する開発プロジェクトにおいて、組織をまたいで、プロジェクト計画書の統一を図る場合

実プロジェクトの開発計画書の策定

基本的に本ガイドのプロジェクト計画書標準定義書をそのまま利用するというよりは、これをもとに策定した自組織の標準的なプロジェクト計画書書式を利用する形となりますが、具体的な実プロジェクトでのプロジェクト計画書作成に関しては、Part 2の活用方法に示すように、計画書の内容を段階的に詳細化していくことが必要になります。

利用にあたっての留意事項

通常、プロジェクトの開発計画書は、1つの独立したドキュメントとしてまとめておくほうが、後々の管理などが容易になります。したがって、本書では、Part 2以降に示す計画書記載項目を1つの文書として盛り込んだ「プロジェクト計画書」を策定することを想定しています。

しかし、個々の開発プロジェクトの事情などにより、これらの情報を複数の文書に振り分けて整理する場合もありますが、こうした計画書スタイルを採用した場合にも、計画書として盛り込むべき項目は、それほど大きく変わらないため、本書を十分に参考にしていただけます。

簡易的な利用方法

本ガイドで示すプロジェクト計画書の記載項目ならびにテンプレートは、組込みソフトウェア開発を進める上での一般的にプロジェクトの開発計画書に欠かせない要件を盛り込んだものです。

しかしながら、開発プロジェクトによっては、数人のお互いの情報交換が密にできるような環境下で行われる場合もあり、こうした場合には、本ガイドで示すような多岐に渡る項目を計画書として整理しなくともプロジェクトの運営は円滑に進むことが考えられます。このため、本ガイドで記載した計画書の記載事項や標準記述フォームについては、適宜、対象プロジェクトの特性などを考慮して、カスタマイズして利用していただくことを想定しています。

得られる効果

本ガイドを利用することで得られる効果は下記のとおりです。

開発計画書のモレやヌケが少なくなる

本ガイドを利用して組織内の標準的なプロジェクト計画書のスタイルを決めることで、プロジェクト計画書の項目レベルでのモレやヌケを少なくすることができるため、結果として作業の後戻りも少なくすることができます。

他部門との調整が容易になる

複数部門や組織が関係するプロジェクトの場合、関連する部門や組織のすべてでプロジェクト計画書のベースを本ガイドに置くことで、それぞれの開発計画のすりあわせが容易になります。

開発計画データの再利用が容易になる

プロジェクト計画書の書式を統一することで、過去のプロジェクトの計画（工程計画や作業計画）あるいは、個々の作業量や期間などのデータを再利用することができます。

1.4　本ガイドの構造

全体構造

本ガイドは本章を含めて4つのパートから構成されます。

Part 1　解説編

本ガイドの位置づけや想定利用者、利用方法などの概要を記載してあります。プロジェクトの開発計画書の位置づけや役割などを十分に理解できていない方、あるいは、プロジェクトの開発計画書をはじめて作成される方などは、まず、このパートを読んでいただければ、その意味付けなどをご理解いただけます。

Part 2　技術編

本ガイドの中心となるプロジェクトの開発計画書の標準記述フォームとその記載内容の解説を掲載してあります。このパートで記載する計画書の記載項目と標準記述フォームは、国際規格や国内の企業などで利用されている開発計画書をもとに、検討して定めたものです。個々のプロジェクトや組織の特性に応じて、カスタマイズして利用していただくことを想定しています。

Part 3　事例編

Part 2で掲載した標準記述フォームの利用例として、サンプルプロジェクトにおけるプロジェクト計画書の事例を掲載してあります。

また、経済産業省が実施した産業実態調査のデータをもとにした我が国の組込み産業における平均的なプロセス像を想定し、計画書に盛り込む具体的な参考値などを掲載してあります。

Part 4　参考情報

計画書を書く際に必要となる作業分割の方法や、見積りの考え方など具体的なテクニックを記載してあります。これらは、あくまでもテクニックの1つとして紹介するものであり、他にもさまざまな手法やテクニックが存在します。プロジェクトや組織の特性に応じた使い分けや、他の手法なども検討いただき、自組織に適した方式を採用していただければよいでしょう。

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の開発プロジェクトにおけるプロジェクト計画書の標準化を進めるためのドキュメントであり、そのため、

①プロジェクト計画書に盛り込むべき事項のプロジェクト計画書標準記述フォーム
②実際の開発プロジェクトでの適用事例

などを参考情報として掲載してあります。

開発プロジェクトの開発計画書については、IEEE 1058 (1998) などで規定がなされており、本ガイドの作成段階でもこれらを参考としています。しかしながら、本ガイドはその名称が示すとおり、組込みソフトウェア開発プロジェクトに携わる皆様の参考情報として利用いただくことを目的に整理したものです。したがって、本ガイドの活用方法として、このガイドに関するいかなる認証や認定といった枠組みを想定するものではありません。

プロジェクト計画書定義書＆標準記述フォーム

本ガイドのPart 2で示すプロジェクト計画書定義書ならびに標準記述フォームは、本ガイドに関する上記の趣旨を踏まえ、あくまでも参考情報として整理したものです。したがって、これらの情報の利用方法も含め、特定の利用・参照方法を指定するものではありません。

プロジェクト計画書例

本ガイドのPart 3で示すプロジェクト計画書例は、上記のプロジェクト計画書標準記述フォームの具体的な利用方法を示すことを目的としています。記載された具体項目はあくまでも事例です。

1.6　関連する規格など

国際標準他

ソフトウェアプロジェクトの開発計画書については以下のような規格などが関連します。

IEEE 1058

すべてのソフトウェアプロジェクト管理計画に盛り込むべき要素を記載しており、ソフトウェアプロジェクト管理計画の内容を規定しています。

SPMP：Standard for Software Project Management Plans

ISO/IEC 16326

ソフトウェア開発プロセスを規定したISO/IEC12207に関して、そこで規定されているマネジメントプロセスに関する導入の指針を規定しています。

国際規格と本ガイドとの関係

本ガイドの策定にあたっては、これらの国際標準類を参考として、その内容を吟味・咀嚼して反映しています。ただし、これらの国際規格のほとんどは、組込みソフトウェアの開発プロジェクトを明示的に意識した形とはなっていないため、本ガイド策定にあたっては、経済産業省 組込みソフトウェア開発力強化推進委員会 エンジニアリング領域 プロジェクトマネジメント技術部会での専門家の意見を反映する形で、組込みソフトウェアを念頭に解釈をした部分が含まれています。

このため、上記の国際規格に対する厳密な準拠性などを求められる開発プロジェクトにおいては、あくまでもこれらの規格を参考にされることをお勧めします。

Part 2
技術編

ここでは、組込みソフトウェアの開発プロジェクトを対象に、そのプロジェクト計画書に記載するべき事項や注意点、標準的な記述フォームなどを紹介します。

全体構成

Part 2では、組込みソフトウェアの標準的な開発プロジェクトを想定し、そこで作成すべきプロジェクト計画書の項目とその記載内容・記載方法などを規定しています。プロジェクト計画書に盛り込むべき主な記載項目には、以下のようなものがあります。

- **プロジェクト計画の概要情報**
  この部分を読むことでプロジェクト計画の概要レベルでの全体把握ができるようになります。
- **プロジェクト運営に関する情報**
  プロジェクトの体制やメンバーの役割と責任などプロジェクトの運営面からの詳細計画を把握できるようにします。
- **プロジェクト作業などに関する詳細情報**
  プロジェクトのQCD（Quality, Cost, Delivery）に関する当初計画や具体的な作業の詳細に関する計画を把握できるようにします。特に、予算計画（Cost）と作業計画（Delivery）、品質保証計画（Quality）の3点に関して具体的に計画を策定し記載します。

プロジェクト概要情報	Chapter 1　プロジェクトの概要	
	1.1　プロジェクトの目的 1.2　プロジェクトの目標 1.3　目標達成のための方針・手段 1.4　プロジェクトの範囲	1.5　プロジェクトの前提条件 1.6　プロジェクトの成果物 1.7　スケジュールと予算 1.8　計画の更新
参照情報	Chapter 2　参照・定義	
	2.1　参照	2.2　定義
プロジェクトの体制	Chapter 3　体制	
	3.1　製品開発プロジェクトの体制 3.2　外部インタフェース	3.3　ソフトウェア開発プロジェクト内部体制 3.4　役割分担
プロジェクト詳細計画	Chapter 4　リソース計画	
	4.1　開発規模と工数の計画 4.2　人員計画 4.3　設備、機器等の調達計画	4.4　プロジェクトの人員研修計画 4.5　予算計画書
	Chapter 5　作業計画	
	5.1　開発作業の洗い出し 5.2　開発作業の順序付け	5.3　開発作業担当者の割付 5.4　作業計画
	Chapter 6　品質保証計画	
	6.1　品質目標 6.2　品質保証の体制と仕組み	6.3　品質保証に関する主要なイベント
	Chapter 7　リスクマネジメント	
	7.1　リスクマネジメントの方針と仕組み	7.2　リスク一覧表

1　プロジェクトの概要

プロジェクト計画書はプロジェクトを遂行する中のさまざまな局面でさまざまな関係者が手に取り、さまざまな使い方をします。たとえば、

①プロジェクトマネージャが担当するプロジェクトの状況確認のベースとして利用する。
②関連する部門のマネージャや担当者が関連部門の計画や状況を把握するために用いる。
③経営者や管理者が経営の視点からの判断を下す材料にする。

というようにさまざまな利用方法が考えられます。このようにプロジェクト計画書に関連するステークホルダは多岐に渡りますが、それぞれによって、必要とする情報は異なっています。しかし、どのようなステークホルダであっても、少なくとも、当該のプロジェクト計画書がどのようなプロジェクトを対象に作成されているかといった全体概要のレベルでの把握は必須になります。このため、プロジェクト計画書の冒頭では、開発計画の詳細まで入らずに、数ページでプロジェクト開発計画の概要が把握できるようにプロジェクト概要情報を簡潔に整理しておく必要があります。

なお、プロジェクト概要情報は、本来、Chapter 3、4、5、6、7で記載する情報をサマライズした情報になりますが、Chapter 3、4、5、6、7はプロジェクトのキックオフ後、段階的に検討・詳細化される情報です。このため、プロジェクト概要情報については、

第1バージョン：開発プロジェクトのキックオフ時、主要な関係者が確定してプロジェクトの細部を決定した時点
第2バージョン：開発対象システムで実現する機能などがほぼ確定し、プロジェクトとして実施する主要な作業が把握できる時点

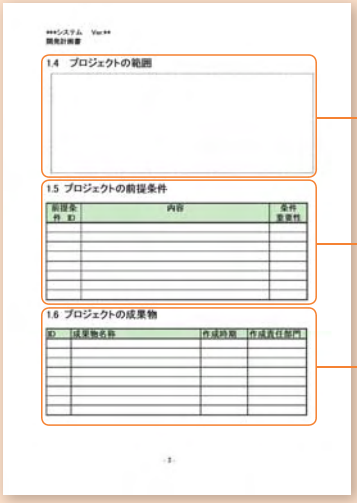
Chapter 1　プロジェクトの概要		
Section	タイトル	概要
1.1	プロジェクトの目的	対象となる製品（組込みシステム）開発全体の中における本プロジェクト（組込みソフトウェア開発）の目的を明確にする。
1.2	プロジェクトの目標	プロジェクトの達成目標を特にQCDの観点から具体的に記述する。
1.3	目標達成のための方針・手段	目標達成のための手段を具体的に記述する。
1.4	プロジェクトの範囲	プロジェクトの成果・責任の範囲を明確にする。
1.5	プロジェクトの前提条件	プロジェクトを遂行する上で、現実に起こると予想される条件を、経営層あるいは市場との合意された仮定条件や制約条件を記載する。
1.6	プロジェクトの成果物	プロジェクトの成果物およびその目標を明確にし、内容や定量的な達成目標を具体的に記述する。
1.7	スケジュールと予算	プロジェクトの時間的な枠組み、マイルストーン、予算について記述する。
1.8	計画の更新	計画を更新する場合の手順（レビュー、承認、配布等）を記載する。

Document Format Image

1.1 プロジェクトの目的

1.2 プロジェクトの目標

1.3 目標達成のための方針・手段



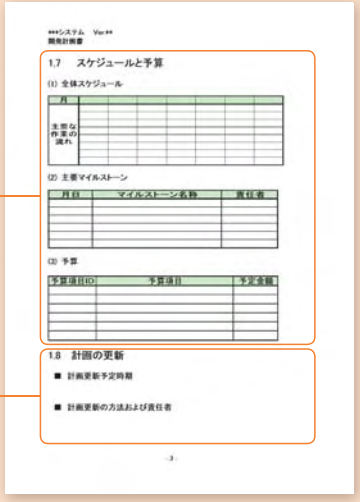
1.4 プロジェクトの範囲

1.5 プロジェクトの前提条件

1.6 プロジェクトの成果物

1.7 スケジュールと予算

1.8 計画の更新



1.1 プロジェクトの目的

対象となる製品（組込みシステム）開発全体の中における本プロジェクト（組込みソフトウェア開発）の目的を明確にする。

● 記述すべき内容

- この組込みソフトウェア開発プロジェクトがどの製品、システムにどのように関係するかを明確にする。
- 開発対象の組込みソフトウェアが搭載される製品あるいはシステムがどのような位置づけ、どのような役割や目的を持っているかを明確にする。
- 想定されるユーザ、顧客や市場に関して開発プロジェクトに関係するであろう特徴などを明確にする。

● 記述の際に注意／考慮すべき内容

- ハードウェアとともに実現すべき機能等を明確にしておく。
- プロジェクト概要を把握するためにプロジェクトの位置づけを明確にする観点から、経営層・開発メンバー・外部の関係者で共通に理解できるような具体的な記述を心がける。

1.2 プロジェクトの目標

プロジェクトとして達成が求められる開発コスト（Cost）、開発期間（Delivery）、製品品質（Quality）などについて具体的に記述する。

● 記述すべき内容

- 開発コスト面での目標値
- 開発期間やソフトウェアのリリースなどの目標値
- 開発するソフトウェアの品質に関する目標値（性能・機能なども含める）
  また出荷後の品質についての目標も記述する。
- 上記以外の目標についてもプロジェクトの目標であれば記述する（例：特定の技術獲得、新規分野への参入、…）。

● 記述の際に注意／考慮すべき内容

- 関係者全員が理解できる用語を使用する。
- 具体的な数値で達成度の判断ができるようにしておく（must、want、challenge等の表現も可）。
- 優先度を下げる場合にも最低線は明確にする。
- プロジェクトによっては技術獲得やメンバーのスキル向上などを副次的な目標におく場合もあるため、これらについても必用に応じて記述しておく。

1.3 目標達成のための方針・手段

目標達成のための手段を具体的に記述する。

● 記述すべき内容

- 目標を達成するために特別に考慮・実施しなければならない開発上の留意点・項目や方針を記述する。
- 組織・プロジェクト体制を特別に編成する場合はそれを明記する。
- 目標達成のための手段を具体的に記述する。

● 記述の際に注意／考慮すべき内容

- QCD目標を達成するために、ソフトウェア開発プロジェクト内で工夫すべき事項、ハードウェア開発部門など製品やシステム開発に関する関連部門との連携において開発プロセス面や開発工程面で工夫すべき事項などを明確にしておく。

  (例) ハードウェア性能確認のため、ソフトウェア開発メンバーとハードウェア開発メンバーが共同で作業する特別なチームを編成する。
- QCD目標達成のための手段として、過去のソフトウェア資産や開発プロセス上の工夫なども最大限に活用する。

1.4 プロジェクトの範囲

プロジェクトとして開発するプロダクト（成果物）の範囲と開発組織の範囲（責任範囲）を明確にする。

● 記述すべき内容

- 本プロジェクト（組込みソフトウェア開発）で開発する組込みソフトウェアが、搭載される製品やシステムの中でどのような位置付けになるかを明確にする。
- 製品やシステムを構成するハードウェアユニットやメカニカルユニット、あるいは既開発で今回の開発範囲に含まれないソフトウェアユニット、あるいは、この先のバージョンアップなどで開発されるであろうソフトウェアユニットなどとの関係を明確にする。
- 開発組織の視点から当該プロジェクトが関連する他のプロジェクト（例：ハードウェア開発プロジェクト、他のソフトウェア開発プロジェクト）との関係（それぞれのプロジェクトの境界、インタフェース）を明確にする。
- 開発当事者としてのプロジェクトの成果や責任の範囲を明確にする。
- 本プロジェクトの開始と終了を明確にする。

● 記述の際に注意／考慮すべき内容

- 関連するプロジェクトは漏れなく記載する。
- 図や表を用いて分かりやすく表現する。
- 診断・検査用ソフトウェア、マニュアル作成等も開発対象の場合には明記する。

1.5 プロジェクトの前提条件

プロジェクトを正常に遂行するために、必要な条件と外部への約束事、プロジェクト推進に関する制約事項などを列挙する。

● 記述すべき内容

- プロジェクトを遂行する上で、現実に起こると予想される条件を、経営層あるいは市場との合意された仮定条件や制約条件とともに記載する。
- プロジェクト遂行における制約条件をリストアップする。その際、制約条件の優先順位も明確にする。
- プロジェクトを遂行する上で、現実に起こると予想される条件をリストアップする。
- プロジェクトにおける

  スケジュール（納期）　予算　リソース（外部委託も含む）
  ソフトウェア（再利用ソフトウェア含む）　利用技術　他製品へのインタフェース
  ハードウェア（CPU性能など）　開発環境（テスト環境も含む）

  などに課せられている制約条件を明確にする。
- 優先順位も明確にする。

● 記述の際に注意／考慮すべき内容

- 計画段階での未決事項も明記する。
- プロジェクトの前提条件や制約条件は「プロジェクトの内部要因に起因する制約」「プロジェクトの外部要因に起因する制約」の2種類が存在する。
- 「プロジェクトの外部要因に起因する制約」の中には、搭載される製品やシステムに関するビジネス面の制約事項や、ソフトウェア開発プロジェクトの外部に位置するハードウェア開発プロジェクト側の制約事項なども含まれる。

  （例）製品のバージョンアップなどビジネス上想定されるシナリオとそれによる制約条件についても記述する。
- 「プロジェクトの内部要因に起因する制約」は、たとえば、開発メンバーや組織に関する制約なども含まれる。
- ここで記載した前提条件はプロジェクト実施段階でのリスクマネジメントの中で常時確認し、適宜、プロジェクトの軌道修正の判断材料とする。

プロジェクト前提条件表

前提条件を識別できる番号をつける

考慮すべき優先度や条件の重要度などを評価しておく

前提条件ID	内容	条件重要性

計画の前提となる条件あるいはプロジェクト遂行上の制約条件などをリストアップする

1.6 プロジェクトの成果物

プロジェクトとして作成する主要な成果物について、名称、作成時期や作成責任者を明らかにする。

● 記述すべき内容

- ソフトウェア、仕様書、マニュアル等のドキュメント類などプロジェクトで作成する主要な成果物（サービスを含む）を明確にする。
- これらの成果物をいつごろ、どの部門で作成するかも合わせて明示する。
- また、プロジェクトの成果物を完成させるために必要となる中間成果物についても、主だったものをリストアップする。なお、中間成果物とは、各プロセスにより作成されるアウトプットであり、最終成果物とは異なるものを指す。

● 記述の際に注意／考慮すべき内容

- プロジェクトの主要な成果物とその作成提供時期などはステークホルダ間で合意をとることを基本とする。
- また製品体系により納品物が決まっている場合には、成果納品のチェックリストなどで代用してもかまわない。
- 成果物の名称や作成工程（プロセス）などは、開発プロセスも考慮して決定する（SEC開発プロセス標準なども参照するとよい）。

予定成果物表

成果物を認識できる番号をつけておく

成果物を作成し完成させる時期（日付あるいは工程名など）を明らかにしておく

ID	成果物名称	作成時期	作成責任部門

作成予定の主要な成果物の名称を記載する

どの部門やグループ、担当者が作成するかも明らかにしておく

1.7 スケジュールと予算

プロジェクトの大日程計画（マスタスケジュール相当）のスケジュールを記載する。また、プロジェクト実行に要する全体的な費用（予算）についても記述する。

● 記述すべき内容

1. スケジュール

- プロジェクトの開始、終了の予定日、および、要求、設計、実装、テストといった大きなレベルでの工程単の日程スケジュール（大日程計画）を記載する。
- また、開発日程上の主要なマイルストーンは、日付、責任者も含めて記載する。
- プロジェクトに対する主要な支援プロセスや付加作業に関する日程についても記載する。

2. 予算

- 予算については、ソフトウェア開発に関する人件費、部材費、開発環境整備費用など大きなくくりで記載する。

● 記述の際に注意／考慮すべき内容

- スケジュール、予算はともに計画書のChapter 4、5の記載をサマライズした情報を記載する。
- プロジェクト提案時に作成する計画書では、Chapter 4、5に記載される詳細情報が存在しないため、過去の類似プロジェクトのデータなどを参考に概算の計画値を記載し、詳細な計画を検討した時点でより精度の高い情報に修正していく。

スケジュール概要表

月単位くらいで主要な作業や工程を明らかにしておく
工程や作業の担当グループが分かれる場合には、グループごとに把握できるように工夫しておく

月							
主要な作業の流れ							

月日	マイルストーン名称	責任者

主要なイベントやマイルストーンを日付・名称とあわせて明らかにしておく

イベントなどの実施責任者も明記しておく

概算予算表

必要な予算項目を識別できる番号をつけておく

概算の見積もり金額を記載しておく

予算項目ID	予算項目	予定金額

プロジェクト遂行上、必要な予算項目をリストアップしておく

1.8 計画の更新

計画を更新する場合の手順（レビュー、承認、配布等）を記載する。常に最新版を参照して作業できるようにする。

● 記述すべき内容

- 計画の見直し計画を記載する。
- 計画更新時の手順（レビュー、承認、配布等）を記載する。
- 最新版の所在を記載する。
- プロジェクト計画＆実績のレビュースケジュールを明確にし、レビュー時の計画更新のルーチンも明らかにしておく。

● 記述の際に注意／考慮すべき内容

- 承認者を明確にする。
- 計画書の状態（未承認、承認待ち、承認済等）を明確にする。
- 版数アップの意味、定義を明確にする。
- 更新内容、更新箇所の記載方法を明確にする。なお、ここでの「計画の更新」とは「計画を詳細化する」作業は指しておらず、あくまでもいったん確定した計画を見直しすることを指している。

2　参照・定義

通常、ソフトウェアの開発を進めていく中では、さまざまな情報が交錯します。こうした情報の中には、確かな情報もありますが、ときには、あいまいなあるいは怪しげな情報も飛び交うことがあります。こうしたさまざまな情報を相手にする場合、どのような出所のどのような情報を参照すべきかなどを明示しておくことが望まれます。

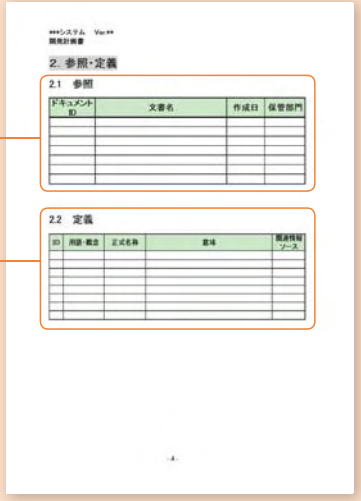
また、開発プロジェクト内では、これに付随してさまざまな技術用語やその考え方がメンバー間でやり取りされます。しかし、こうした用語や考え方の中には、開発にかかわるすべてのステークホルダ間で共通理解が得られていないものや、解釈がまちまちのものなども含まれており、往々にして誤解のもととなります。こうした事態を防ぐために、開発計画書やプロジェクトで使う用語などに共通の定義を与えておきます。

Chapter 2　参照・定義		
Section	タイトル	概要
2.1	参照	対象となる開発プロジェクトの遂行上、参照すべき文書の情報を記載する
2.2	定義	プロジェクトの関係者間で誤解や認識が異なることのないよう、必要な用語や概念に定義を明記しておく

Document Format Image

2.1 参照

2.2 定義



2.1 参照

計画作成時やプロジェクトの遂行上で参照する情報を記述し、その出所を明確にする。

● 記述すべき内容

- 計画作成／更新の手順書（フォーマット等）を記載する。
- 計画作成のための参照する上位ドキュメント（製品企画書、要件仕様書等）、関連他部門ドキュメント（ハードウェア／メカの計画書等）を記載する。

● 記述の際に注意／考慮すべき内容

- 参照する文書名、バージョンを明確にする。
- 参照する文書の所在を明確にする。
- 参照文書の内容ではなく名称や参照箇所・ページなどを明記する。

参照ドキュメント表

参照ドキュメントを識別できる番号をつける

文書の作成日なども記載しておく

ドキュメントID	文書名	作成日	保管部門

参照する文書の正式名称を記載する
文書のバージョンなども記載する

文書の出所や保管部門の情報も記載する

2.2 定義

プロジェクトの関係者間で誤解や認識が異なることのないよう必要な用語等を定義する。特にプロジェクト内、関連他部門（ハードウェア／メカプロジェクト等）で認識の違い、誤解を招かないようにする。

● 記述すべき内容

- 計画書を読む人の一部にしか理解できない用語を全員に理解できるように解説を記載する。
- 以下のような用語に対する解説を記載する。
  －略語に対してのフルスペル
  －英語に対しての和訳
  －特殊用語に対しての解説
- プロジェクト内で独自に定義した用語の解説を記載する。
- また、用語についてはその意味などを含めて参照できる情報があれば明示しておく。

● 記述の際に注意／考慮すべき内容

- 複数の意味にとれる用語、人により異なって理解される可能性のある用語を使用しての解説は避ける。

用語など定義表

用語などを識別できる番号をつけておく

略称などがある場合正式名称などを記載する

用語定義などについて参考にした情報があれば出所も記載しておく

ID	用語・概念	正式名称	意味	関連情報ソース

定義する用語や概念名を記載

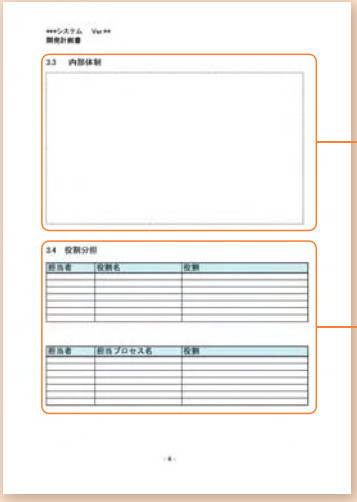
用語などの定義を簡潔に記載する

3　体制

通常、ソフトウェア開発は複数の組織や部門、担当者が関係し、プロジェクトチームを組織して開発に当たるのが一般的になりつつあります。組込みソフトウェア開発の場合には、特に、ソフトウェア開発のみだけでなく、関連するハードウェアの開発なども密接に関係するため、開発プロジェクトの全体像を整理・把握し、関係者の役割などを明確にしておく必要があります。

Chapter 3　体制		
Section	タイトル	概要
3.1	製品開発プロジェクトの体制	製品やシステム全体の開発プロジェクトの体制を整理する。
3.2	外部インタフェース	プロジェクトとプロジェクト外部との関係を明確にしておく。
3.3	ソフトウェア開発プロジェクトの内部体制	本計画書が対象とする範囲のソフトウェア開発プロジェクト遂行に関わる内部の体制を記述し、役割や責任関係を明確にする。
3.4	役割分担	各担当者の役割分担を明確にする。また、支援プロセスについても役割分担を明確にする。

Document Format Image

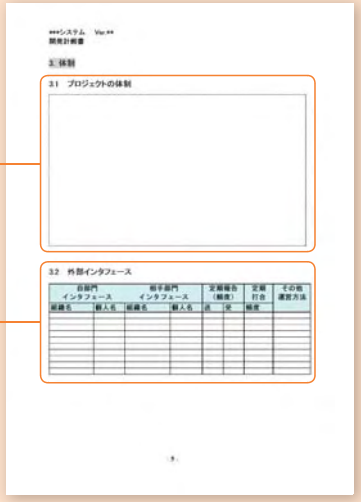


3.1 製品開発プロジェクトの体制

3.2 外部インタフェース

3.3 ソフトウェア開発プロジェクトの内部体制

3.4 役割分担



3.1 製品開発プロジェクトの体制

本計画書が対象とするソフトウェアが搭載される製品やシステム全体の開発プロジェクトの体制、責任分担(ミッション)を、図表を活用して、プロジェクトの内部および外部からわかりやすい形に記述する。

● 記述すべき内容

この計画書で対象としているソフトウェアが搭載される製品やシステム全体の開発プロジェクトの体制や責任分担(ミッション)を、図表を活用して整理する。

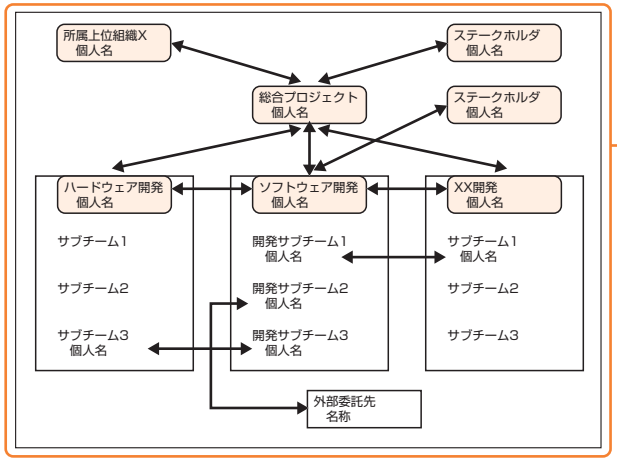
具体的には、

- 製品開発を担う開発プロジェクトの全体構成(ソフトウェア開発プロジェクト、ハードウェア開発プロジェクトなど)
- それぞれのサブプロジェクト同士の関係やこれらと外部組織やステークホルダとの関係
- 開発プロジェクトを構成する各サブプロジェクトを構成するチームおよびその責任者名

などを記述する。

● 記述の際に注意／考慮すべき内容

- プロジェクトの開始時のみに着目するのではなく、(1.8 計画の更新)の項目で記述されるプロジェクト開始後の変更管理への対応なども含めた指示命令系統や運営体制も考慮する。
- マトリクス組織においてのミッションとプロジェクトのミッションが整合しない場合などは、マトリクス組織とプロジェクト両者の関係を考慮する。



・ソフトウェアを構成する要素(ユニットレベル)でグループ構成や編成を考慮する
・関連するハードェア開発グループや外部委託先なども考慮しておく
・個々のグループについては責任者も明示する

3.2 外部インタフェース

プロジェクトとプロジェクト外部の要素別接点（インタフェース）を明確に記述する。

● 記述すべき内容

ソフトウェア開発プロジェクトとプロジェクト外部の要素別接点（インタフェース）を明確に記述する。

基本的な外部インタフェースとしては、以下のようなものが考えられる。

- 上位組織あるいは発注元との間（ステークホルダとの間）
- 関連する他プロジェクトとの間（たとえばハードウェア開発プロジェクトなどとの接点）
- 外部業務委託先との間

外部インタフェースごとに、双方の個人名を特定して、調整方法、連絡方法などを明記する。

週報などによるインタフェースのとり方なども検討する。

● 記述の際に注意／考慮すべき内容

- 上位組織あるいはステークホルダとのインタフェースについては、報告に対する承認の要否、あるいは承認の必要な内容などを明記する。
- 必用に応じ組織図などを活用して、プロジェクトの内部および外部からわかりやすい形で表示する。
- 記載方法については、各々の組織で通常使用されている組織図などの中に外部インタフェースを特定して明記し、具体的運用などで図示しきれない内容については下記のような補足説明のための表などもを添付する。
- ここで規定される外部インタフェースは、「3.1 製品開発プロジェクトの体制」において記載された個々のサブプロジェクトやチーム、その他のステークホルダなどとの相互の接点について整理していく。

外部インタフェース定義表

自部門および関連部門の双方でどの部門の誰が関係するかを明確にしておく

関係者間でどのような形で情報伝達や打ち合わせを進めるかを明確にしておく

自部門インタフェース		相手部門インタフェース		定期報告（頻度）		定期打合	その他運営方法
組織名	個人名	組織名	個人名	送	受	頻度	

3.3 ソフトウェア開発プロジェクトの内部体制

本計画書が対象とするソフトウェアの開発プロジェクトに関して、その内部の体制を記述し、役割や責任関係を明確にする。

● 記述すべき内容

- ソフトウェア開発プロジェクトを構成するチームやグループ、あるいは個人レベルまで含めた体制図を作成する。体制図には以下の要素を盛り込む。
  - 対象ソフトウェアを構成するユニット（開発サブ単位）とそれぞれを担当するグループ名や担当者（リーダ）名とその役割
  - 個々のグループや担当者の責任関係
  - ソフトウェア開発プロジェクトの外部（ハードウェア開発プロジェクト、顧客など）との接点の明示

● 記述の際に注意／考慮すべき内容

- ソフトウェア開発プロジェクトに関わるチームやグループの責任者の「個人名（実名）」をなるべく記述するようにし、不明の場合は不明がわかるように記述する（なるべく早期に明確にする）。また、個人名とともに役割を記述する。
- 下記組織の担当者と調整する役割を持った担当者も明確にする。

  外部組織

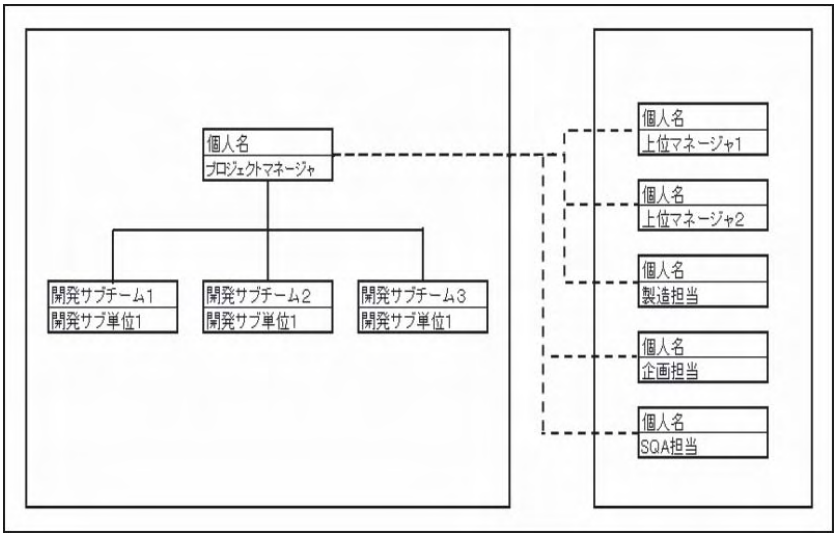
  社外：派遣業者、請負業者

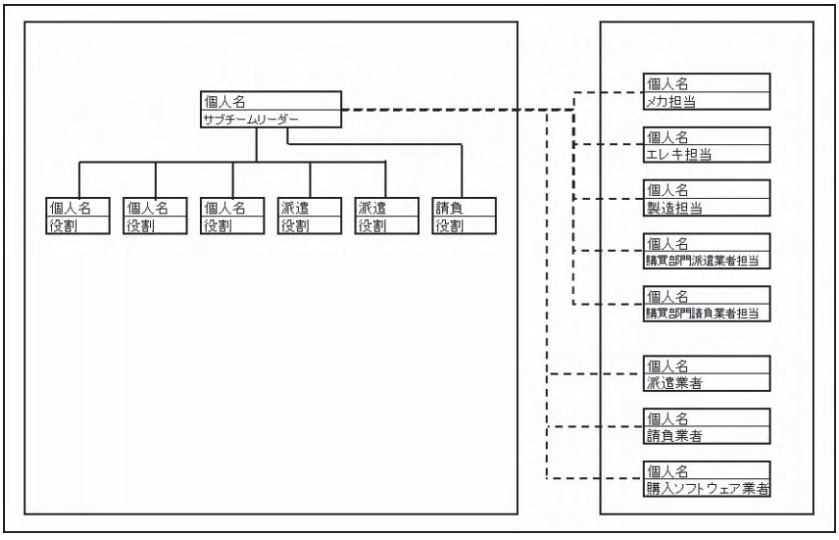
  社内：上位マネージャ、ハードウェア、企画部門、製造部門

  支援プロセス担当組織

  SQA 部門、購買部門

- 担当者と役割表と一貫性を持たせる。
- 組込みシステム開発の場合は、関係する外部組織が多いので、すべての外部組織に対して調整窓口を漏れなく割り当てる。
- 「5.1 作業担当者」での詳細なリソース計画などとの整合性を取るよう注意する。


個人名
プロジェクトマネージャ
開発サブチーム1
開発サブ単位1
開発サブチーム2
開発サブ単位1
開発サブチーム3
開発サブ単位1
個人名
上位マネージャ1
個人名
上位マネージャ2
個人名
製造担当
個人名
企画担当
個人名
SQA担当


個人名
サブチームリーダー
個人名
役割
個人名
役割
個人名
役割
派遣
役割
派遣
役割
請負
役割
個人名
メカ担当
個人名
エレキ担当
個人名
製造担当
個人名
購買部門派遣業者担当
個人名
購買部門請負業者担当
個人名
派遣業者
個人名
請負業者
個人名
購入ソフトウェア業者

3.4 役割分担

ソフトウェア開発プロジェクトにおける主要な担当者について、その役割分担を明確にする。

● 記述すべき内容

- ソフトウェア開発プロジェクトにおける主要な担当者をリストアップし、その役割を整理する。役割分担表には、「担当者名」「担当者の役割名と役割」などを明記する。
- また、支援プロセスの担当者と役割表についても同様にリストアップしておく。
  - －支援プロセスの担当者名
  - －担当者の役割（支援プロセス）名と役割

● 記述の際に注意／考慮すべき内容

- 表でわかりやすく記述する。
- 担当者の「個人名（実名）」を記述する。
- プロジェクト体制図と一貫性を持たせる。
- 役割の表記などについてはETSSのキャリア基準なども参考にすると良い。
- 「5.4 作業担当者」の作業者の割付と矛盾しないようにする。

役割分担表

担当者の個人名を明記する

役割名、役割は組織内で通用する用語を利用するETSSなども参考にして記載する

担当者と役割

担当者	役割名	役割
個人名	プロジェクトマネージャ	開発全体の技術リーダ。 開発プロジェクト全体のプロジェクトマネージメントを行う。 上位マネージメント、企画部門、製造部門、SQA部門との連携を行う。
個人名	開発サブチーム1のリーダ	技術リーダ。 メカ部門とエレキ部門との連携を行う。 サブチーム内の派遣社員管理を行う。
個人名	開発サブチーム2のリーダ	技術リーダ。 買入れするミドルウェアの選定や導入を行う。 サブチーム内の派遣社員管理を行う。
個人名	開発サブチーム3のリーダ	技術リーダ。 請負会社との連携や管理を行う。
個人名	開発サブチームメンバー	開発担当者
個人名	開発サブチームメンバー	開発担当者
個人名	開発サブチームメンバー	開発担当者
個人名	開発サブチームメンバー	開発担当者
個人名	開発サブチームメンバー	開発担当者
個人名	開発サブチームメンバー（派遣）	開発担当者
個人名	開発サブチームメンバー（派遣）	開発担当者

支援プロセスの担当者と役割

担当者	役割名	役割
個人名	構成管理	
個人名	検証と有効性確認	
個人名	文書化	
個人名	品質保証	
個人名	レビューと監査	
個人名	問題解決	
個人名	外注管理	
個人名	プロセス改善	

担当役割については開発プロセスなども考慮する
特に支援プロセスなどについても担当者、役割を明確にしておく

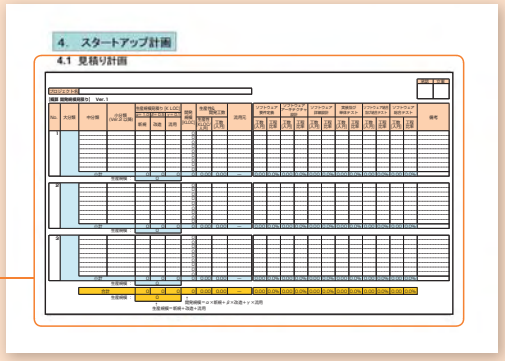
4　リソース計画

プロジェクトを推進する上で、さまざまな費用が必要となります。こうした費用はプロジェクトの企画段階でおおよそが把握されている必要があります。また、プロジェクトの承認後はより精細なコスト計画がないとプロジェクトは運営できません。このため、ここではプロジェクトの開発規模の見積り、必要な人員・機器・設備の計画とそのコストを見積りなどを通して、プロジェクトとしての予算計画を立てていきます。

Chapter 4　リソース計画		
Section	タイトル	概要
4.1	開発規模と工数の計画	開発対象システムの規模、開発の投入工数などを見積もる
4.2	人員計画	開発のためのプロジェクト編成の視点から、必要な開発要員数、スキルレベル、開発者の投入時期などを計画する
4.3	設備・機器等調達計画	開発に必要となる設備や機器などをリストアップし、これらの調達についての計画を立てる
4.4	プロジェクトの人員研修計画	プロジェクト参加予定者にプロジェクト活動で必要となるスキルを取得させるための研修計画を作成する
4.5	予算計画書	上記の4.1～4.4の結果を整理し、プロジェクトで必要なコストなどの予算を整理し、予算計画書を作成する

Document Format Image

4.1 開発規模と工数の計画



4.2 人員計画

4.3 設備・機器等調達計画

4.4 プロジェクトの人員研修計画

4.5 予算計画書

4.1 開発規模と工数の計画

開発プロジェクトで開発するソフトウェアの規模を見積る。また、規模見積りをベースに開発に必要な工数の見積りを行う。

● 記述すべき内容

開発規模と工数の見積りでは、

Step 1：見積り単位を洗い出し、
Step 2：開発規模を見積り、
Step 3：開発の総工数を見積り、
Step 4：総工数を開発工程単位に振り分ける

といった手順で進める。

Step 1：見積り単位の洗い出し

- 初期段階：ソフトウェアを構成するサブシステムレベルに分解し、これを見積りの単位とする。
- 開発前半：確定したシステム仕様を参考に、システムを構成する詳細機能単位を洗い出し、これを見積もり単位とする。

Step 2：開発規模の見積り

- 製品としてROMや機器に開発されるソフトウェアの規模を算出する。規模の算出はたとえば、ソースコードライン数（LOC：Lines of Code）やソフトウェアの機能規模などいくつかの考え方があり、プロジェクトの特性を考慮して採用する。
- コードライン数はStep 1で洗い出した見積り単位ごとに新規分、流用分、再利用分をそれぞれ区分して算出する。

流用	既存資産を一部改造する
再利用	既存資産をすべて使用する

既存資産を利用する流用と再利用については、どのプロジェクト、機能、処理、版数のソースをベースとするのかを明確にし、版数アップやトラブル発生時のリンク情報とする。

- 再見積り時の精度向上のため、生産規模見積り時の算出根拠を記録しておく。

Step 3：開発の総工数の見積り

- 開発で利用する言語や技術者のレベルなどを考慮し、生産性（1000LOCあたりの開発に必要な工数（人月））を設定する。
- 生産性の設定では社内の基準値、過去の類似プロジェクトの実績を参考にし、機能や処理の実現性の難しさなども考慮する。
- 開発に要する総工数は上記の開発規模と生産性より算出する。単位は人月とする。
  開発工数＝開発規模［1000LOC］÷ 生産性［1000LOC/人月］

Step 4：開発総工数の開発工程単位への振り分け

- プロジェクトで実施する工程を考慮し、各工程でどの程度の工数を要するかを考え、開

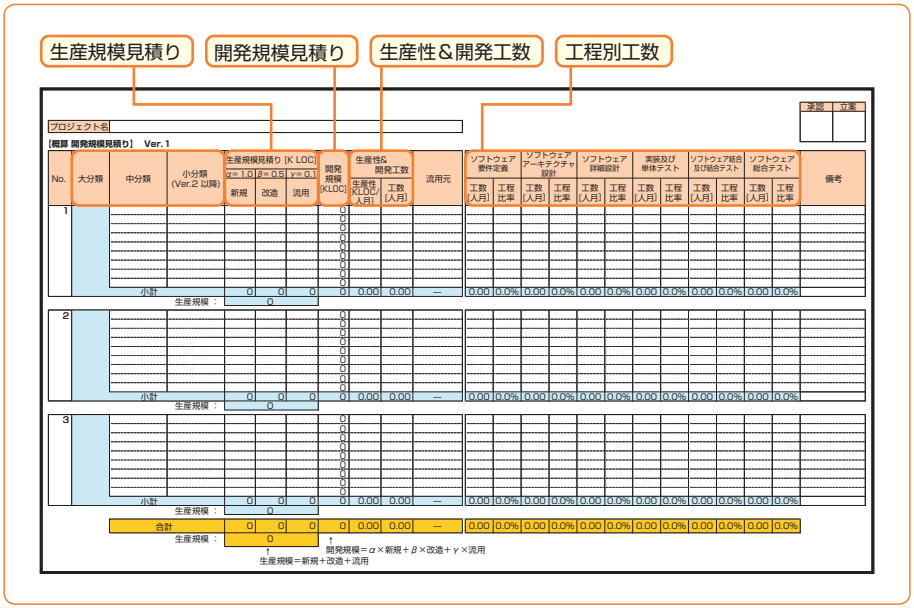
発の総工数の振り分けを行う。

- 工数の振り分けでは、各工程の作業内容などを考慮し、工数振り分けのための工程比率を設定する。工程比率は過去の類似プロジェクトなどの実績を元に設定しても構わない。
- 工程別工数＝開発工数［人月］× 工程比率［%］

● 記述の際に注意／考慮すべき内容

(1) 開発規模見積り

- 類似プロジェクトでの生産規模、開発規模や見積り根拠を確認しておく。
- 類似プロジェクトや社内標準ライブラリの流用や再利用ができないかを検討する。ただし、流用の場合は、改造が入ることにより生産性が低下し、再利用の場合は、改造がないがテストは必要であることを認識する。また、開発ドキュメントの有無も確認しておく。
- 流用する場合は、ベースとなるソースにどのくらいの改造が必要かを見極めること。
- 開発規模を縮小することが可能な、市販ツールの利用を考慮する。
- 類似プロジェクトの見積り担当者とのレビューやアドバイスを受けること。
- 人員計画に基づく工数と開発規模見積りで算出された工数の差異および妥当性を確認し、整合性を保つように見直しする。
- プロジェクトの目的を達成するための困難度、ハードウェア開発遅延、ハードウェアとのすりあわせ結果によるソフトウェアによる実現部分の増加、…等々が想定できる場合は、規模、生産性や係数にてリスクヘッジすること。
- 今後のプロジェクトの見積りをスムースに行うためにも、開発規模の算出根拠を蓄積することが重要であることを認識する。



4.2 人員計画

プロジェクトを推進するための必要人員数、スキルレベル、投入時期、コストを計画し、プロジェクト開始前までに必要な人的リソースを調整する。社内の人員のみで充足できるか、外部委託先からの調達が必要かの否かの判断材料として用いる。

● 記述すべき内容

プロジェクトのアクティビティごとに必要な人員数、職種、スキルレベル、投入時期、必要期間、必要コストを計画する。また、社内の人員のみで実施するか、外部委託先からの調達を行うかを計画する。

(1) 山積み表形式

プロジェクトで実施するアクティビティ（作業）ごとのおおよその作業量を基に、必要な人員に関わるスキル、アクティビティ、担当者、調達形態、コストを一覧形式で記述し、月ごとに必要な人員数を「山積み表」の形式で図式化する。

時間軸には、マスタスケジュールを併記し、各アクティビティの計画人員との関連性を見えるように記述する。

- **開発チーム名：**プロジェクトを構成するチーム名。
- **スキル：**職種（ソフトウェアエンジニアなど）を記述し、プロジェクトにおける役割を決定する。
- **アクティビティ：**担当するアクティビティ（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結合テスト、ソフトウェア総合テスト）を記述し、担当タスクを決定する。
- **担当者：**担当者の氏名。
- **調達先（社内／社外）：**社内、社外人員かを明確にし、外部委託が必要な場合、外部契約形態（請負契約、準委任契約、派遣契約）を検討し、調達責任者との連携情報とする。
- **調達金額：**社内、社外の人員をとわず、調達するためのコストを記述し、計画に対する必要総コストも算出する。

(2) 一覧表形式

必要な人員に関わる下記項目について、「一覧表」形式にて記述する。ただし、作成は必須ではない。

- **開発チーム名：**プロジェクトを構成するチーム名。
- **スキル：**職種（ソフトウェアエンジニアなど）とスキルレベル（レベル1～4）を記述し、プロジェクトにおける役割を決定する。
- **アクティビティ：**担当するアクティビティを記述し、担当タスクを決定する。
- **技術要素：**特に必要な技術要素を明確にする。
- **担当者：**担当者の氏名。
- **期間：**必要な期間（開始／終了）。
- **調達先：**外部委託が必要か否か、外部委託が必要な場合の委託先を検討する。

- **外部委託の場合の契約形態：**契約形態（請負契約、準委任契約、派遣契約）を検討し、調達責任者との連携情報とする。
- **調達責任者：**計画に基づき、人員を調達し、外部委託先と契約する責任者を決定する。
- **調達単価／調達金額：**社内、社外の人員をとわず、調達するための単価、コストを記述し、計画に対する必要総コストも算出する。

● 記述の際に注意／考慮すべき内容

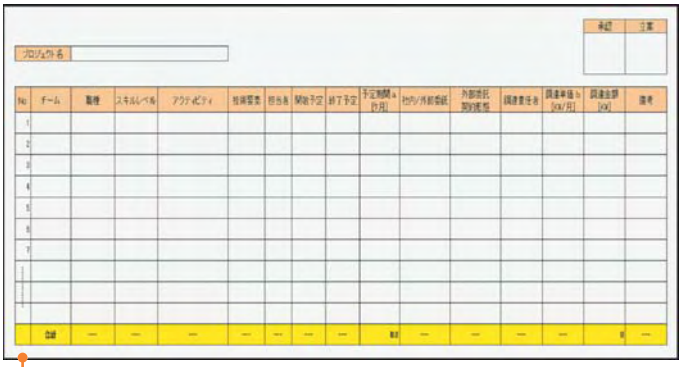
- 類似プロジェクトでの人員計画を確認しておく。
- プロジェクトの目的を達成するためのキーパーソンを決定し、そのキーパーソンに何名のサポーターを割り当てるかを検討する。
- プロジェクト立ち上げまでの計画作成、関係者へのレビュー、見直し、調達や契約までの期間を考慮する。
- 必要なスキルを明確にし、社内や外部委託先への募集要項や判断材料とする。
  －既存領域のスキルで可能か否か
  －新領域のスキルが必要か否か
  －技術以外のスキル要素が必要か否か
- 社内の人員において、そのプロジェクトが占有できる人員であるのか、他のプロジェトと共有する人員であるかを明確にし、関係者と調整する。
  －他部門、関連部門との調整が必要な要員であるか否か
- 設計や開発のみにとらわれず、テストプログラム開発やマニュアル作成等の作業を考慮し、必要な人員を計画する。
- 社外の人員については、契約形態（請負契約、準委任契約、派遣契約）を検討する。
- 社内、社外の人材を問わず調達責任者を決定しておく。
- 社内、社外を問わず、必要なコストを計画する。
- 外部委託先との契約金額が未決定の場合のコストは、職種ごとの工数と標準的な職種別の単価で算出する。
- 計画作成後に変更が発生した場合は、変更要因、変更日、変更者を明記しておく。
- 記述項目で調整が必要な事項や明確にならない事項は、未定と記し、いつまでに決定すべきかを検討する。
- 「プロジェクト体制図」や「担当者と役割」との一貫性を保つように注意する。
- 個人名、調達単価等、個人情報に関わる事項については、組織における規約等にのっとり十分に注意する。

（1） 山積み表形式

開発フェーズの進行に応じた投入人数を時系列的に整理

プロジェクト名						
No	チーム	職種	アクティビティ	担当者	社内/社外委託	費用
1						
2						
3						
4						
5						
6						
7						
					合計	

（2） 一覧表形式



・開発に必要な要員を職種ごと、スキルレベルごとで一覧形式で整理
・投入時期や調達先などを明示する
・必用な要員の手配状況なども分かるようにしておく

4.3 設備機器等調達計画

設備、機器等の必要時期、コストを明確化し、調達活動を遅滞なく実行することで、設備、機器等の調達遅れによるプロジェクトへの影響をなくす。

● 記述すべき内容

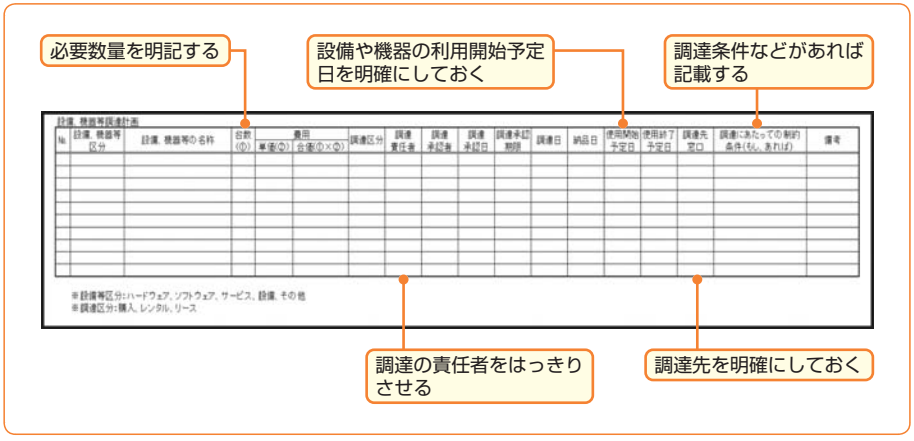
ハードウェア（試作機、本番機等）、ソフトウェア（OS、ミドルウェア等）、サービス（保守サービス等）、設備（ICE、計測器、オシロスコープ等）の調達計画を記述する。

プロジェクト活動に必要であると考えられるリソースは、調達先（社内外）、調達費用に関わらず、すべてを計画する。

● 記述の際に注意／考慮すべき内容

以下の手順を行い、漏れのない計画を行う。

- 調達すべきリソースを適切にカテゴリ分け
- 過去の類似プロジェクトで必要となったリソースを確認
- プロジェクト関係者による計画書レビューの実施
- 調達リスクが高い時はセカンドソースを考慮
- 調達のために秘密保持契約（NDA：Non-Disclosure Agreement）を結ぶ必要がある場合等はその旨を備考欄に記入し、事前調整予備期間を考慮
- 調達品受け入れのための、検収方法を明確にする
- 設備調達などはさまざまな事情を考慮し、余裕をもって発注できるように配慮しておく



4.4 プロジェクトの人員研修計画

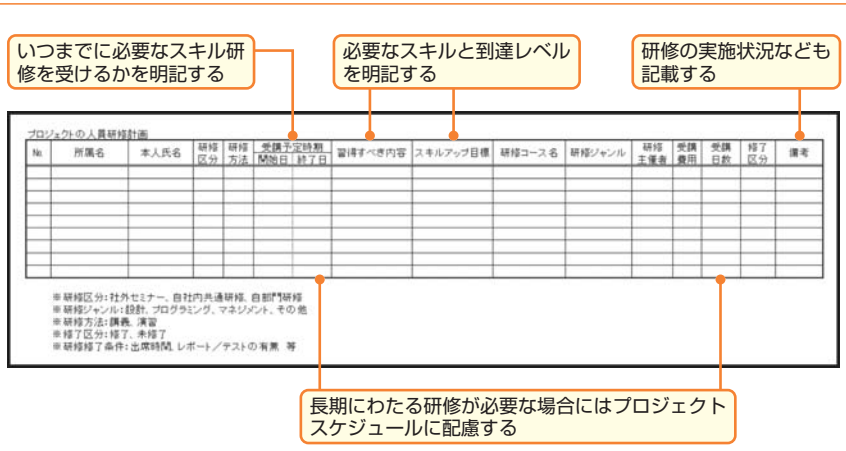
プロジェクト参加予定者がプロジェクト活動で必要となるスキルを必要時点までに取得するため、研修計画を作成する。これにより、必要な時期までに必要スキルをもった人員を確保することができ、プロジェクトをスムースに進捗させることが可能となる。

● 記述すべき内容

- プロジェクトを成功させるために必要となる研修計画を記載する。
- 実ジョブ実施に必要となる技術およびマネジメントに関するスキル習得に関する研修計画とする。
- 人員ごとに、研修区分（社内、社外）、研修方法（講義、演習）、受講予定時期、受講日数、研修コース名等を記載する。

● 記述の際に注意／考慮すべき内容

- プロジェクトに関わる全員（プロジェクトマネージャを含む）についての研修計画を立てる。
- 当該技術者が必要な時期までに受講を修了するようなスケジュールとする。
- できる限り、具体的な研修受講日程を記入する。
- 既存の研修コースにとらわれず、必要な研修は講師を発掘もしくは育成しての実施も検討する。
- ドメイン依存、製品に関する知識などは社内研修で実施する。
- プロジェクトに限った短期的なスキル研修のみではなく、個々の技術者の育成も考慮した長期レンジの教育カリキュラムとの整合性なども考慮する。



4.5 予算計画書

スケジュール割当て計画、人員リソース割当て計画、人員計画、リソース調達計画に基づいて、毎月の予算の割当てを計画する。毎月のプロジェクトの進捗にあわせて、実際のコストを管理する。

● 記述すべき内容

この内容は、この文書を社外に出す場合には削除されるべきものである。

スケジュール割当て計画、人員リソース割当て計画、人員計画、リソース調達計画に基づいて、毎月の予算の割当てを計画する。そして、毎月のプロジェクトの進捗にあわせて、実際のコストを管理する。ここでは、月ごとのコストを計画、予測、実際で管理する。内訳として、人員によるもの、人員以外のリソースによるものが記述される。

● 記述の際に注意／考慮すべき内容

運用において、上位マネジメントに報告する場合には、コスト予測が計画コストとどれくらい差があるのかが重要になる。特に、超過の場合には、その理由と対策が求められる。

5　作業計画

作業計画は対象ソフトウェアを開発するのに、どのような作業をどのような順序で実施し、開発の時間軸上でどのように配置していくかを明確にするものである。また、これらの必用な作業を誰が担当するかについても合わせて検討する。これらの検討結果は作業計画書として整理する。

作業計画書は開発プロジェクト内の作業遂行のベースドキュメントとなるもので、プロジェクト開始後は、プロジェクトの進捗管理のための基準となる。

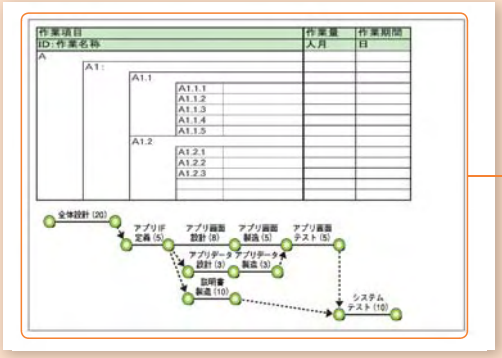
また、同時に作業計画書は開発を担当する部門として外部と部門内の双方に対し責任範囲を明確化する役割も併せ持っている。このため計画策定時には、開発に必要な作業を詳細に記述し、品質、納期、予算がステークホルダから求められる範囲内にあるかの検証も必要となる。

作業計画は対象ソフトウェアの仕様の確定度に応じて、それぞれのタイミングで詳細化していく。

Chapter 5　作業計画		
Section	タイトル	概要
5.1	開発作業の洗い出し	プロジェクトとして実行すべき作業をリストアップし、その入出力、成果物などを明確に定義する。
5.2	開発作業の順序付け	個々の作業に必要な工数、時間数などをもとに開発線表上に個々の作業項目の割付を行う。
5.3	作業担当者の割付	個々の作業に必要な工数や準備できる人的リソースを考慮して、個々の作業の担当者をアサインする。
5.4	作業計画	個々の作業者の作業量や必要な開発資材などを考慮しプロジェクトの予算消化の計画を決定する。

Document Format Image

5.1 開発作業の洗い出し



5.2 開発作業の順序付け

5.3 作業担当者の割付

5.4 作業計画

5.1 開発作業の洗い出し

対象ソフトウェアの開発を進める上でプロジェクトとして実施する作業とその作業の結果作成される成果物を洗い出す。

● 記述すべき内容

(1) ID（各作業を示す一連番号）
(2) 作業名称（内容が明確に理解できること）
(3) 作業標準（作業が従うべき社内ルール等）
(4) 開始条件
- 入力（仕様書、設計書、プログラム等。同一プロジェクト内での中間成果物とプロジェクト外から導入するものがある）および必要となる環境（ツール類、作業場所等）

(5) 終了条件
- 成果物
- 成果物の検証方法

(6) 備考
- 作業内容の説明、特に重要な作業項目、仕様未決等により後日変更の可能性があることを明確化する等で使用。

● 記述の際に注意／考慮すべき内容

作業の洗い出し

- プロジェクトで実施する作業の洗い出しについては、各部門のプロセス標準やESPR（組込みソフトウェア向け開発プロセスガイド）などを参考に洗い出しを進め、プロジェクトとしての開発プロセスを検討する。
- 対象ソフトウェアに関する品質要求、開発コスト面の制約、開発期間の制約や、ソフトウェアが対象とする分野やビジネスの特性などを考慮して必用な作業を洗い出す。
- また、ソフトウェア開発において既存のソフトウェア資産を流用したり再利用したりする場合には、開発プロセスに変更の可能性を生ずるため注意を要する。
- 最も詳細な作業の分割は40人時（1人1週間程度の作業量）を目安とするとよい。

洗い出した作業の確認と整理

- 作成した作業項目は上位マネージャ、類似の開発の担当者等にレビューしてもらう。またソフトウェア以外の設計者とのすりあわせを行い、作業の抜けがないこと、開始条件となる入力が確保できることを確認しておく。
- 作業はESPRに示すように、プロセス、タスク、サブタスク、アクティビティといった階層構造で関連する作業を階層的に整理していくと分かりやすくなる。

作業の開始条件などの確認と成果物、WBS（Work Breakdown Structure：Part 4参照）作成

- 作業の開始条件の入力がプロジェクトと無関係かどうか検討が必要（プロジェクトに依存せず準備されるものかどうかの検討が必要。たとえば治具のチェックプログラム開発等）。

- 成果物は主要な中間生成物、レビュー等の記録も含めることが望ましい。
- ハードウェアやソフトウェアの他の担当者とのレビューは作業の1項目としてWBSに記述する。
- 他のプロジェクトから入手するもの、たとえば仕様書、冶具等のツール類については、その評価も作業として考慮しておく(完全なものを期待しない)。
- ハードウェアとの組み合わせテスト、システムテストの結果、修正や仕様変更への対応を作業として考慮しておく。

作業一覧表（WBS）

5.2 開発作業の順序付け

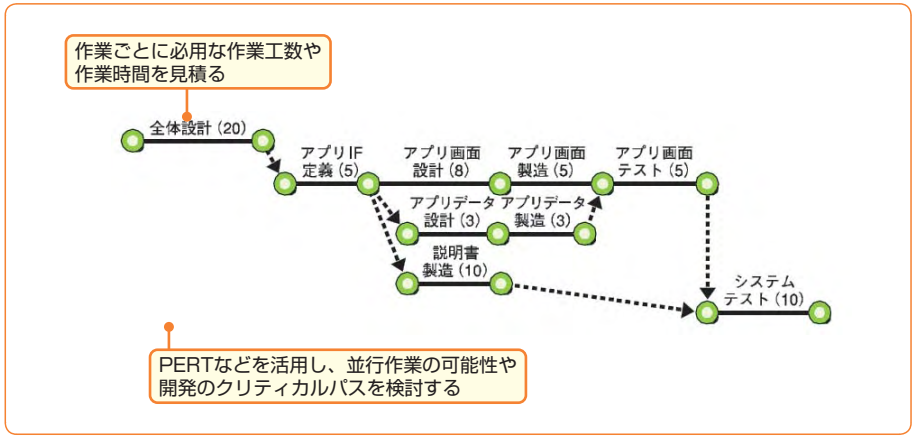
個々の開発作業に必用な作業量（工数）や時間数などを参考に作業の実施順序を決定する。

● 記述すべき内容

- 個々の開発作業に必用な作業量（工数）や時間数を見積もる。
- 各作業では、見積もり結果に基づきPERTなどを利用し、作業の順番を明確にするとともに、直列作業と並行作業を明確にしていく。

● 記述の際に注意／考慮すべき内容

- 開発作業の工数見積りでは担当部分以外のレビューに必要な工数も考慮して計画する。
- ハードウェア部門などとの作業タイミングの調整ポイントなども検討する。
- 長期休暇など、長期に作業できない期間を考慮しておく。
- 作業順序の検討後に机上で作業をシミュレーションし、無理のない作業順序とする。
- リスクの発生確率を軽減するために必要な作業も計画に含めておく。たとえば、外部依存関係（ハードウェアの受け入れ遅れ）など、十分に軽減することのできないリスクは、Contingency（リスクが発生した場合の対応策）を事前に織り込んで、計画上のマージンを持たせておく。
- ハードウェアとの結合は、WBSごとに結合の順番を十分考慮して計画するとともに、仕様を決める際にも必要なすりあわせ期間をレビュー期間として織り込んでおく。
- 作業の独立性と各作業成果物のリンク（関係）を考慮する。



※PERT：Program Evalution and Review Technique（Part 4参照）

5.3 作業担当者の割付

WBSやPERTを使ってスケジュール計画を作成していく過程で、それぞれの作業単位の作業量、工数量、所要時間を見積もり、その作業単位ごとに人員を割り当てる。

● 記述すべき内容

WBSやPERTを使ってスケジュール計画を作成していく過程で、それぞれの作業単位の作業量、工数量、所要時間を見積もり、その作業単位ごとに人員を割り当てる。ここでは、スケジュール割当表の中に、作業担当者として記述する。

● 記述の際に注意／考慮すべき内容 (Part 4参照)

- スケジュール割当て計画(WBSやPERT図)におけるクリティカルパスは、より多くの、または熟練の人員の割当てが必要となる部分である。したがって、人員割当てでは、スケジュール割当てと常に密接に連携しながら、人員計画を参照して、必要なスキル、経験などの観点からその作業に合った人員を割り当てる。
- 前のプロジェクトや他のプロジェクトと今回のプロジェクトとの兼任の度合い(工数%)を明確にし、考慮する。
- 同じような作業を異なる作業者が別々に担当する割当てにならないように注意する。
- プロジェクトの進捗に伴ってクリティカルパスが変わった場合は、それに合わせてリソース割当てを変更する。
- 前項「5.2 開発作業の順序付け」の内容も考慮し、相互で調整しながら決定していく。

作業担当者割付表

個別の作業担当者が把握できるようにする
作業充当率なども考慮する

作業項目		担当者					
ID:作業名称		所属	リーダ	作業期間	所属	メンバー	作業期間
A							
	A1:						
	A1.1						
		A1.1.1					
		A1.1.2					
		A1.1.3					
		A1.1.4					
		A1.1.5					
	A1.2						
		A1.2.1					
		A1.2.2					
		A1.2.3					

5.4 作業計画

個々の開発作業の順序や作業担当者の割付をもとに、プロジェクトとしての作業計画を確定させる。

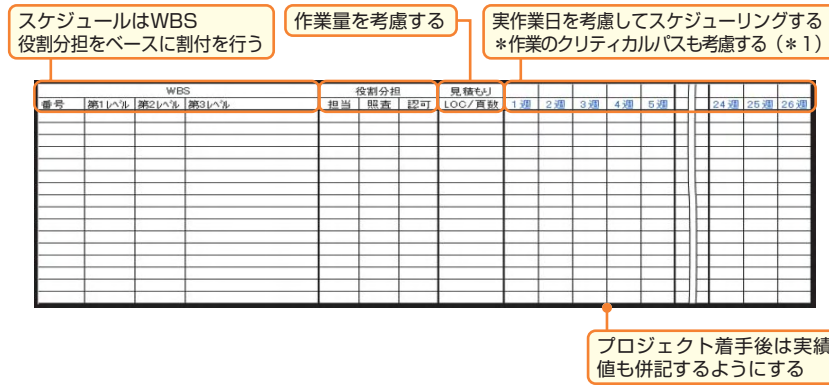
● 記述すべき内容

- **個々の作業**：5.1で洗い出した個々の作業名を記載する。作業は5.2の作業の順序付けの結果をもとに、作業順に従って表記する。
- **作業時期**：開発の実時間上で上記作業の作業開始日/作業終了日を、作業の見積り結果を参考に明記する。
- **作業担当者**：当該作業を誰が担当するかも合わせて表記する。
- 作業場のマイルストーンや主要なイベントなども線表上に織り込んでおく。

● 記述の際に注意／考慮すべき内容

- 作業スケジュールの実日程上の割付では、個々の作業順序や作業担当者の割付などを考慮する。特に、5.2の検討で並行作業として検討した作業についても、実際の作業担当者の割付状況などによっては、作業者の重複などのため、直接作業にならざる得ない場合などが発生する。
- スケジュールの最小単位は、所属する組織で定義されたプロセスで提起されたタスクとし、必要によっては、サブタスクまで詳細化されたスケジュールを作成する。
- 最も細かい粒度の作業スケジューリングは、段階的な見積りの中で、開発前半のシステム仕様が明確になった時点で行うことが現実的である。
- 担当部分以外のレビューに必要な工数も考慮してスケジュールを作成する。また、長期休暇など、長期に作業できない期間も考慮しておく。
- スケジュールの作成後に机上で作業をシミュレーションし、無理のない作業計画とする。

作業計画表



クリティカルパスを明確にすることで、以下の利点が得られる。

・計画策定時に、クリティカルパスに対して積極的に人員を投入することで、スケジュールを最適化できる。
・進捗管理時に、クリティカルパス上の作業に対して課題の対策を行うことで、計画全体の遅れを防止することができる。
・仕様変更依頼があった場合、追加の作業がクリティカルパスに関係するかどうかを見極めることができ、変更がプロジェクト全体期間に影響を与えるかどうか確認することができる。
また、スケジュールに従い、プロジェクトの進行を事前にシミュレーションすることができ、プロジェクトの無理な部分やリスクを洗い出し、進捗管理時にチェックポイントとすることができる。

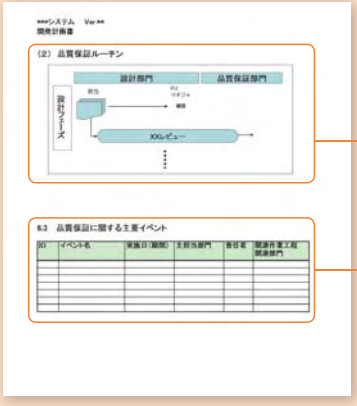
6　品質保証計画

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をあらかじめ決めておく必要があります。

開発計画書の中の品質保証計画の部分では、まず、対象ソフトウェアに求められる品質の目標を記載し、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きます。また、たとえばレビューやテストといった対象ソフトウェアの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なども決めておきます。

Chapter 6　品質保証計画		
Section	タイトル	概要
6.1	品質目標	対象ソフトウェアを利用するユーザやコンテキストを考慮し、製品としてどの程度の品質を目標に開発を進めるかを明確にする。
6.2	品質保証の体制と仕組み	上記の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。
6.3	品質保証に関する主要イベント	レビューやテストなどソフトウェアの品質保証において主要なイベントとその進め方などを明確にしておく。

Document Format Image



6.1 品質目標

6.2 (1) 品質保証体制

6.2 (2) 品質保証ルーチン実施方法

6.3 品質保証に関する主要イベント

6.1 品質目標

対象ソフトウェアを利用するユーザやコンテキストを考慮し、製品としてどの程度の品質を目標に開発を進めるかを明確にする。

● 記述すべき内容

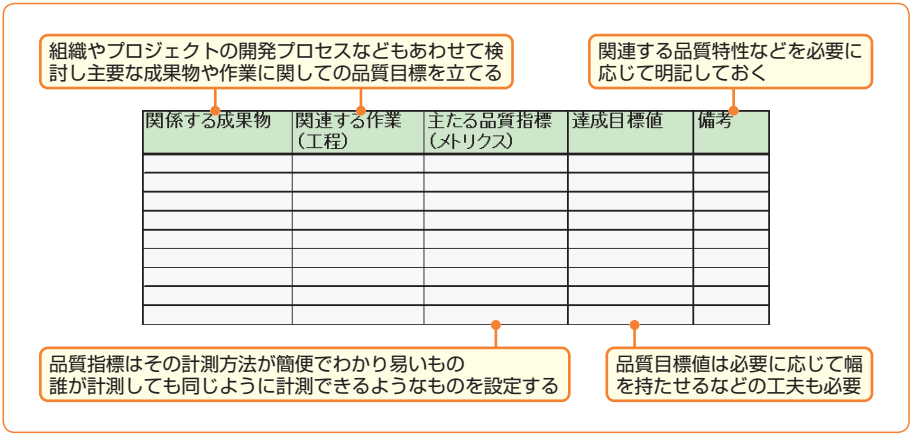
品質目標の基本は、数値などで定量的にソフトウェアの品質を押さえることにある。このためソフトウェア開発の過程で作成される成果物ごとに何らかの評価指標を活用して、その目標値を設定することになる。

たとえば、対象がテスト工程のテスト成績書などであれば、そこに記載されるソフトウェアテスト段階での不具合数や不具合の修正率なども目標値として設定することができる。品質目標の設定では、

- 対象とする成果物や、対象とする作業（工程）を明確にする。
- それぞれをどのような品質指標（メトリクス）で押さえていくかを明確にする。
- 個々の品質指標の目標値を設定していく。

● 記述の際に注意／考慮すべき内容

- ソフトウェアの品質は一般的に不具合数などがクローズアップされるが、それだけでないことに注意が必要。
  一般的に、ISO/IEC9126のソフトウェア品質特性モデルに記載されているように、ソフトウェア製品の品質には「機能性」「信頼性」「効率性」「使用性」「保守性」「移植性」の6つの特性があると考えられている。このため、この品質目標の記載に当たっても、これらの特性を十分に考慮して、品質目標値を設定することが望ましい。たとえば、使用性などについて、システムの起動時間やマニュアルの記載内容なども品質目標の一つとなる。
- 品質目標の設定に関しては、ソフトウェアの対象ユーザや利用コンテキストなどによって、同じソフトウェアであっても品質目標を変えなければならない場合がある。このため、これらも十分に考慮に入れて目標値を設定する。
- 対象ソフトウェアのユーザが特定できる場合には、品質目標値について合意を取っておくことが望ましい。


組織やプロジェクトの開発プロセスなどもあわせて検討し主要な成果物や作業に関しての品質目標を立てる
関連する品質特性などを必要に応じて明記しておく
関係する成果物
関連する作業（工程）
主たる品質指標（メトリクス）
達成目標値
備考
品質指標はその計測方法が簡便でわかり易いもの
誰が計測しても同じように計測できるようなものを設定する
品質目標値は必要に応じて幅を持たせるなどの工夫も必要

6.2 品質保証の体制と仕組み

上記の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、どのような実施方法で品質保証を進めていくかを明確にする。

● 記述すべき内容

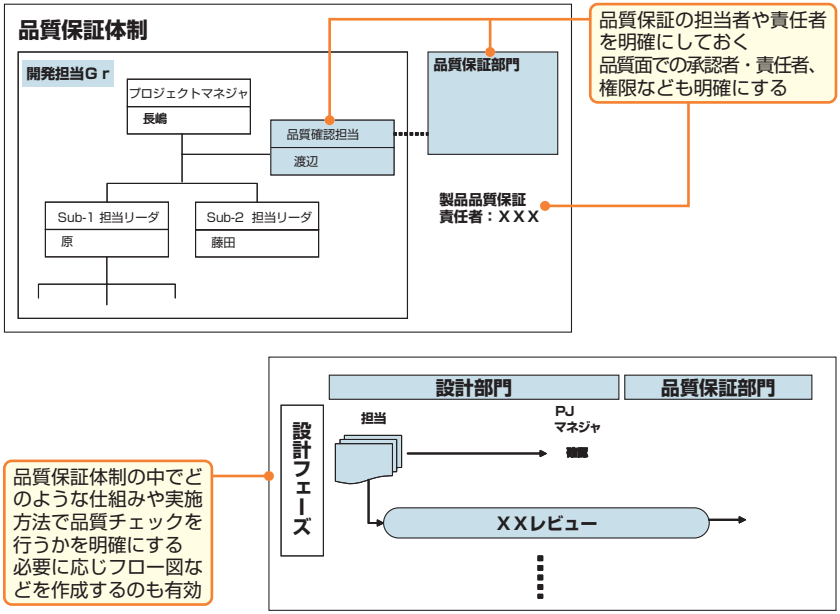
- 開発グループやプロジェクト内で都度、品質面の確認や品質保証活動に当たる担当者を明確にする。
- 企業によっては、製品出荷などの品質保証を担う専門の品質保証部門などが整備されている場合もあるため、こうした品質保証専門の部署と自プロジェクトの品質保証活動に当たる担当者や組織間の間合いを品質保証体制図などによって明確に定めておく。
- 開発の中で、具体的に品質面で確認すべき成果物などがどのような手順で流れ、品質面での確認を行うかなどを明確にわかるように品質保証フローなどを整備しておくとよい。

● 記述の際に注意／考慮すべき内容

- ソフトウェア開発では設計者やプログラマなど一人ひとりの開発技術者は、基本的に自己完結の世界で正しいと考える作業を行っており、自らの過ちなどには気づきにくい面がある。このため品質保証活動においては、

  ①個々の設計者自らが自己チェックなどによって品質面を確認する仕組み
  ②第3者が客観的な目で品質面をチェックする仕組み

  の両方を融合して品質保証を推進する体制や仕組みを構築することが望ましい。


品質保証体制
開発担当Ｇｒ
プロジェクトマネジャ
長嶋
品質確認担当
渡辺
品質保証部門
Sub-1 担当リーダ
原
Sub-2 担当リーダ
藤田
製品品質保証
責任者：ＸＸＸ
品質保証の担当者や責任者を明確にしておく
品質面での承認者・責任者、権限なども明確にする
設計部門
品質保証部門
設計フェーズ
担当
PJ
マネジャ
ＸＸレビュー
品質保証体制の中でどのような仕組みや実施方法で品質チェックを行うかを明確にする
必要に応じフロー図などを作成するのも有効

6.3 品質保証に関する主要なイベント

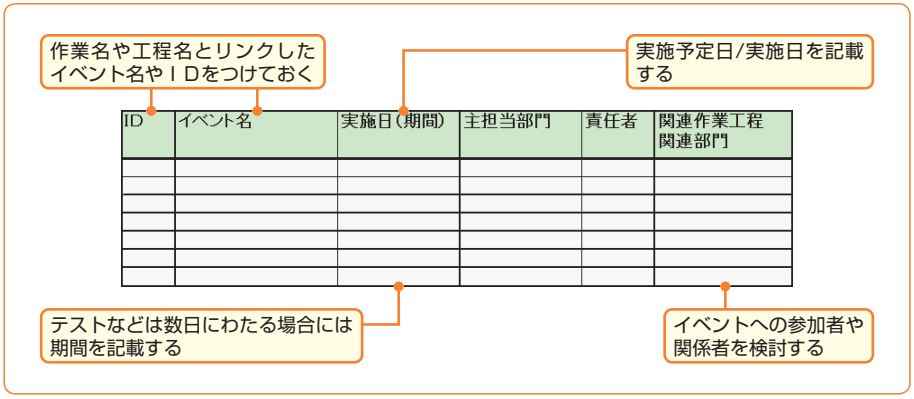
レビューやテストなどソフトウェアの品質保証において主要なイベントとその進め方などを明確にしておく。

● 記述すべき内容

- 開発対象ソフトウェアの品質保証上で欠くことのできない主要なイベントを定めておく。主要なイベントとしては、開発の節目で行われるレビューや開発後半のテストなどを考える。
- これらの主要イベントについては、実施予定日や実施責任部門や担当者なども明確にしておく。

● 記述の際に注意／考慮すべき内容

- 主要イベントの日程設定については、作業計画で立案した作業日程なども考慮し、無理のない日程を設定する。
- レビューなどイベントへの参加者が複数名になる場合には、主要な参加メンバーのスケジュールなども考慮しておく。

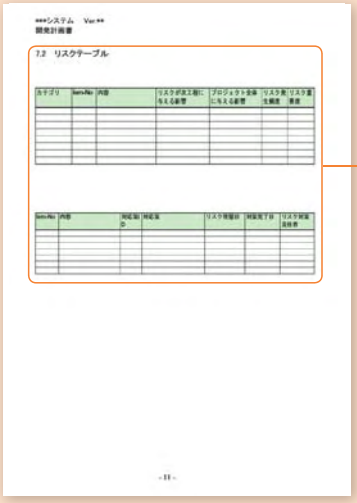


7　リスクマネジメント計画

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します。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ます。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り、実際のプロジェクトで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必要があります。これらの活動は一般的にプロジェクトにおけるリスクマネジメント活動と呼ばれます。ここではプロジェクトのリスクマネジメント活動の計画を立案します。

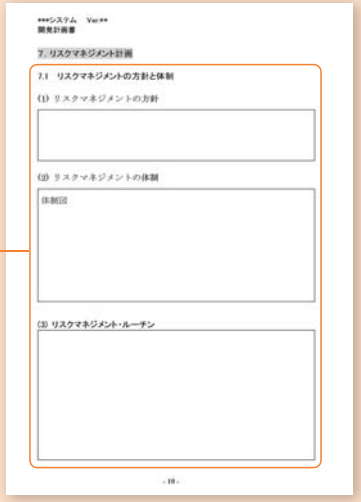
Chapter 7　品質保証計画		
Section	タイトル	概要
7.1	リスクマネジメントの方針と仕組み	プロジェクトのリスクマネジメントに関する基本方針とそれを実行に移すための仕組みを明確にします。
7.2	リスク一覧表	プロジェクトにおいて発生が予見されるさまざまなリスクを洗い出し、リスク一覧表として整理します。

Document Format Image



7.2　リスク一覧表

7.1 リスクマネジメントの方針と仕組み



7.1 リスクマネジメントの方針と仕組み

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。

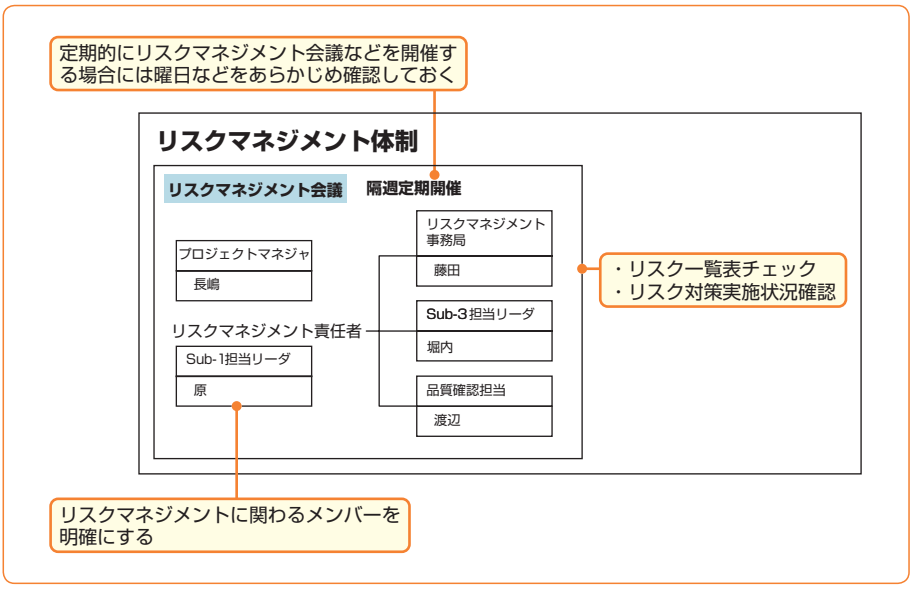
● 記述すべき内容

リスクマネジメントは、リスクのモニタリング、軽減策の検討・実施と効果確認といった活動を進めるのが基本となる。ここでは、どのような方針でこのような活動を行い、実行に移していくか、その仕組みを決めておく。

- **リスクマネジメントの体制：**プロジェクト内で誰がリスクマネジメントを主体的に進めていくかを明確にする。また、個々の技術者やマネージャにも、プロジェクトにおけるリスクマネジメントの考え方を明確に伝え、徹底する。
- **リスクマネジメントの仕組み：**リスクの監視、軽減策の検討・実施・評価といった活動を、どのような実施方法でどのようなタイミングで進めるかを明確にする。

● 記述の際に注意／考慮すべき内容

- リスクの監視や洗い出しについては、プロジェクトの将来を見通したリスクの予見を行う。類似プロジェクトなどにおけるプロジェクト推進上の過去のトラブル事例などを参考にすると良い。
- プロジェクトのリスクは、プロジェクトの進行に伴い、さまざまな新規リスクが発生したり、既に予見できているリスクがさらに別のリスクを誘引したりする場合がある。このため、リスクマネジメント活動は、プロジェクトの進行に合わせて適宜、サイクルをまわしながら継続的に実施するようにその仕組みを構築する必要がある。
- リスクの監視（可視化）、軽減策の検討や実施状況の確認に関しては、リスクマネジメント面でのレビュー会議なども適宜織り込むことが望ましく、同時に、プロジェクト全体の開発節目でのレビューでもリスクマネジメント面を留意する必要がある。


定期的にリスクマネジメント会議などを開催する場合には曜日などをあらかじめ確認しておく
リスクマネジメント体制
リスクマネジメント会議
隔週定期開催
プロジェクトマネジャ
長嶋
リスクマネジメント責任者
Sub-1担当リーダ
原
リスクマネジメント事務局
藤田
Sub-3担当リーダ
堀内
品質確認担当
渡辺
・リスク一覧表チェック
・リスク対策実施状況確認
リスクマネジメントに関わるメンバーを明確にする

7.2 リスク一覧表

洗い出したリスクを整理し、それらに対する軽減策やその実施状況を整理し、リスクの状況を常に見えるようにしておく。

● 記述すべき内容

リスクの洗い出しと評価

- リスク一覧表では、プロジェクトで将来発生する可能性があるリスク（トラブル）を体系的に整理する。
- 洗い出されたリスクについては、そのリスクが発生する確率や可能性などを検討する。
- また実際にリスクが開発のどの時点で発生し、そのリスクによるプロジェクトに対する影響がどの程度かも合わせて評価する。

リスク軽減策の検討と実施フォロー

- 洗い出されたリスクは重要度や発現の可能性を考慮して、リスク軽減のための対策を検討し整理する。
- また軽減策実施の期限や責任者なども明確にしておく。

● 記述の際に注意／考慮すべき内容

- ソフトウェアプロジェクトで発現するリスクには技術面のリスク、リソース面のリスクなどいくつかのカテゴリで分類することができる。Part 4にこれらのリスク分類や典型的なリスクなどを参考例として掲載してあるので参照することが望ましい。
- リスクはプロジェクトの進行とともに変化するため、随時、リスク一覧表の見直しをリスクマネジメント会議などで行う。

プロジェクト名：PH8-sys

カテゴリ	Item-No.	内容	リスクが次工程に与える影響	リスクがプロジェクト全体に与える影響	リスク発生頻度	リスク重要度	リスクポイント
技術	1	FWを初めて製品適用した	従来のテスト項目が利用できない可能性有り	テスト長期化により最終リリースが遅れる	2	2	4
	2	新規機能の検討が遅れている	実装が長引きテスト着手が遅れる可能性有り	同上	4	4	16
	3	新規機能に変更が多い	テスト着手後にも変更入る可能性あり	同上	4	4	16
リソース	4	新規メンバーによる設計部分が多い	新規メンバー担当部分から不具合多発する可能性有り	テストで不具合をとりきれず最終品質が低下する	4	3	12

リスクカテゴリごとに分類
R1: 製品規模
R2: ユーザ特性
R3: ビジネス特性
R4: プロセス
R5: 技術
R6: 開発環境
R7: リソース

リスクインパクト
発生頻度1～5
重要度1～5

リスクID	リスク内容	対応策ID	対応策	リスク検出日	対策完了日	リスク滞留日数	リスク滞留率	リスク対策担当
PJ-22	プロジェクトへの投資継続が未定	PJ-22	投入コストなどを含めプロジェクトの収支を再度見積もる	12/27	未	∞	∞	PM
PM-1	開発コストの総額が未定	PM-1	開発グループとの予算すり合わせを行う	9/4	12/13	99	0.17	PM
PM-6	長期レンジでのリソース調達が見通せない	PM-6b	今後の製品展開を考慮したリソース計画を策定する	9/20	11/27	68	0.38	PM

活用方法

一般的に開発プロジェクトで実施する個々の作業は開発対象となるシステムで実現する機能などの細部が明確にならないと、見えてこないものもあります。このためプロジェクト計画書に盛り込むべき各項目については、必ずしもすべての項目を一度に決めて開発計画書として整理することは現実的ではありません。

本開発計画書のテンプレートの利用法としては、以下のように、段階的にその内容を詳細化し最終的に実行可能な開発計画書として整理することを推奨します。開発計画書を策定する時期としては、下記の2回のタイミングで計画書の精度を段階的に上げていきます。

第1バージョン：開発プロジェクトのキックオフ時、主要な関係者が確定してプロジェクトの細部を決定した時点

第2バージョン：開発対象システムで実現する機能などがほぼ確定し、プロジェクトとして実施する主要な作業が把握できる時点

	プロジェクト・キックオフ時	システム仕様確定時点
Chapter 1　プロジェクトの概要		
1.1　プロジェクトの目的	●	
1.2　プロジェクトの目標	●	
1.3　目標達成のための方針・手段	●	
1.4　プロジェクトの範囲	●	
1.5　プロジェクトの前提条件	●	
1.6　プロジェクトの成果物	●	
1.7　スケジュールと予算	●	
1.8　計画の更新		●
Chapter 2　参照・定義		
2.1　参照		●
2.2　定義		●
Chapter 3　体制		
3.1　製品開発プロジェクトの体制		●
3.2　外部インタフェース		●
3.3　ソフトウェア開発プロジェクト内部体制		●
3.4　役割分担		●

	プロジェクト・キックオフ時	システム仕様確定時点
Chapter 4　リソース計画		
4.1　開発規模と工数の計画		●
4.2　人員計画		●
4.3　設備、機器等の調達計画		●
4.4　プロジェクトの人員研修計画		●
4.5　予算計画書		●
Chapter 5　作業計画		
5.1　開発作業の洗い出し		●
5.2　開発作業の順序付け		●
5.3　開発作業担当者の割付		●
5.4　作業計画の確定		●
Chapter 6　品質保証計画		
6.1　品質目標		●
6.2　品質保証の体制と仕組み		●
6.3　品質保証に関する主要なイベント		●
Chapter 7　リスクマネジメント		
7.1　リスクマネジメントの方針と仕組み		●
7.2　リスク一覧表		●

●：最終案確定

Part 3
事例編

ここでは、本書で紹介したESMR Ver.1.0のプロジェクト計画書テンプレートを活用して作成した「プロジェクト計画書」のサンプルを示します。

3.1 事例プロジェクトの概要

1. 開発製品

・**計測器**（無線通信波形計測、分析、信号発生）

2. 機器構成

・**表示部**（信号波形等表示用ディスプレイ）

・**操作部**

ジョグダイヤル、機能設定ボタン、数値入力ボタン（テンキー相当）

・**信号受信部**

・**信号発生部**

・**入出力・記憶**

LAN、シリアル、USB、カード・コンパクトフラッシュ、30GB・HDD、信号記憶用メモリ

3. 開発目標

・次世代高速通信向けの計測器を早期に市場投入し、シェアを確保する。

・従来製品の最高周波数200MHzを300MHzに向上させる。

・既存資産を活用し、開発期間短縮・生産性向上を図る。

4. ソフトウェア開発項目

4-1 操作関連（アプリケーション）

①受信機能設定

②信号発生機能設定

③データ表示機能設定

④データ編集機能設定

⑤入出力設定

4-2 ファームウェア

①表示関連

②ASICまわり

③データ編集

④入出力関連

⑤自己診断・製造検査プログラム

4-3 ミドルウェア

①HDD

②通信（LAN、USB、シリアル）

5. ソフトウェア以外の開発品目

- 機構設計
- 回路設計
- ASIC 設計

6. 開発条件・制約等

- ケースその他機構部は既存製品のものをできるだけ流用する。
- ASIC はソフトウェアに先行して３ヶ月前より開発中。
- その他　ハードウェア技術は既存あるいは他製品開発よりできるだけ流用するが場合によって新規開発が必要となる。

7. 開発人員

20名（プロジェクトマネージャ１名含む）
（外部委託　派遣４名、請負６名　含む）

8. 開発期間

９ヶ月

9. 製品に搭載されるソフトウェアの規模

約８万 step
（既存資産の３割を追加・改造）

10. 製品開発の課題と優先順位

①製品化時期
②コスト
③品質

11. 製品に搭載されるソフトウェアのシステム構成

- アプリケーション
- ミドルウェア　一部購入コンポーネント
- デバイスドライバ
- OS

12. ソフトウェア開発プロジェクトのステークホルダ

- 企画
- 購入先
- ハードウェア

3.2　事例プロジェクトの計画書

仮想プロジェクト計画書

版数	作成日	作成者	査閲者	承認者
初版	06-10-25	高橋	原	わたなべ

ソフトウェア開発１部

1.1　プロジェクトの目的

・本プロジェクトは既存製品XXの後継機種のソフトウェア開発全般を対象とする。
・開発対象機種は既存製品XXの高周波数対応を主眼としている。
・シェアを確保するために次世代高速通信向けの計測器を早期に市場投入する。
・性能を達成することを第一優先とし、そのためにハードウェア開発に対し、最大限の支援を行う。

1.2　プロジェクトの目標

・本プロジェクトに付与される開発費用は総額１億５千万円である。
・工場へのソフトウェア製造情報の送達期限は06年９月28日とする。
・最高対応周波数は300MHz以上を達成していなければならない。
・製品化時残存バグ数を既存製品XX対1/10を目標とする。

1.3　目標達成のための方針・手段

・周波数300MHz動作を確認するために評価プログラムを最優先に開発する。
・周波数300MHz動作を確認する作業は、ハードウェア開発プロジェクトメンバーとサブプロジェクトを構成して行う。
・既存資産を活用し、開発期間短縮･生産性向上を図る。

1.4　プロジェクトの範囲

・本プロジェクトの範囲は、製品に搭載される全プログラムを対象とする（機構設計、回路設計、ASIC設計は対象としない）。
・ミドルウェアは購入して組込む。
・自己診断および検査プログラムを開発して正式リリースする。
・本プロジェクトは、1stロット出荷確認試験に合格したことを持って終了とする。
・外部接続試験は本プロジェクトの範囲外とし、別プロジェクトでの実施とする。
・製品マニュアル原稿を作成する。

1.5　プロジェクトの前提条件

前提条件ID	内容	条件重要性
1-1	ASIC開発：　2006年3月31日納品予定	A
1-2	ハードウェア試作機：　2006年4月20日受領予定（n台）	A
1-3	ハードウェア実機：　2006年8月10日受領予定（n台）	A
1-4	ハードウェアテストプログラム：　2006年5月25日リリース予定	B
1-5	総合テスト用プログラム：　2006年8月10日リリース予定	B
2-1	既存資産（型式XXXXX）を流用 　現時点での見積は、既存資産の60%を再利用することを前提とする （アーキテクチャ設計により再利用度を確定）	C
3-1	開発環境 　既存開発環境の利用を前提とする	B
3-2	ミドルウェア：　2006年3月31日迄に入手	A

1.6　予定成果物表

ID	成果物名称	作成時期	作成責任者
D-001	ソフトウェア要求仕様書	1月25日	山田
D-002	ソフトウェア・アーキテクチャ設計書	2月20日	佐藤
D-003	ソフトウェア詳細設計書	3月28日	鈴木
D-004	テスト仕様書	3月31日	佐藤
D-004-1	テスト報告書	9月20日	山田
D-005	共同レビュー記録	9月23日	山田
D-006	自己診断プログラム	9月15日	鈴木
D-007	マニュアル	9月15日	佐藤

1.7　スケジュール概要表

月	1	2	3	4	5	6	7	8	9
アプリケーション設計	→	→	→						
ファームウェア設計	→	→	→						
アプリケーション実装				→					
ファームウェア実装				→					
ハードウェアテスト					→				
ソフトウェアテスト						→	→	→	
結合・総合テスト								→	→

月日	マイルストン名称	責任者
2月22日	アーキテクチャ設計レビュー	山田
3月30日	詳細設計内部レビュー	佐藤
6月30日	第1回設計審査	山田
9月10日	第2回設計審査	山田
9月28日	工場への情報送達	佐藤

1.7　概算予算表

予算項目ID	予算項目	予定金額
Y-1	社内人件費	￥×××百万
Y-2	外部人件費	￥×××百万
Y-3	購入品	￥×××百万

1.8　計画の更新

見直し計画	進捗ミーティングにおいて各マイルストン期日を守れないと判断された場合、下記更新手順にのっとり計画を見直す。
見直し手順	1. 進捗ミーティング主催者はプロジェクトマネージャに速やかに状況を報告する。 2. プロジェクトマネージャは速やかに関係者を招集しレビューを実施する。 3. レビューの結果、計画を見直さざるを得ない時は、プロジェクトマネージャは計画書を更新し部門長に報告する。 4. 部門長承認をもって正式に計画書は更新される。 5. プロジェクトマネージャは更新された計画書を関係者すべてに配布し、周知する。
最新版の所在	計画書の最新版は必要項目が捺印された文書を計画書ファイルに綴じる。
更新ルール	版数付与基準、更新箇所表記等は社内基準に準拠する。

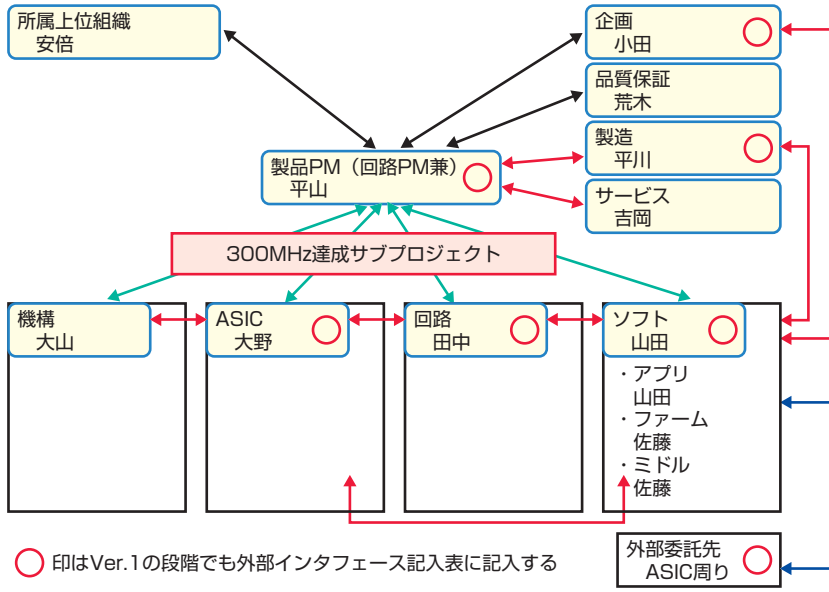
2.1　参照ドキュメント表

ドキュメントID	文書名	作成日	保管部門
RD-001	製品要求仕様書	11月15日	企画部
RD-002	システム設計書	12月20日	製品統括部
RD-003	ハードウェア設計書	12月25日	ハードウェア設計部

2.2　用語等定義表

用語・定義ID	用語・概念	正式名称	意味	関連情報ソース
T-001	GUI	Graphical User Interface	本製品の場合操作画面を指す	
T-002	スペクトラム	スペクトラム拡散（spread spectrum）	拡散符号によって元のデジタル信号を復元すること	IEEE802

3.1　製品開発プロジェクトの体制

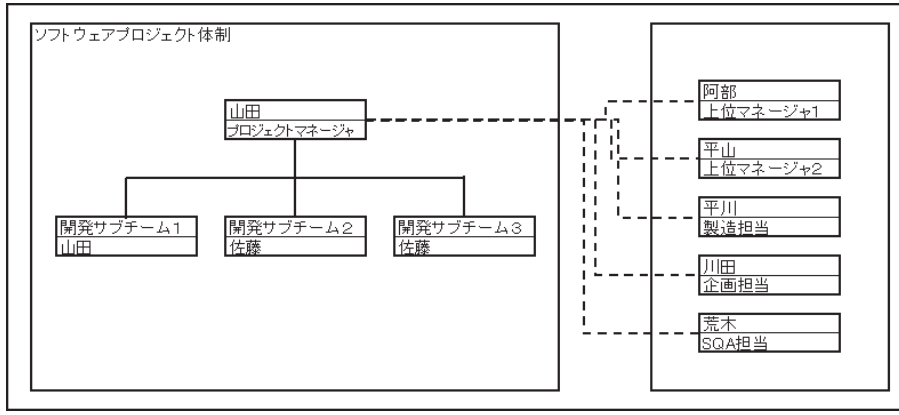


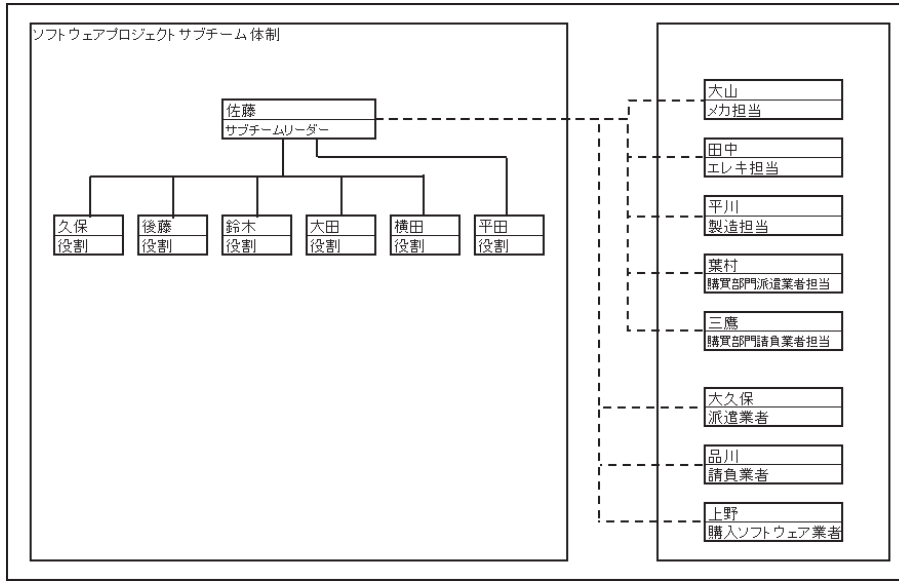
3.2　外部インタフェース

外部インタフェース

自部門インタフェース		相手部門インタフェース		定期報告（頻度）		定例打合（頻度）	その他運営方法
組織名	個人名	組織名	個人名	送	受		
ソ開1	山田	回路	田中	週1	週1	週1	
	佐藤	ASIC	大野	週1	週1	週1	
	佐藤	外部委託先	渡辺	週1	週1	月1	
	山田	企画	川田	週1	月1	月1	
	佐藤	製造	平川	月1	月1	月1	製造移管時別途

3.3　ソフトウェア開発プロジェクト内部体制





3.4 役割分担

担当者と役割		
担当者	**役割名**	**役割**
山田	プロジェクトマネージャ	開発全体の技術リーダ。 開発プロジェクト全体のプロジェクトマネージメントを行う。 上位マネージメント、企画部門、製造部門、SQA 部門との連携を行う。
佐藤	開発サブチーム 1 のリーダ	技術リーダ。 メカ部門とエレキ部門との連携を行う。 サブチーム内の派遣社員管理を行う。
加藤	開発サブチーム 2 のリーダ	技術リーダ。 買入れするミドルウェアの選定や導入を行う。 サブチーム内の派遣社員管理を行う。
徳川	開発サブチーム 3 のリーダ	技術リーダ。 請負会社との連携や管理を行う。
久保	開発サブチームメンバー	開発担当者
後藤	開発サブチームメンバー	開発担当者
鈴木	開発サブチームメンバー	開発担当者
真田	開発サブチームメンバー	開発担当者
大田	開発サブチームメンバー（派遣）	開発担当者
横田	開発サブチームメンバー（派遣）	開発担当者
支援プロセスの担当者と役割		
担当者	**役割名**	**役割**
後藤	構成管理	部署サーバとファイル管理
山田	検証と有効性確認	検証についての計画、内容、および有効性の確認
鈴木	文書化	定められた文書の作成、管理
加藤	品質保証	開発ソフトウェアの品質保証
山田	レビューと監査	内部レビューの実施確認、監査
山田	問題解決	発生した問題の解決計画、管理
佐藤	外部委託	外部委託業者との連絡、管理

4.1 開発規模と工数の計画

プロジェクト名	

【概算 開発規模見積り】 Ver.1

No.	大分類	中分類	小分類 (Ver.2 以降)	生産規模見積り [K LOC] α = 1.0 新規	β = 0.5 流用	γ = 0.1 再利用	開発規模 [KLOC]	生産性 & 開発工数 生産性 [KLOC/人月]	工数 [人月]	流用元
1	アプリ	受信機能設定	----	16	4	2	18	1.43	12.74	XYZ ver1.3
		信号発生機能設定	----				0			
		データ表示機能設定	----				0			
		データ編集機能設定	----				0			
		入出力設定	----				0			
							0			
							0			
							0			
		小計		16	4	2	18	1.43	12.74	----
			生産規模：	22						
2	ファーム	表示関連	----				0			
		ASICまわり	----				0			
		データ編集	----				0			
		入出力関連	----				0			
							0			
							0			
							0			
		小計		0	0	0	0	0.00	0.00	----
			生産規模：	0						
3	ミドル	HDD	----				0			
		LAN、USB、シリアル	----				0			
		計測用信号入出力	----				0			
							0			
							0			
							0			
		小計		0	0	0	0	0.00	0.00	----
			生産規模：	0						
		合計		16	4	2	18	1.43	12.74	----
			生産規模：	22			↑			
				↑			開発規模＝α×新規＋β×流用＋γ×再利用			
				生産規模＝新規＋流用＋再利用						

(Part 2 4.1 参照)

承認	立案

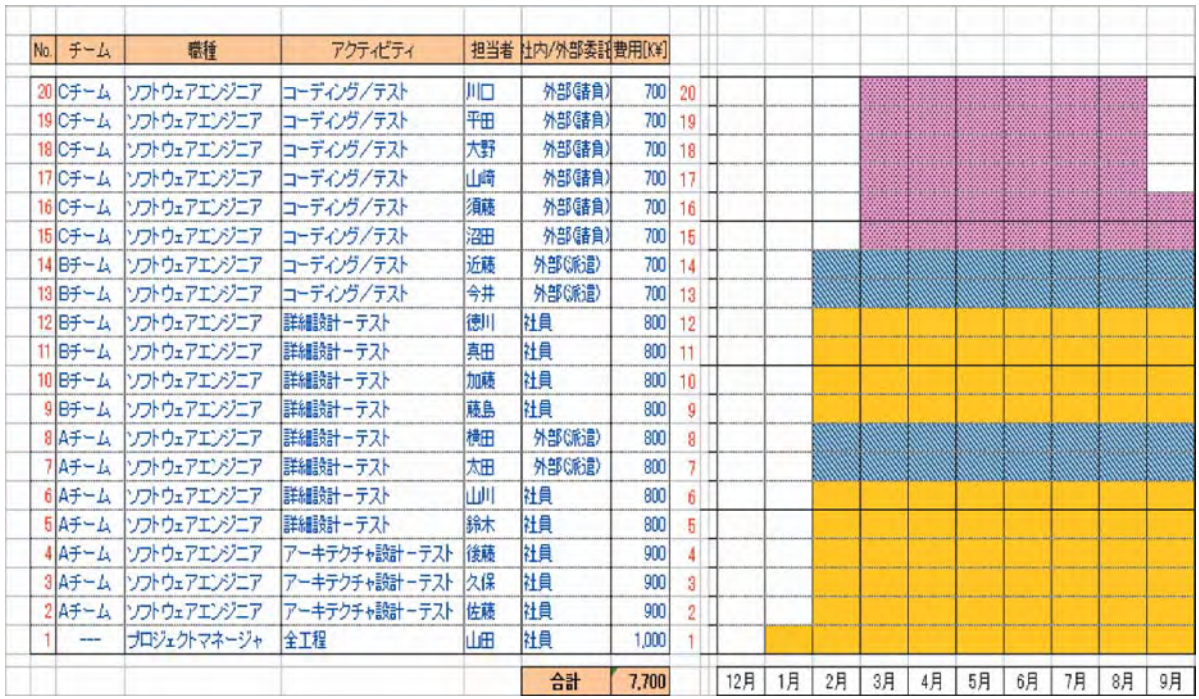
ソフトウェア要件定義		ソフトウェアアーキテクチャ設計		ソフトウェア詳細設計		実装及び単体テスト		ソフトウェア結合及び結合テスト		ソフトウェア総合テスト		備考
工数[人月]	工程比率	工数[人月]	工程比率	工数[人月]	工程比率	工数[人月]	工程比率	工数[人月]	工程比率	工数[人月]	工程比率	
0.64	5.0%	1.27	10.0%	3.19	25.0%	5.10	40.0%	1.91	15.0%	0.64	5.0%	
0.64	5.0%	1.27	10.0%	3.19	25.0%	5.10	40.0%	1.91	15.0%	0.64	5.0%	

0.00	0.0%	0.00	0.0%	0.00	0.0%	0.00	0.0%	0.00	0.0%	0.00	0.0%	

0.00	0.0%	0.00	0.0%	0.00	0.0%	0.00	0.0%	0.00	0.0%	0.00	0.0%	

0.64	5.0%	1.27	10.0%	3.19	25.0%	5.10	40.0%	1.91	15.0%	0.64	5.0%	

4.2 人員計画



No.	チーム	職種	アクティビティ	担当者	社内/外部委託	費用[K¥]
20	Cチーム	ソフトウェアエンジニア	コーディング/テスト	川口	外部(請負)	700
19	Cチーム	ソフトウェアエンジニア	コーディング/テスト	平田	外部(請負)	700
18	Cチーム	ソフトウェアエンジニア	コーディング/テスト	大野	外部(請負)	700
17	Cチーム	ソフトウェアエンジニア	コーディング/テスト	山崎	外部(請負)	700
16	Cチーム	ソフトウェアエンジニア	コーディング/テスト	須藤	外部(請負)	700
15	Cチーム	ソフトウェアエンジニア	コーディング/テスト	沼田	外部(請負)	700
14	Bチーム	ソフトウェアエンジニア	コーディング/テスト	近藤	外部(派遣)	700
13	Bチーム	ソフトウェアエンジニア	コーディング/テスト	今井	外部(派遣)	700
12	Bチーム	ソフトウェアエンジニア	詳細設計ーテスト	徳川	社員	800
11	Bチーム	ソフトウェアエンジニア	詳細設計ーテスト	真田	社員	800
10	Bチーム	ソフトウェアエンジニア	詳細設計ーテスト	加藤	社員	800
9	Bチーム	ソフトウェアエンジニア	詳細設計ーテスト	蔵島	社員	800
8	Aチーム	ソフトウェアエンジニア	詳細設計ーテスト	横田	外部(派遣)	800
7	Aチーム	ソフトウェアエンジニア	詳細設計ーテスト	太田	外部(派遣)	800
6	Aチーム	ソフトウェアエンジニア	詳細設計ーテスト	山川	社員	800
5	Aチーム	ソフトウェアエンジニア	詳細設計ーテスト	鈴木	社員	800
4	Aチーム	ソフトウェアエンジニア	アーキテクチャ設計ーテスト	後藤	社員	900
3	Aチーム	ソフトウェアエンジニア	アーキテクチャ設計ーテスト	久保	社員	900
2	Aチーム	ソフトウェアエンジニア	アーキテクチャ設計ーテスト	佐藤	社員	900
1	---	プロジェクトマネージャ	全工程	山田	社員	1,000
					合計	7,700

(Part 2 4.2山積み表参照)

承認	立案

プロジェクト名

No.	チーム	職種	スキルレベル	アクティビティ	技術要素	担当者	開始予定	終了予定	予定期間 a [ヶ月]	社内/外部委託	外部委託契約形態	調達責任者	調達単価 b [K¥/月]	調達金額 [K¥]	備考
1	A	ソフトウェアエンジニア	3	アーキテクチャ設計 アーキテクチャテスト		佐藤	2/1	9/28	8	内			900	7200	
2	A	ソフトウェアエンジニア	2	アーキテクチャ設計 アーキテクチャテスト		久保	2/1	9/28	8	内			900	7200	
3	A	ソフトウェアエンジニア	2	詳細設計テスト		後藤	2/1	9/28	8	内			900	7200	
4	A	ソフトウェアエンジニア	2	詳細設計テスト		鈴木	2/1	9/28	8	内			800	6400	
5	A	ソフトウェアエンジニア	2	詳細設計テスト		山川	2/1	9/28	8	内			800	6400	
6	A	ソフトウェアエンジニア	1	詳細設計テスト		太田	2/1	9/28	8	外	派遣	佐藤	700	3600	
7	A	ソフトウェアエンジニア	1	詳細設計テスト		福田	2/1	9/28	8	外	派遣	佐藤	700	3600	
	合計	---	---	---	---	---	---	---	0.0	---	---	---	---	416000	---

（Part 2 4.2一覧表参照）

4.3 設備機器等調達計画

NO.	設備、機器等区分	設備、機器等の名称	数量(①)	費用		調達区分	使用開始予定日	使用終了予定日	調達先窓口	調達にあたっての制約条件(もし、あれば)	備考
				単価(②)	合価(①×②)						
	H	試作機	5		-	社内借用	5/1	8/14	ハード/ 名		デバッグ開始時最低 3 台
	H	本番機	5		-	社内借用	8/15	9/28	ハード/ 名		
	S	OS	5	5	25	購入	4/20	9/28	代理店名	version 指定	
	S	ミドルウェア	5	10	50	購入	4/20	9/28	代理店名	version 指定	
	S	コンパイラ	20		-	既存設備流用	4/5	9/28	設備管理者/ 名	version 指定	
	H	ICE	5		-	既存設備流用	5/1	9/28	設備管理者/ 名		

(Part 2 4.3参照)

4.4 プロジェクトの人員研修計画

NO	所属名	本人氏名	研修区分	研修方法	受講予定時期		習得すべき内容	スキルアップ目標	研修コース名	研修ジャンル	研修主催者	受講費用	受講日数	研修区分	備考
					開始日	終了日									
1			自部門	講義/演習			ファーム開発基礎	ファーム開発初級	ファーム開発基礎	設計/プログラ	部門教育担当者/名	0	3		xx ファーム開発にアプリ担
2	A チーム	佐藤					外部委託管理	外部委託管理者	外部委託管理	社内共通	全社購買担当者/名	0	3		xx 開発で外部発注するため

4.5 予算計画

月	予算リソース（百万円）			人員リソース			社員コスト（百万円）			外部委託（人数）			外部委託コスト（百万円）			人員以外のリソース（百万円）		
	計画コスト	予測コスト	実際コスト	社員（人数）			計画コスト	予測コスト	実際コスト	計画	予測	実際	計画コスト	予測コスト	実際コスト	計画コスト	予測コスト	実際コスト
				計画	予測	実際												
1	1			1			1											
2	14.5			10			11.5			4			3					
3	18.7			10			11.5			10			7.2					
4	19			10			11.5			10			7.2			0.25		
5	19.3			10			11.5			10			7.2			0.5		
6	18.7			10			11.5			10			7.2					
7	18.7			10			11.5			10			7.2					
8	18.7			10			11.5			10			7.2					
9	15.9			10			11.5			6			4.4					

5.1 作業一覧表（WBS）

作業項目 ID：作業名称		使用する作業標準	開始条件 入力	終了条件 成果物	終了条件 検証方法	備考
A：		社内技術選定				
A.1：ファームウェア開発			ソフトウェア要求仕様書			
A.1.1：表示関連						
A.1.1.1	アーキテクチャ設計			アーキテクチャ設計書	レビュー	
A.1.1.2	詳細設計			詳細設計書	レビュー	
A.1.1.3	実装				テスト	
A.1.1.4	テスト仕様			テスト仕様書	レビュー	
A.1.1.5	テスト				レビュー	
A.1.2：ASICまわり						
A.1.2.1	詳細設計					
…						
A.1.2.4						
A.1.2.5						
A.1.3：データ収集						
A.1.3.1						
…						
A.1.3.5						

5.3 スケジュール割当て

Part 4

参考情報

プロジェクトを成功に導くためには、まず、プロジェクトのゴールを明確にして、そのゴールを達成できる計画を立案する必要があります。このための基本的な手法がWBS（Work Breakdown Structure）とPERT（Program Evaluation and Review Technique）です。
ここでは、上記の他に開発プロセス、見積り手法、品質特性モデルなど、プロジェクトマネジメントを行う上で、最も重要な領域である計画策定および管理に関連する参考情報をいくつか紹介します。

4.1 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、細かな成果物へと階層的に詳細化し、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ーなど）によって事前に確認することで、プロジェクトのスコープ（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。

<table>
<tr><td rowspan="15">携帯電話
端末開発</td><td rowspan="3">A. マイルストーン</td><td colspan="3">A 01 要求分析完了</td></tr>
<tr><td colspan="3">A 02 全体設計完了</td></tr>
<tr><td colspan="3">A 03 ……</td></tr>
<tr><td colspan="4">B. プロジェクトマネジメント</td></tr>
<tr><td rowspan="3">C. システム</td><td colspan="3">C 01 全体設計</td></tr>
<tr><td colspan="3">C 02 システムインストール</td></tr>
<tr><td colspan="3">C 03 システムテスト</td></tr>
<tr><td rowspan="4">D. アプリプログラム</td><td rowspan="2">D 01 画面</td><td colspan="2">D 011 設計</td></tr>
<tr><td colspan="2">D 012 制作</td></tr>
<tr><td rowspan="2">D 02</td><td colspan="2">D 021 設計</td></tr>
<tr><td colspan="2">D ……</td></tr>
<tr><td rowspan="5">E. アプリデータ</td><td rowspan="4">E 01 自社商品</td><td>E 011 画像データ</td><td>E 0111 撮影</td></tr>
<tr><td>E</td><td>E 0112 データ化</td></tr>
<tr><td>E 012 説明書</td><td>E 0121 作成</td></tr>
<tr><td>E</td><td>E 0122 ……</td></tr>
<tr><td colspan="3">E 02 他社製品
……</td></tr>
<tr><td rowspan="2">F. ハードウェア</td><td>F 01 プロセッサ</td><td colspan="2">F 011 調達</td></tr>
<tr><td>⋮
F 02 ……</td><td colspan="2">F 021 ……</td></tr>
</table>

図4.1 WBSの例

4.2 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。

クリティカルパスを明確にすることのメリットとしては、以下のような点があります。

- **計画策定時**
  - クリティカルパスを短くすることで、プロジェクト全体の納期を早くすることができる
  - クリティカルパス上のリスクを特定し、事前に対応策を検討できる
- **プロジェクト遂行時**
  - 問題発生時、クリティカルパス上の作業を遅らせないよう対策することで、プロジェクト全体の遅延を防止できる

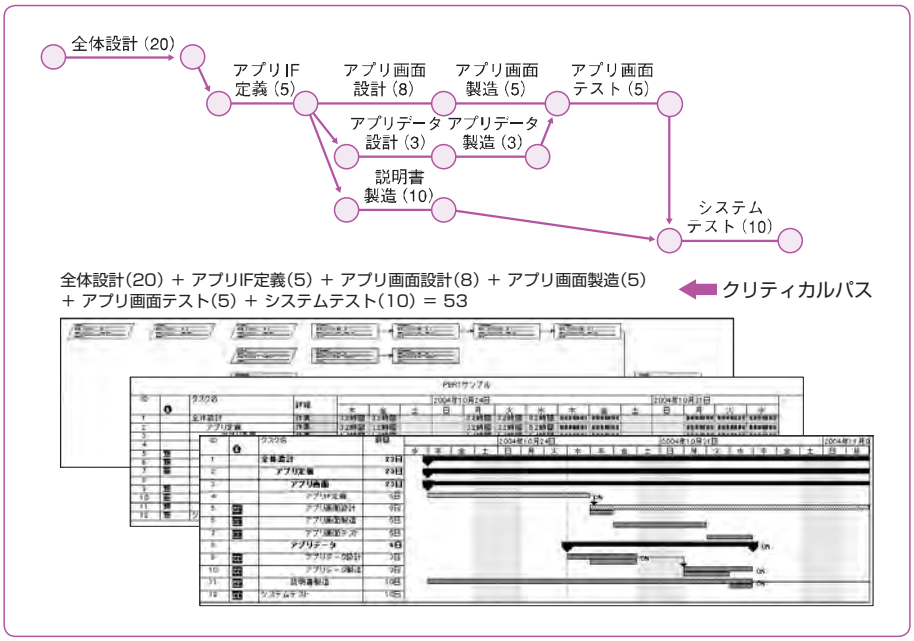


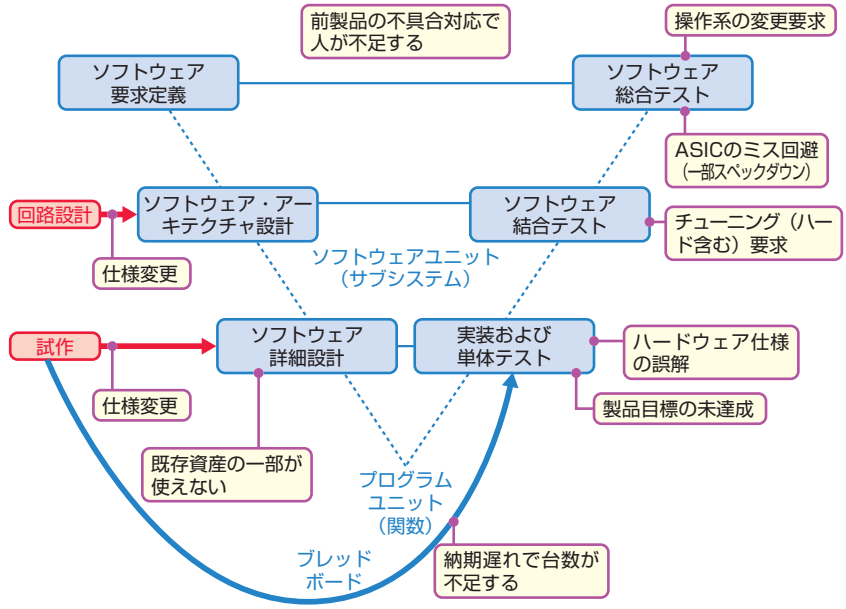
図4.2　PERTの例とクリティカルパス

4.3　プロジェクトリスクとリスク対応計画

リスク対応計画は、不確実な要素をあらかじめリスクとして特定し、対応策を計画しておくことです。

どんなプロジェクトでも不確実な要素（リスク）は含まれています。リスクを洗い出すときには、問題となりそうな現象をリスクとしてとらえるだけではなく、その原因（リスクソース）を洗い出す必要があります。リスクの根源に対策を打つことで、単一の問題だけでなく、そのリスクソースから生まれる種々の問題を未然に防止することができるためです。リスクソースを分類することにより、リスク特定時の発見漏れを少なくすることができます。

また、プロジェクトは進行具合によって状況が変化することから、定期的にリスクの見直しを行い、常に重要度の高いリスクに注意を払うことが必要です。



リスク	計画書で留意する点
ハード仕様変更	計画上の余裕（期間・技術者・設備・機器）を持つ
既存資産の問題	早期の見極め、影響範囲の特定と対策の計画化
試作機台数不足	必要な作業に優先的に配置する
製品目標の未達成	積極的なリソース（技術者・設備・機器）割り当て
ハード仕様誤解	十分なレビュー期間とハード担当者のレビューへの参画
チューニング	計画上の余裕（期間・技術者・設備・機器）を持つ
ASIC不具合	計画上の余裕（期間・技術者・設備・機器）を持つ
操作性要求変更	計画上の余裕（期間・技術者・設備・機器）を持つ
前機種不具合による技術者不足	計画上の余裕（期間・技術者・設備・機器）を持つ スキルを持つ技術者のレビューへの参画

図4.3　組込みソフトウェア特有のリスク

4.4　開発プロセスと工程設計

開発プロセスは、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。

ソフトウェアというモノを一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ります。この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びます。開発プロセスに含まれる作業には、要求定義、アーキテクチャ設計、実装、テストなどがあります。

ソフトウェア開発プロセスについては、これまでもISO/IEC12207 をはじめとする様々な考え方が提唱され実践されてきています。SECでは、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『組込みソフトウェア向け 開発プロセスガイド』(ESPR Ver 1.0)を発刊しています。

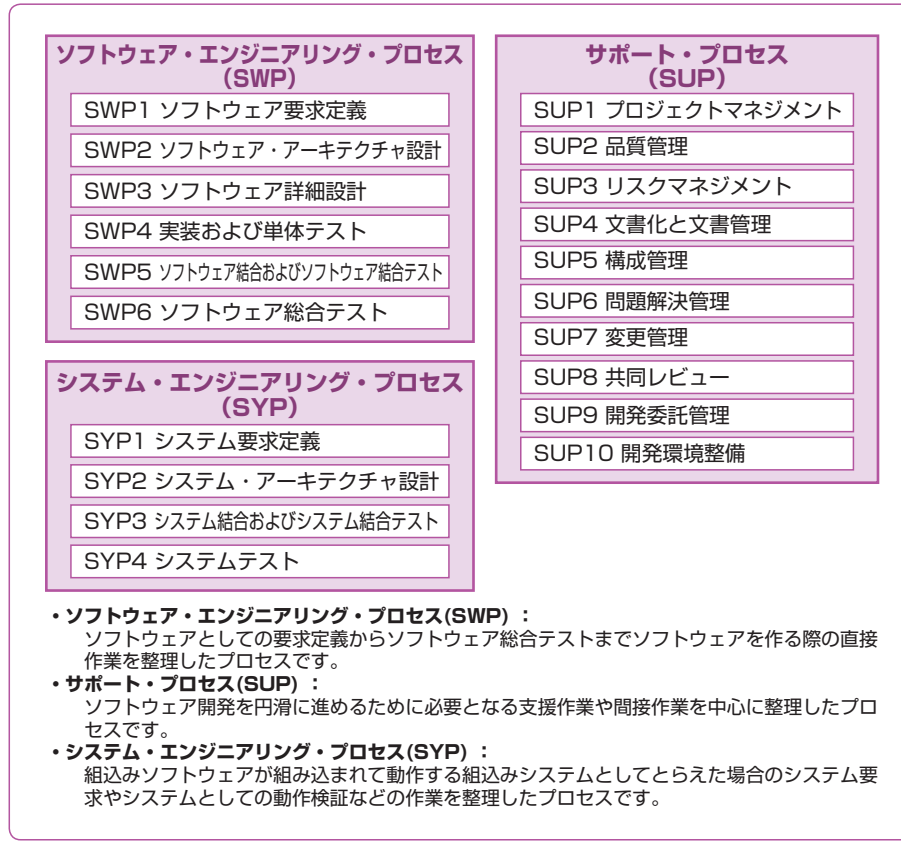


図4.4　『組込みソフトウェア向け 開発プロセスガイド』(SEC発刊)の開発プロセス

工程設計は、開発作業を実際の時間軸上に割り付ける作業です。

工程設計は、開発プロセスを利用して、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がら、作業順序、マイルストーン、各作業にかけるウェイトや作業内容を検討し、実際の時間軸上に開発作業を割り付けます。

また、実際に割り付けたものを開発工程と呼びます。

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を決めていきます。

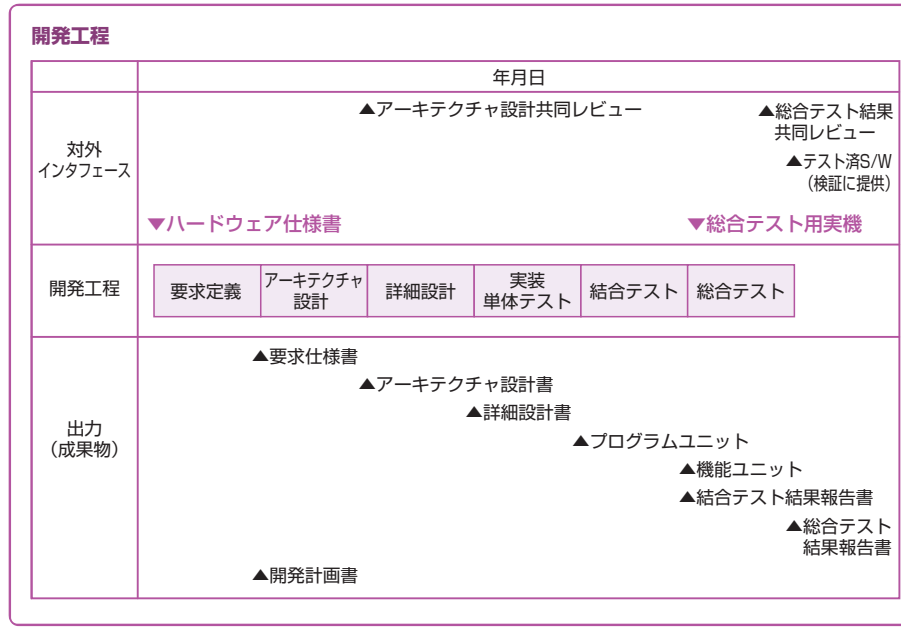


図4.5　開発工程

4.5　見積り手法

見積り手法には、類推見積、係数モデル見積、ボトムアップ見積などがあります。

類推見積（Analogous Estimating）：過去の事例や経験から類推して見積もる

類推見積りは、過去の類似のプロジェクトの実績値から類推して見積る方法で、トップダウン見積りとも呼ばれます。

プロジェクトの初期段階などで、見積りに関する情報が少ない場合に適しています。過去の実績値を収集、分析している場合には精度が高くなる特徴があります。

係数モデル見積（Parametric Modeling）：係数モデル（数式）に当てはめて算出

工数などを目的変数として、説明変数に規模や要因などを設定し、係数モデル（数式）に当てはめて見積もる方法です。

再現性があり、条件を変えることでシミュレーションが可能である、などの特徴がありますが、数式において不確定な変数があると誤差が大きくなります。

プログラムのソースライン数と生産性からプログラム開発工数を見積る方法や、ソースライン数の代わりにファンクションポイント値を使用するなどの方法があります。

ボトムアップ見積（Bottom-up Estimating）：WBSなどで作業を分割して積み上げる

WBSなどでプロジェクトの作業を洗い出し、それぞれに必要な工数を見積って積み上げる方法です。

見積の精度は、作業の洗い出しの精度に依存します。このため、新規開発プロジェクトの初期段階など、見積りに関する情報が少ない場合には適用が難しいですが、類似のシステムを繰り返し開発している場合で、かつ過去の情報を収集、分析している場合には、見積りの精度が高くなります。

4.6　品質特性モデルと品質メトリクス

品質特性モデルは、ソフトウェアを機能性、信頼性、使用性、効率性、保守性、移植性の6つの視点から評価する指標です。

組込みソフトウェアは、ハードウェアと一体となって初めて意味をなすものであるため、多くのソフトウェアの中でも、特に捉えどころがないものです。しかしその一方で、一度不具合を出せば、その影響は計り知れないほど広い範囲に及ぶこともしばしばです。

このため、開発している組込みソフトウェアがどのような特性をもったものであるかを早めに察知し、その特性に応じて対策を講じることがきわめて重要になってきています。

たとえば、ソフトウェアの信頼性などの観点で、ソフトウェアのある部分を新人が担当しており、構造的にも複雑で分かり難くなってしまった場合、こうしたことがあらかじめ数字として判明していれば、その数字を根拠に、その部分を徹底的にレビューするとか、テストするといった対策を講じることができます。

米国のBehm氏らが提案したソフトウェア品質特性モデルをベースに作られたISO/IEC9126は現在、ソフトウェア品質の計測の視点として広く認知されています。代表的な視点として、この規格では**機能性**、**信頼性**、**使用性**、**効率性**、**保守性**、**移植性**の6つの視点から評価するように指導しています。

外部および内部品質

特性	副特性	説明
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	ソフトウェアが、指定された条件の下で利用されるときに、明示的および暗示的必要性に合致する機能を提供するソフトウェア製品の能力。
信頼性	成熟性 障害許容性 回復製 信頼性標準適合性	指定された条件の下で利用するとき、指定された達成水準を維持するソフトウェア製品の能力。
使用性	理解性 習得性 運用性 魅力性 使用性標準適合性	指定された条件の下で利用するとき、理解、習得、利用でき、利用者にとって魅力的であるソフトウェア製品の能力。
効率性	時間効率性 資源効率性 効率性標準適合性	明示的な条件の下で、使用する資源の量に対比して適切な性能を提供するソフトウェア製品の能力。
保守性	解析性 変更性 安定性 試験性 保守性標準適合性	修正のしやすさに関するソフトウェア製品の能力。修正は、是正もしくは向上、または環境の変化、要求仕様の変更および機能仕様の変更にソフトウェアを適応させることを含めてもよい。
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	ある環境から他の環境に移すためのソフトウェア製品の能力。 **備考**：環境には組織、ハードウェアまたはソフトウェアの環境を含めてもよい。

利用時の品質

特性	説明
有効性	利用者が指定された利用の状況で、正確かつ完全に、指定された目標を達成できるソフトウェア製品の能力
生産性	利用者が指定された利用の状況で、達成すべき有効性に対応して、適切な量の資源を使うことができるソフトウェア製品の能力 **備考**：資源には、作業を完了するまでの時間、利用者の労力、材料または使用した費用を含めることができる
安全性	利用者が指定された利用の状況で、人、事業、ソフトウェア、財産または環境への害に対して、容認できるリスクの水準を達成するためのソフトウェア製品の能力 **備考**：リスクは、一般に、機能性（セキュリティを含む）、信頼性、使用性または保守性の欠陥の結果である
満足性	指定された利用の状況で、利用者を満足させるソフトウェア製品の能力 **備考**：満足性は、製品を対話的に利用したときの利用者の反応であり、製品の利用を含む

品質メトリクスは、ソフトウェアの信頼性を測るための尺度（メトリクス）です。

プログラムの複雑さは、条件分岐などが多く含まれるほど、そしてその階層が複雑なほど、高いといえます。これらを論理的に整理した尺度がMcCabeのCyclomatic Numberと呼ばれる尺度です。この尺度を計測するには、プログラムを図4.6のようなフロー図に変換し、その上で、各ノード、エッジなどの数を求め、これらから複雑さを計算します。なお、この値は一般に条件分岐数に1を加えた値で近似されます。

このCyclomatic Numberが10を超えるとプログラムとしてはかなり複雑であるという見解は、この分野での常識となっています。またプログラムの複雑さと混入される不具合の数の間には直線的な相関関係が成り立つことも立証されています。

このため、Cyclomatic Numberなどをモニタすることで極めて複雑な部分を早めに検出し、レビューやテストなどを重点的に行い不具合を押さえ込むといった対策をとることが重要になります。

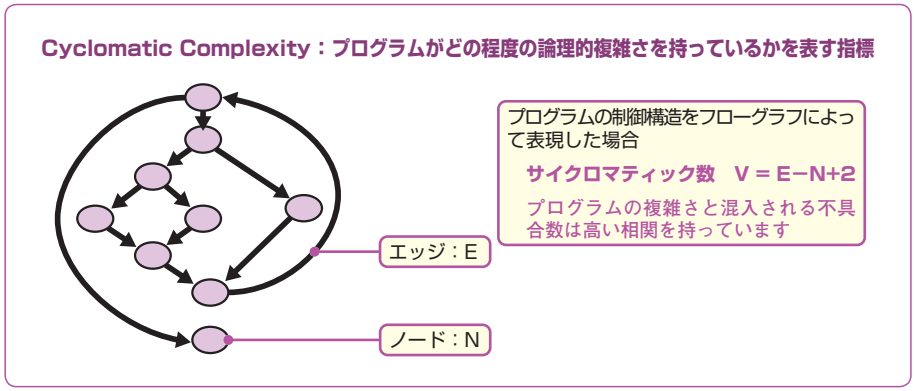


図4.6　プログラムの複雑さ

さらにソフトウェアの複雑さとサイズの間にもかなり強い相関関係があることが知られています。

図4.7のように、複雑さとサイズを軸にとった管理図を作成することで、ソフトウェア内の信頼性の低い危険個所をあらかじめあぶり出すことが可能になります。

信頼性の観点からは、複雑さについてCyclomatic Numberが10を超えるモジュールをピックアップできます。また、保守性の点からは、極端にサイズの大きなモジュールは保守が難しくなるため、ある一定サイズ以上のモジュールをピックアップできます。

さらにこれ以外にも、サイズの割にやたらと複雑さが大きい、あるいはサイズの割にやたらと複雑さが小さいといった相関上のはずれ値などもピックアップできます。

このようにしてピックアップしたモジュールやクラスを重点的にレビュー＆インスペクションしたり、テストしたりといった対策を講じることが有効になります。

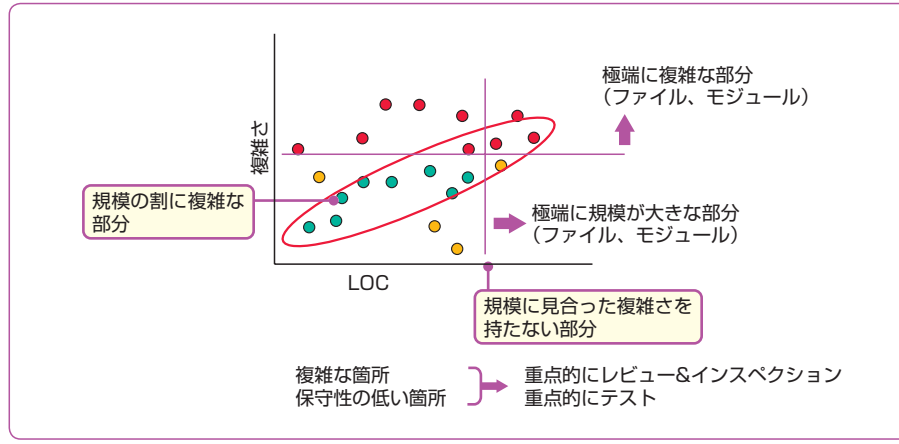


図4.7　品質メトリクスの利用

あとがき

経済産業省 組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 プロジェクトマネジメント技術部会では、2004年10月の発足以来、組込みソフトウェア開発におけるプロジェクトマネジメントとはどうあるべきかについて検討を重ねてきている。

従来の組込みソフトウェア開発は、開発規模や関わる人員数が現在と比べ小さく、また良い意味での職人的風土もあいまって、勘と経験で開発を全うできる状況であったが、近年の爆発的な開発規模の増大とそれに伴う開発人員の増加をはじめとする諸条件の変化により、従来のやり方では収拾がつかなくなってきている。

こういった状況を鑑み、プロジェクトマネジメント技術部会では、組込みソフトウェア開発の現場に対し、まず2005年に「プロジェクトマネジメント」の考え方・知識を紹介する『組込みソフトウェア開発におけるプロジェクトマネジメント導入の勧め』（翔泳社刊）を発刊した。

さらに、実際のプロジェクトマネジメントの現場で活用できる情報が必要と考え、検討を重ね編纂したものがこの『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］』である。マネジメントの最初の段階である計画について共有情報としてのかたちとなる「計画書」を実際に作成していくことを主要な目的として編纂した。

今後もプロジェクトマネジメントの他の要素について検討を進め、同様に発刊していく予定である。

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る。

2006年11月
組込みソフトウェア開発力強化推進委員会

執筆者

主　査	福　井　信　二	オムロン株式会社
副主査	川　崎　健　司	日本アイ・ビー・エム株式会社
	井　沢　澄　雄	日本電気株式会社
	石　川　和　信	株式会社アプリックス
	大　野　克　巳	IPA/SEC（トヨタテクニカルディベロップメント株式会社）
	川原林　　隆	株式会社ルネサスソリューションズ
	熊　谷　　馨	キヤノン株式会社
	西　條　達　也	ソラン株式会社
	佐　藤　和　雄	日本ユニシス・ソリューション株式会社
	舘　内　嗣　治	株式会社ルネサステクノロジ
	長谷川　賢　一	富士通株式会社
	畑　　幸　一	松下電器産業株式会社
	室　　修　治	IPA/SEC（横河ディジタルコンピュータ株式会社）
	山　浦　恒　央	日立ソフトウェアエンジニアリング株式会社
	山　崎　太　郎	IPA/SEC（日本ユニシス株式会社）
	渡　辺　正　文	エルミック・ウェスコム株式会社
	平　山　雅　之	IPA/SEC組込みエンジニアリング領域幹事（株式会社東芝）

監修

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向け

プロジェクトマネジメントガイド［計画書編］

2011年6月15日発行

発行　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/）