

01

中小企業における
システム環境の現状

企業におけるシステムの活用範囲は、年々と広がっています。一方で、リソースが大企業に比べて限られている中小企業では特に、システムの開発に手が回らないことが多くあります。まずは中小企業とシステムに関連する現状について解説します。

◎ 本書における中小企業の定義

本書では、**中小企業のシステム開発**に携わる、次のような方を想定して解説しています。

- 中小企業に所属し、社内のシステム開発に取り組もうと考えている方
- 中小企業に所属し、社内のシステム開発を他企業に外注しているが、うまく進めるのが難しいと感じている方

基本的には上記を想定していますが、**ベンダー**（システム開発を行う企業）に所属し、中小企業向けのシステム開発に携わる方にとっても役立つ内容となっています。なお、中小企業基本法における中小企業の定義は次の通りです。

■ 中小企業基本法における定義

業種分類	中小企業基本法の定義
製造業その他	資本金の額又は出資の総額が3億円以下の会社 又は常時使用する従業員の数が300人以下の会社及び個人
卸売業	資本金の額又は出資の総額が1億円以下の会社 又は常時使用する従業員の数が100人以下の会社及び個人
小売業	資本金の額又は出資の総額が5千万円以下の会社 又は常時使用する従業員の数が50人以下の会社及び個人
サービス業	資本金の額又は出資の総額が5千万円以下の会社 又は常時使用する従業員の数が100人以下の会社及び個人

出典： <https://www.chusho.meti.go.jp/soshiki/teigi.html>

本書ではこれほど厳密に定義しませんが、おおむね次のような事業規模を想定しています。

■ 本書における中小企業の定義

従業員数	売上高
数十～数百人程度	数億～100億円程度

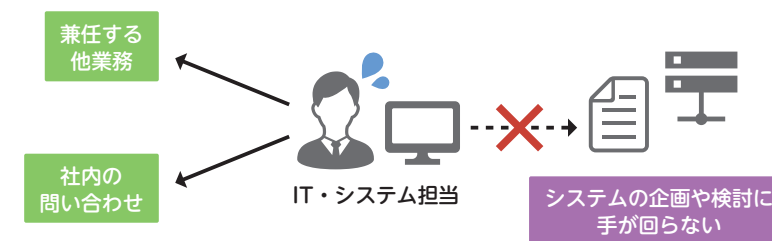
◎ 中小企業のシステム環境の実態（内部環境）

システムを取り巻く環境は、大きく企業の内部と外部に分けられます。まずは、中小企業における社内の実態（内部環境）として、よく見られる特徴について説明します。

中小企業においては、**IT・システムの担当者がいない**企業が少なくありません。担当者がいたとしても、次のような状況であることがほとんどです。

- 他業務との兼任で、IT・システム関連の業務には手が回っていない
- パソコンのセットアップ、社内の問い合わせ対応といった日常業務に追われ、経営者が期待する調査や企画、検討などの業務に手が回っていない

■ IT・システム担当者が企画や検討に手が回らない



また、中小企業では10年や20年もの間、業務のルールやプロセスの見直しを一切しておらず、FAXやExcelを駆使して仕事をしていることが多々あります。業務が見直されないと基幹システムの見直しもされず、**時代遅れになっているシステムを長い期間利用している**ことが多くなります。新しいシステムに

03

システム外注の流れ

システム開発を外注する（内製をしない）場合には、それを任せられるベンダーを選定する必要があります。ここでは、ベンダーの選定を含む、システム外注の流れについて、全体像を解説します。

○ システム外注では大きく6つのプロセスがある

システム外注の大まかなプロセスは、**企画**、**要求定義**、**ベンダー選定**、**開発**、**受入**、**運用・保守**の大きく6段階です。

■ システムを外注するときの流れ

工程	内容	本書で解説する章
企画	どんなシステムを開発するのかを考える	2章
要求定義	ベンダーへの要求を具体的に定義する	3章
ベンダー選定	外注するベンダーを選ぶ	4章
開発	システムを開発する	5章
受入	システム稼働の準備をする	6章
運用・保守	システムの不具合への対応や改善を行う	7章

それぞれの段階について、詳しくは2章以降で解説していきます。ここでは、各段階の概要だけ紹介するので、どんな工程があるのかを、簡単に押さえておきましょう。

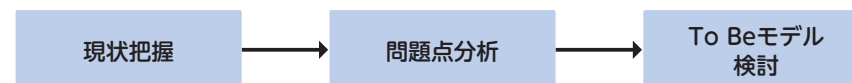
○ 企画

システムを外注するにあたり、まず実施するのが企画です。どんなシステムを開発する必要があるのかを考えていきます。

企画は、大まかに**現状把握**、**問題点分析**、**To Beモデル検討**の3つのプロセスに分かれます。なお、新規ビジネスのためのシステム開発などの場合は、現状把握や問題点分析のプロセスは省略するなど、多少の違いは発生します。

現状把握や問題点分析を行い、開発するシステムで何を解決・改善する必要があるのかを明らかにします。その後、それを解決・改善ができる「**To Beモデル（あるべき姿）**」を描きます。新規ビジネスなど、現状把握の対象がない場合は、現状把握や問題点分析を省略し、「**Will Beモデル（ありたい姿）**」を描きます（「Will Beモデル」は青山システムコンサルティング(株)の登録商標です）。企画した内容については、プロジェクトオーナーなどしっかりと議論し、合意を得る必要があります。

■ 企画の流れ



○ 要求定義

システムの企画ができれば、それをベンダーに伝えるための具体的な要求へと落とし込んでいく必要があります。業務の流れをまとめた**業務フロー**を作成し、システムに求める要求事項を一覧にまとめます（**要求定義**）。

内容については、業務部門とシステム部門のどちらかだけで作成・確認するのではなく、両者の視点で確認し、抜け・漏れ・認識違いがないようにする必要があります。社内の関係者が理解できる内容であることはもちろん、システム開発を担うベンダーが理解しやすい内容にするのもポイントになります。

■ 要求定義の流れ



09

システム外注に
必要な社内体制

発注者側の社内体制が不十分であったことに起因するシステム開発のトラブルは、よく発生します。システム開発をする際には、発注者側としての社内体制の整備は、とても大切なテーマです。

○ システム開発におけるプロジェクト体制とは

システム開発においては、プロジェクトの**ステークホルダー（利害関係者）**の責任と役割を明確にした体制作りが欠かせません。日々の業務で手一杯の中、システム開発の担当者をアサインするのは大変……というのは、どの企業でも抱える悩みではありますが、社内体制の整備はプロジェクトの成功に不可欠です。

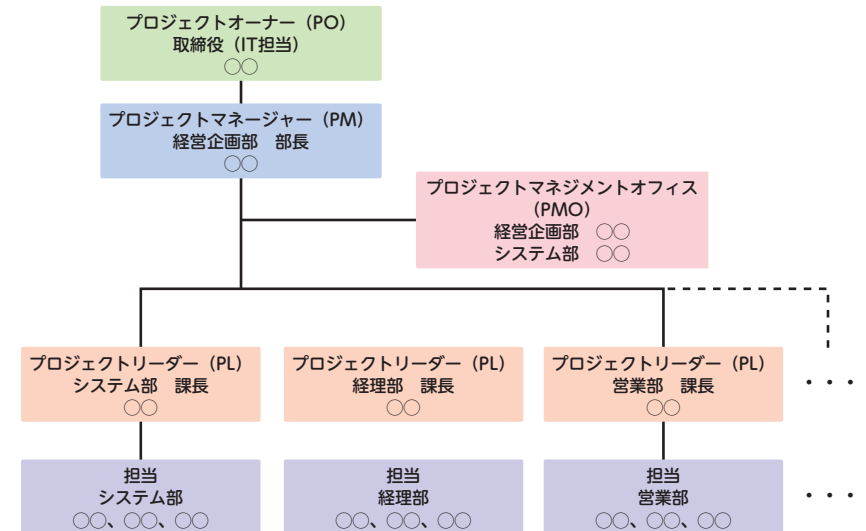
システム開発においては、次のような役割の担当者で体制が構成されます。

■ ステークホルダーと主な役割

ステークホルダー	主な役割
プロジェクトオーナー (PO)	プロジェクトの開始、終了を決定する プロジェクトに関連する重要事項（費用の増減、スケジュール変更、担当者アサイン、大きなタスクや課題の解決策など）の最終決定をする
プロジェクトマネージャー (PM)	プロジェクトのスケジュール、コスト、品質を管理する
プロジェクトリーダー (PL)	プロジェクトを計画の通りに推進する
プロジェクト担当者	プロジェクトリーダーの指示の下、各種タスクを実行する
事務局・プロジェクトマネジメントオフィス (PMO)	プロジェクトマネージャーをサポートする

それぞれのステークホルダーは、**発注者側、受注者（ベンダー）側のそれぞれに必要**であることが重要なポイントの1つです。上述したステークホルダーについて発注者側のプロジェクト体制図を表現すると、次のようになります。

■ 発注側のプロジェクト体制図の例



体制を決めたら、**各自が責任を持って、その役割を全う**しましょう。意思決定をしないPM、現場担当者任せのPL……というのはよく見かける光景ですが、それではプロジェクトは成功しません。

○ 社内体制の整備が不十分だとどうなるのか

社内体制の整備が不十分で、役割が明確になっていないままシステム開発を進めた場合、次ページの例のようなトラブルが発生することが多くなります。

また、同じ役割の担当者が複数いた場合、「誰かがやるだろう……」と考えていたのに実際には誰も手をつけず、スケジュールが遅延してしまうというトラブルも起こりえます。

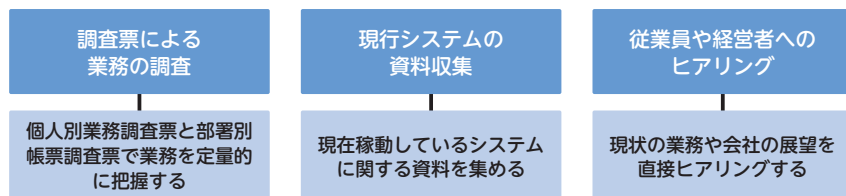
11 現状を誤解なく把握する

システムの企画では、自社の現状を把握することが必要です。本節では現状を誤解なく正確に把握する方法を解説します。なお、新規事業などで現行業務やシステムがない場合は、この節と次の節は読み飛ばしてください。

○ 現状把握の実施

企画で最初に行うことは**現状把握**です。現状を正確に把握せずシステム要求を挙げても、実情に合わないシステムが導入され、期待する効果が得られないためです。現状把握では、次のようなことを実施していきます。

■ 現状把握で実施すること



○ 調査票による業務の調査

どの業務に時間がかかっており、改善が必要なのは、感覚的にはわかりませんが、人間の感覚は誤っていることも多く、実は「他の業務と大差はない」ことが少なくありません。このようなことを防ぐには、何の業務にどの程度時間を要しているかを**調査票**で定量的に把握できるようにします。

現状把握では、**個人別業務調査票**と**部署別帳票調査票**の2種類の調査票を従業員に自己申告してもらいます。

個人別業務調査票は、**どのような作業をどのくらいの時間をかけて行っているかの作業の実態を客観的に把握する**ための調査票です。1日や1週間、1カ月といった一定の期間で、どの作業にどれだけの時間をかけているのかを入力

してもらいます。入力内容は、1人あたり10~20項目程度が目安です。

■ 個人別業務調査票の例

区分	正社員	部課名	工場部	部課種別	製造・開発・工場系	役職	課長	名前	鈴木 イチロウ			
基本業務	出荷調整・工場応援業務						勤続(年)	10.0	現職(年)	20.0	月間所要時間(平均)	170
分類	作業内容	作業区分	処理周期	月間処理量			月間所要時間(H)			問題点および改善点		
				最小	最大	単位	最小	最大	平均			
試験・検査	立会い業務・試験業務	手作業	随時	18	22	件	50	60	55			
社内資料作成	立会い業務・試験業務・資料作成作業	ERP	日次	15	22	件	30	44	35	入力項目が多いため、時間がかかる		
電話対応	電話対応業務	手作業	日次	3	20	件	10	20	15			
製造・開発	現場確認作業	手作業	日次	10	15	件	3	10	5			
その他	営業活動(工場訪問等)	手作業	日次	15	40	件	40	50	45	その場で見積書の作成および提出ができない		
その他	予定確認作業	手作業	日次	20	40	件	1	5	2			
社内資料作成	各種資料作成業務	Excel	日次	4	15	件	1	5	2			

部署別帳票調査票は、**システムが出力するものや、各部署においてExcelなどによりローカルで作成されている帳票数を把握するための調査票**です。システムが出力するものは、比較的簡単に種類や数を把握することができますが、それ以外に、各部署が独自にExcelなどで作成している帳票も多数存在することがあるので、ある程度時間をかけて集める必要があります。

■ 部署別帳票調査票の例

部課名	第一営業部													
No	帳票名	使用目的	作成区分	処理周期	取扱区分				帳票の流れ		月間データ量		問題点および改善点	
					作成	確認	入力	その他	入手先	配布先	最小	最大		
1	見積書		表計算ソフト	随時	○							144	150	基幹システムがすべての見積パターンに対応できていないため、Excelで作成している
2	配合計画書		会計システム	日次	○					工場部		30	30	
3	入金チェックリスト	商品受け入れデータシート	表計算ソフト	日次		○	○			経理部	経理部	10	30	Excelではなくシステム化してほしい
4	デリバリーシート		その他	日次	○					工場部		3	10	
5	作業日報		その他	日次		○						40	50	手書きのため、時間がかかる

19 予算の上限を決める

RFI発行後、各ベンダーからの回答資料を基に資料の内容から費用感と実現可能性を把握します。この節では、回答を比較・評価して把握する方法と、自社のシステム導入予算の上限を決める際のポイントを解説します。

最後にベンダーからの回答を評価する

ベンダーからRFIの回答を受領し、製品やサービスの情報を幅広く収集した後は、最後のステップとして**回答を比較・評価し、費用感と要件の実現可能性を把握**します。具体的には次のような手順で行います。

- ① 評価項目を決める：自社がベンダーやシステムに期待する項目を設定する
- ② 評価項目の配点を決める：各項目の重みを考慮して点数を割り振る
- ③ 評価基準を決める：各項目に対して3段階や5段階などの基準を設ける
- ④ 評価を行う：項目や基準に沿って評価を実施する
- ⑤ 評価結果を比較する：ベンダー各社の定量化した評価結果を集計し、内容を協議する

評価項目は、選定対象にするシステムや自社の状況に応じて変える必要がありますが、一例として次のようなものがあります。

■ 評価項目の例

ベンダーの信頼性	ベンダーの会社規模、財務状況、導入実績などから信頼できる企業であるかを評価する
回答の妥当性	現状の課題や要望を理解し、適切な回答であるかを評価する
システムの対応分野の適合性	自社の業界や対象業務に対応できるかなどを評価する
導入・運用費用	見積り金額が想定を遥かに超える金額になっていないか、算出根拠は明確であるかなどを評価する

このように、評価項目を設定すると、ベンダー間の差異、強み・弱みを明確にでき、比較しやすくなります。

また、評価基準は段階別に点数をつけて評価するといったように、可能な限り**定量的に評価**することがポイントです。そうすれば、客観的な評価ができ、主観的な評価を避けることができます。

なお、**色や記号を使って直感的にわかりやすく**することも大切です。例えば、色は緑や赤など対立する色でポジティブ・ネガティブな部分を示します。

RFIの評価結果の例は、以下の通りです。

■ 評価結果の例

評価項目	評価基準	配点	A社		B社		...
			評価	評価点 (配点×評価)	評価	評価点 (配点×評価)	
企業規模	3：大手 2：中堅 1：中小	5	2	10	1	5	...
導入実績	3：実績十分 2：実績普通 1：実績不足	10	3	30	2	20	...
RFIの理解	3：十分 2：普通 1：不十分	5	3	15	2	10	...
回答の差別化ポイント	3：予想以上 2：妥当 1：不足	5	2	10	2	10	...
システム範囲適合	3：適合 2：部分的に適合 1：不適合	10	2	20	3	30	...
費用 (10年間)	3：1億円未満 2：1億円～2億円 1：2億円以上	20	1	20	2	40	...
...
評価点合計	150	...	170	...

21

To Be業務フローを作成する

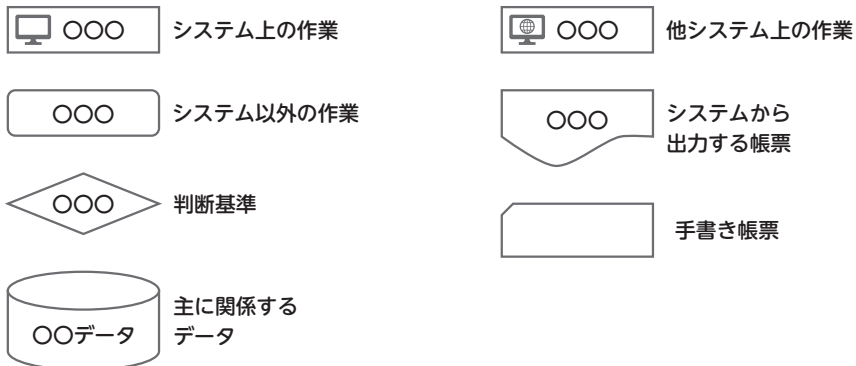
To Be業務フローは、事前知識のない業者へと自社の業務内容を伝える資料ですが、社内の人間にも理解してもらう必要があります。一番大切なのは「読み手が理解できること」です。ここでは、To Be業務フローの作成方法について解説します。

○ 業務フローに記載する内容

To Be業務フローを作成する際は、**ExcelやPowerPointで作成すること**をおすすめします。業務フローの作成に特化したツールを利用する方法もありますが、そうしたツールで作ったものは、再利用しづらかったり、共有の際に見方がわからなかったりします。他社が理解しやすいものだけではなく、社内の一般従業員が理解でき、その後も活用できるものを作成しましょう。もしすでに社内に業務フローがあれば、それを活用します。

業務フローで区別すべき主な内容とその記号は、次のようなものです。

■ 業務フローで主に記載する内容とフローチャートの記号（例）



上記の記号を使って、フローチャートに記載する内容の詳細は次の通りです。

■ フローチャートに記載する内容

内容	概要
手作業で行う業務内容	現場の担当者が手作業で行う必要がある作業。手作業で行う業務を記載すると、新システムでの機能を加えることで手作業がなくなるといった、業務改善のアイデアにもつながる
システムで行う作業内容	実際にシステムで行うことを想定している作業。データの登録・参照・更新・削除 (P.109参照) を意識して記載する
分岐・判断基準	処理の分岐を表す。多用すると複雑怪奇なフローになるので注意。実際のフロー上では、できるだけ少なくし、分岐要素や判断要素をコメントとして記載することが多い
帳票	手書伝票なのかコンピュータから出力された帳票なのかを明示する。具体的には、「Aシステムから出力した帳票を(システムの作業)、担当者が特定の区分に集計してBシステムに入力し(手作業)、伝票を出力する」といったシーンで記載する
データベース	システムそのもののデータベースやCSVデータ、テキストデータなど。データの登録や修正、削除、(検索のための)参照を意識して記載する。この時点(要求定義)では、データ項目までは厳密に定義する必要はない。ただ、業務フロー上にとあるデータ項目を修正するフローが出てくると、データ登録時にその修正対象データが含まれていることがわかる

過去にシステム会社が作成した業務フローを流用する場合には注意しましょう。現行システム上での作業のみに特化した業務フローになっており、手作業で行う内容が十分に記載されていないことが多いです。また、後から読み込んでも業務全体の流れがうまく把握できないこともよくあります。そのため、To Be業務フローの作成時に手作業で行う業務を表現していくことは、業務全体像の把握にも役立ちます。

○ 登場人物を漏れなく表現する

業務フローには**登場人物(登場部署)を漏れなく記載**しましょう。例えば、受注登録のフローであれば、お客様、営業、受注センター、倉庫、工場などが挙げられます。また、システムそのものを1つの登場人物と考えて記載することも多くあります。

24

要求定義には
現場を巻き込む

To Be業務フローや要求機能一覧を作成する際に、システム部門や企画部門だけで作成することがあります。しかし、システムを操作するのは、各事業部の方々です。適切なタイミングで利用部門へレビューをしましょう。

○ 総論賛成・各論反対

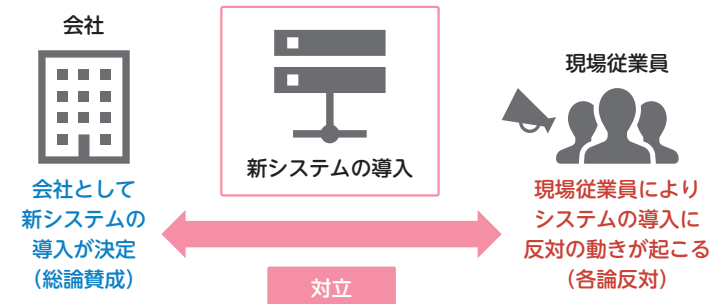
システムのプロジェクトは「**総論賛成・各論反対**」になることが多くあります。総論賛成・各論反対とは、全体としては賛成であるものの、各部門では反対が起きるという状態です。この結果として起きる例で有名な話は、「市として国道を通すことは決まったが、用地買収がうまくいかない」という事例です。

■ 総論賛成・各論反対の例



同じようなことがシステムの導入でも起きます。経営側がシステム導入を決定したものの、**システム導入の目的を理解していない一部現場の従業員が、変化を好まず現状維持を望み、結果としてシステム変更の反対が起きてしまう**というケースです。

■ システム導入における総論賛成・各論反対の例

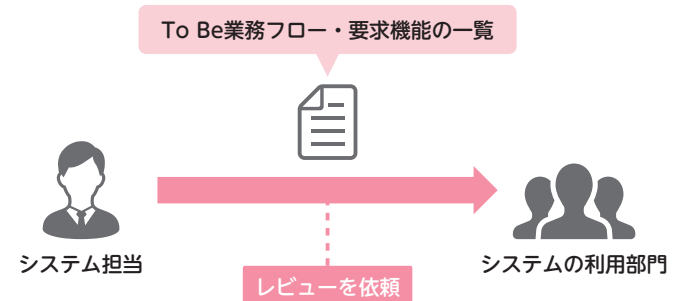


「なぜシステムを変更するのか」「新システムでどのような効果が生まれる想定なのか」を業務現場に理解してもらうには、ある一定の時間と手間が必要です。経営側のパワープレイで解決してしまうこともあります。その場合は結局、導入作業時にしっぺ返しを喰らうことがほとんどです。

○ 利用部門へレビューを依頼しよう

現場で実際にシステムを利用する部門の方々に、システムの導入を理解してもらうためには、作成したTo Be業務フローや要求要件を利用部門に**レビュー**してもらいましょう。これにより、導入目的や想定される効果を理解してもらえます。

■ システムの利用部門にレビューを依頼する



33 開発手法の種類

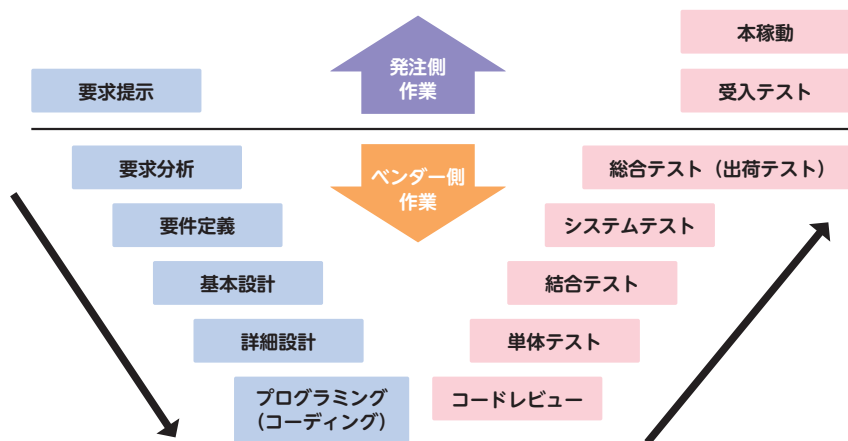
システム開発方式には、大きく分けてウォーターフォールとアジャイルの2つがあります。ベンダーは、これらの考え方を基本とした方式で開発の提案をします。ここでは、2つの開発方式の特徴や注意点について説明します。

○ 開発手法1：ウォーターフォール

ベンダー選定が完了したら、いよいよシステムの開発へと進んでいきます。システム開発の基本的な方式には、大きく分けてウォーターフォールとアジャイルの2種類があります。

まず、ウォーターフォールについて説明します。ウォーターフォールとは、開発工程を順番に1つずつ進めてシステムを完成させる開発手法です。要求提示から本稼働までに必要な工程を順番に進めていきます。**工程が滝のように下に流れていくイメージから、ウォーターフォールと言います。**また、次の図のように、開発前半の設計工程と後半のテスト工程を対比させて、V字モデルと記載することもあります。

■ ウォーターフォール（V字モデル）の工程



ウォーターフォール開発には、次のような工程があります。また、各工程の終了時には、発注側はベンダーから成果物を受領し、工程が正しく終了したかどうかを判定する終了判定を行います。

■ 工程ごとの主な作業や成果物

工程	特徴	成果物
要求提示	要求をとりまとめる	提案依頼資料
要求分析	主にヒアリングなどを通じて要求を詳細に分析する	議事録
要件定義	・開発内容の基本方針を作成する ・システムのデザインを可視化する	要件定義書
基本設計	基本方針に則った、システム設計をする	基本設計書
詳細設計	プログラム仕様書を作成する	詳細設計書
プログラミング	プログラムを作成する	・開発進捗表 ・プログラム
コードレビュー	正しいプログラムが書けているかを確認する	開発進捗表
単体テスト	プログラムが仕様通りに動作することを確認する	単体テスト結果報告書
結合テスト	単体で作成したプログラムがあるひとまとまりで仕様通りに動作することを確認する	結合テスト結果報告書
システムテスト	機能単位で、システムが仕様通りに動作することを確認する	システムテスト結果報告書
総合テスト（出荷テスト）	システム全体単位で仕様通りに動作することを確認する	総合テスト結果報告書
受入テスト	システムが実運用で利用できることを確認する	—

ウォーターフォールは、納品されたシステムのソフトウェアの品質が高い傾向にあるということがメリットです。

一方で、前工程の残タスクや後になってからの追加要求が発生すると、手戻り作業（前工程の作業のやり直し）が多くなります。これは、完了した工程の内容と不整合がないかを確認したり、一旦納品した成果物を修正したりと、発注側（ユーザー）が考えるよりも作業が多いことが理由です。

また、工程の終盤にならないと、発注側が動作するソフトウェアにさわらな

39 受入テストの流れと注意点

開発方式にもよりますが、実際に動作するシステムにはじめてさわることができるようになるタイミングが、受入テストです。一般に、このプロセスでさまざまな問題が判明します。

○ 受入テストとは

ベンダーによるシステム開発が完了したら、すぐにシステム稼働となるわけではありません。開発を依頼したシステムが、発注側（自社）が要求した通りの動きをするかを確認しないといけません。その確認をする工程が、**受入テスト**です。

受入テストは、**開発されたシステムを使って正常に業務ができることを確認するプロセス**です。どの工程も作業負荷はかかりますが、**受入テストが発注側にとって最も作業負荷がかかる工程**です。限られた期間で、システムが使えるレベルにあるかどうかを判断するため、集中して作業しましょう。

受入テストが大変な理由に、次のようなことが挙げられます。

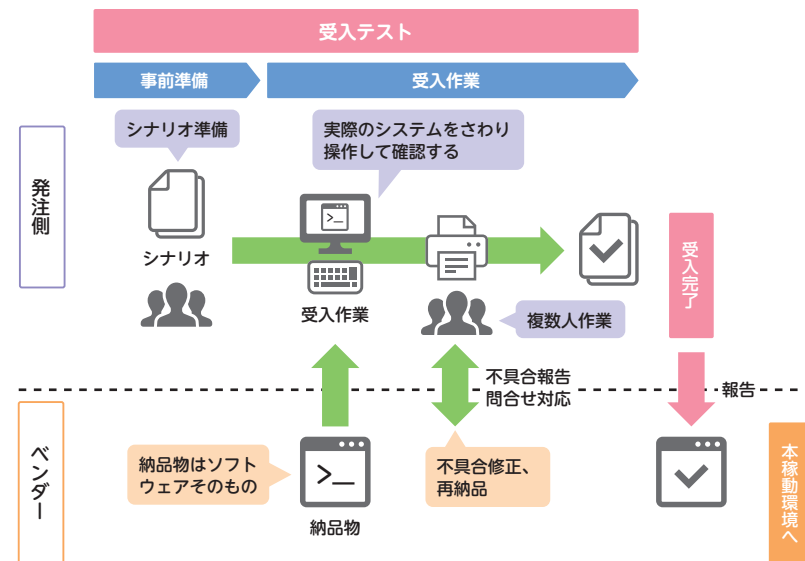
- システムがさわられるようになってから、真剣に考え出すメンバーがいる
- ベンダーから提示される契約書の雛形では、受入テストの期間が短く定められており、契約後にそのことに気づく
- ベンダーの関与度が低下し、発注側が主体性に動く必要がある

また、受入テストは、これまでの工程の作業品質の悪さが表面化する工程でもあります。前の工程で問題があると、バグや仕様不具合が混在した状態での受入テストとなり、結果として修正、改修が多発するもぐらたたき状態になってしまうことが多々あります。

○ 受入テストの流れ

ここまでの開発工程の判定と比べて、受入テストは、発注側の主体的な取り組みが求められる工程です。受入テストは、次の図のように、事前準備と受入作業という流れで進んでいきます。

■ 受入テストの流れ



ここでは、受入テスト時の注意点について説明していきます。

○ 受入テストのシナリオは事前に準備する

シナリオとは、テストの一連の流れのことです。受入テストでは、システムが実際に業務で問題なく使用できるかを確認できるようなシナリオを、発注側が作成しないといけません。

受入テストの成功・失敗は、**事前にシナリオを準備できるかどうか**に左右されます。受入テスト期間に入ってから、シナリオ作成に着手すると、システム機能に左右されたシナリオになってしまい、業務に合っているかを判断できないことが多くなります。そのため、5章でも説明しましたが、**開発工程で発注**

48

ドキュメントの
メンテナンス

システム開発プロセスの各工程終了後やシステム稼働後には、ベンダーからさまざまなドキュメントが提出されます。保管も重要ですが、稼働後に改善や改良した内容は、都度都度ドキュメントに反映することが大切です。

ドキュメント一覧を作成する

システム稼働後は、ベンダーから納品されたドキュメントをとりまとめる時間を作り、一覧表を作成しましょう。その中で、ベンダーが更新すべきものと発注側（ユーザー）で更新すべきものを区分けしていきます。ドキュメントを整理する際は、ウォーターフォール開発時の工程（P.162参照）を参考に、区分けをすると理解しやすいです。次のように、開発工程で作成されたもの（開発請負契約での成果物）と、要件定義工程や稼働準備工程で作成されたもの（準委任工程での成果物）とで、大まかに分類するとわかりやすくなります。

ドキュメント一覧例

工程	契約	ドキュメント	更新担当	更新タイミング
要件定義	準委任	要件定義書	発注側	プログラム変更時
		業務フロー	発注側	必要時
稼働準備		受入テスト仕様書	発注側	必要時
		業務手順書	発注側	必要時
開発	請負	設計書	ベンダー	プログラム変更時
		テスト仕様書	ベンダー	プログラム変更時
		テーブル定義書	ベンダー	プログラム変更時
		操作説明書	ベンダー	プログラム変更時

ドキュメントを整理する期間を設ける

システムの稼働後には、**ドキュメントを整理する期間**をしっかりと取りましょう。プロジェクト期間中、各メンバーは無意識にドキュメント間の大きな行間を読みとってプロジェクトを進めています。ここでいう「行間」とは、設計書に記載はないが、要件定義書には記載がある事項など、明記されていなかったりドキュメント間で記述に差異があったりすることです。

しかし、次のプロジェクトでも同じメンバーが集まるとは限りません。本来は、ドキュメント間に大きな行間があってはいいませんが、自社としてノウハウを蓄積するためにも、ある一定期間をとって（多くの場合は1週間程度）、一覧化や整合性の確認、更新担当者の割り当てなどの整理をしましょう。

ドキュメントを公開して定期的に更新する

ドキュメントを一覧化していても、更新がおろそかになってしまうのが現実です。更新していないことを他の人が指摘するなど、更新の必要性に気がつくようにするため、ドキュメントは誰でも参照できるような場所に保存しておくことが重要です。特に、ドキュメントに修正や追記が多いことが想定されるのであれば、**更新担当や更新タイミングを事前に決めておきましょう**。

システム改修時にドキュメントを更新する

ベンダーがシステムを改修したら、システムだけではなく、**更新されたドキュメント類の納品**も忘れずに依頼しましょう。

しっかりしているベンダーであれば、発注側が言わなくてもドキュメントは納品されます。改修発注時や保守対応の際のドキュメント納品は、保守契約に当然含まれると考えている発注側の企業も多いですが、明確に事前に伝えておかないと資料を更新してくれないベンダーもいます。そのため、保守契約時に、保守作業にドキュメントのメンテナンスが含まれていることを確認しましょう。