30　ドキュメンテーション

目的

本章の目的は、大規模なソフトウェア・システムで作成される可能性のある、 様々な種類のドキュメントについて説明し、高品質のドキュメントを作成するための ガイドラインを提示することである。この章を読むと、以下のことがわかる：

アジャイル手法によるシステム開発であっても、なぜシステム文書を作成することが重要なのかを理解する；

文書作成に重要な基準を理解する；

プロフェッショナルな文書制作のプロセスを紹介した。

内容

大規模なソフトウェア開発プロジェクトでは、アプリケーションに関係なく、大量の関連文書が作成される。これをすべて印刷するとなると、中程度の規模のシステムであればファイリング・キャビネット数個分、外部認証を受けなければならないような超大規模なクリティカル・システムであれば、部屋数個分になってしまうかもしれない。特に規制対象システムの場合、ソフトウェア・プロセス・コストの高い割合が、この文書作成に費やされる。さらに、ドキュメンテーションの誤りや漏れは、エンドユーザーによるエラーや、それに伴うコストや混乱を伴うシステム障害につながる可能性がある。したがって、管理者とソフトウェア・エンジニアは、ソフトウェアの開発そのものと同じくらい、ドキュメンテーションとそれに関連するコストに注意を払う必要がある。

ソフトウェア・プロジェクトと開発されるシステムに関連する文書には、多くの関連要件がある：

1. 開発チームのメンバー間のコミュニケーション媒体として機能するはずだ。
2. メンテナンス・エンジニアが使用する情報リポジトリであるべきだ。
3. 経営陣がソフトウェア開発プロセスを計画し、予算を立て、スケジュールを立てるのに役立つ情報を提供しなければならない。
4. ドキュメントの中には、ユーザーにシステムの使い方や管理方法を伝えるべきものもある。
5. これらは、システム認証のために規制当局に提出する不可欠な証拠となるかもしれない。

これらの要件を満たすには、非公式な作業文書から専門的に作成されたユーザーマニュアルまで、さまざまなタイプの文書が必要です。プロのテクニカルライターが、外部に公開される情報の最終的な推敲を支援することもあるが、通常、ソフトウェアエンジニアがこのような文書のほとんどを作成する責任を負う。

大規模なプロジェクトの場合、開発プロセスが始まるかなり前から文書が作成され始めるのが普通である。システム開発の提案書は、外部顧客からの入札依頼に応じて作成されたり、その他のビジネス戦略文書に対

応して作成されたりする。システムの種類によっては、システムの必要な機能と期待される動作を定義した包括的な要件文書が作成されることもある。開発プロセスにおいては、プロジェクト計画書、設計仕様書、テスト計画書など、さまざまな文書が作成される。

どのようなシステムであっても、作成しなければならない文書一式は、システムの顧客との契約、開発されるシステムの種類と予想される耐用年数、システムを開発する会社の文化と規模、予想される開発スケジュールによって異なる。しかし、ソフトウェアプロジェクトで作成されるドキュメントは、通常2つのクラスに分類される：

1. *プロセス文書* これらの文書は、開発と保守のプロセスを記録する。計画、スケジュール、プロセス品質文書、組織やプロジェクトの標準がプロセス文書である。

2. *製品ドキュメント* 開発中の製品を説明するドキュメント。システム・ドキュメントは、システムを開発・保守するエンジニアの視点から製品を説明し、ユーザー・ドキュメントは、システム・ユーザーを対象とした製品説明を提供します。

プロセスドキュメントは、システムの開発を管理できるように作成され、ソフトウェアエンジニアリングの計画駆動型アプローチに不可欠な要素である。アジャイルアプローチの重要な目標は、プロセスドキュメントの作成量を最小限に抑えることである。

製品ドキュメントは、システムが稼動した後に使用されるが、システム開発の管理にも不可欠である。システム仕様書のような文書の作成は、ソフトウェア開発プロセスにおける重要なマイルストーンとなる。

アジャイル手法の支持者が主張するのは、文書化の必要性が「伝統的な」ソフトウェアプロセスの大きな問題の1つであるということである。ドキュメンテーションを作成し、維持するためには多くのコストと時間がかかり、必然的にシステムの生産が遅くなる。彼らは、要求事項が急速に変化するため、ドキュメンテーションは書かれるとほとんどすぐに古くなり、本質的に無価値になると主張する。

私はこの意見に大いに共感する。文書が読まれず、古くなっていることはよくあることであり、次のセクションで述べるように、必要なプロセス文書の量を最小限に抑えることは可能だと思う。しかし、ユーザーや潜在的なシステム購入者に対してシステムを説明する製品文書を作成する必要性は依然としてある。この製品ドキュメンテーションは、印刷可能なユーザーマニュアルかもしれないし、システムの機能とその使用方法を説明するウェブベースのドキュメントかもしれない。

システムが分散したチームによって開発される場合、特にそれらが異なる国にある場合、定期的な短いミーティングのようなアジャイル手法で好まれる非公式なコミュニケーションの仕組みが効果的に機能しないことがあり、開発者のコミュニケーションを促進するために、より正式な文書による文書化が必要になることがある。請負業者と下請け業者との間の契約では、請負業者が作成する文書と、各下請け業者が作成する対応する

文書を指定することができる。

また、開発手法に関係なく、耐用年数の長いシステムにも文書化が必要な場合がある。このようなシステムにとって重要なドキュメンテーションは、必ずしもシステム設計に関する詳細な情報ではありません。むしろ、これらのシステムにおける重要な依存関係や、設計上の意思決 定 の根拠に関するドキュメンテーションです。したがって、他のシステムに依存している状況では、常にそのシステムと使用されている機能を文書化する必要があります。そうすれば、これらのシステムに変更が加えられて問題が発生した場合、その場所を特定し、うまくいけば迅速に修復することができる。

30.1　プロセス文書

効果的な管理には、管理されているプロセスが可視化されている必要がある。ソフトウェアは無形であり、ソフトウェアプロセスには、明らかに異なる物理的なタスクではなく、見かけ上似たような認知的タスクが含まれるため、この可視性を達成する唯一の方法は、プロセスの文書化を使用することである。

プロセス・ドキュメントは、いくつかのカテゴリーに分類される：

1. *計画、見積もり、スケジュール* 管理者が作成する文書で、ソフトウェアプロセスを予測し、管理するために使用される。
2. *報告書* 開発の過程で資源がどのように使われたかを報告する文書である。
3. *規格* プロセスの実施方法を定めた文書である。これらは、組織的、国家的、または国際的な基準から作成される。
4. *ワーキングペーパー* プロジェクトにおける主要な技術コミュニケーション文書であることが多い。プロジェクトに携わるエンジニアのアイデアや考えを記録し、製品ドキュメントの暫定版であり、実装戦略を記述し、特定された問題を提示する。暗黙のうちに、設計決定の根拠が記録されていることも多い。
5. *電子メール、Wiki など。* これらには、マネージャーと開発エンジニアの日常的なコミュニケーションの詳細が記録されている。

プロセス文書の大きな特徴は、そのほとんどが古くなってしまうことである。計画は、週単位、2週間単位、月単位で作成される。進捗状況は通常毎週報告される。メモには、必然的に変化する考え、アイデア、意図が記録される。

ソフトウェアの歴史家にとっては興味深いが、このようなプロセス情報の多くは、古くなった後はほとんど役に立たない。このため、アジャイル手法の提唱者は、プロジェクトでは、プロセス文書を含め、作成すべき文書の量を最小限に抑えるべきだと主張している。

外部顧客向けのプロジェクトではなく、内部プロジェクトの場合は、通常、定期的なミーティングを利用して作業の進捗状況をチームに報告し、文書ではなくディスカッションを通じて情報を共有することで、プロ

セス文書の量を根本的に削減することが可能である。しかし、プロジェクトが外部顧客向けに実施される場合、必要なプロセス文書の量は、以下によって決まる：

1. *顧客とサプライヤー間の契約上の取り決め* プロジェクト計画書など、いくつかのプロセス文書を作成し、顧客に提供しなければならないと契約書に明記されている場合がある。
2. *契約上のリスクに対する開発者の態度* プロセス文書には、そのプロセスで何が行われたかが記録される。

は、自分たちの仕事への取り組み方を弁護できるよう、プロセス文書を作成することを望んでいる。

3. *規制要件* システムが外部規制を受けており、使用前に認証を受けなければならない場合、規制当局は、システムの開発において適正な慣行が守られていることを証明するために、プロセス文書の作成を要求することがある。

新しい要求事項に応じてソフトウェアが進化する際に、特に有用なプロセス文書がある。例えば、テストスケジュールは、システム変更の検証を再計画するための基礎として機能するため、ソフトウェアの進化中に価値がある。もちろん、自動化されたテストは、システムが変更されたときに再実行できるため、さらに優れている。

デザイン決定の理由（デザインの根拠）を説明するワーキングペーパーも、デザインの選択肢や選択について論じるものとして価値がある。この情報にアクセスすることで、当初の決定と矛盾する変更を避けることができる。もちろん、理想的には、設計根拠はワーキングペーパーから抽出され、別途管理されるべきです。残念ながら、このようなことはほとんどありません。

30.2 製品ドキュメント

製品ドキュメンテーションは、納品されたソフトウェア製品を記述することに関係する。ほとんどのプロセス文書とは異なり、製品文書の寿命は比較的長い。製品ドキュメンテーションは、説明する製品と歩調を合わせて進化しなければならない。製品ドキュメントには、ユーザーにソフトウェア製品の使用方法を説明するユーザー・ドキュメントと、主にメンテナンス・エンジニアを対象としたシステム・ドキュメントがある。

30.2.1 ユーザーマニュアル

システムのユーザーは皆同じではありません。ドキュメンテーションの作成者は、さまざまなユーザーのタスクや、さまざまなレベルの専門知識や経験に対応できるように、ドキュメンテーションを構成しなければなりません。特に、エンドユーザーとシステム管理者を区別することが重要です

：

1. エンドユーザーは、何らかの作業を支援するためにソフトウェアを使用する。航空機の操縦、保険契約の管理、本の執筆などである。彼らは、そのソフトウェアがどのように役立つかを知りたがっています。彼らはコンピュータや管理の詳細には興味がありません。

2. システム管理者は、エンドユーザーが使用するソフトウェアを管理する責任がある。これは、システムが大規模なメインフレームシステムであればオペレーターとして、システムがワークステーションのネットワークを含むものであればネットワークマネージャーとして、エンドユーザーのソフトウェアの問題を解決し、ユーザーとソフトウェアサプライヤの間の連絡を取る技術的な第一人者として行動することを含むかもしれない。

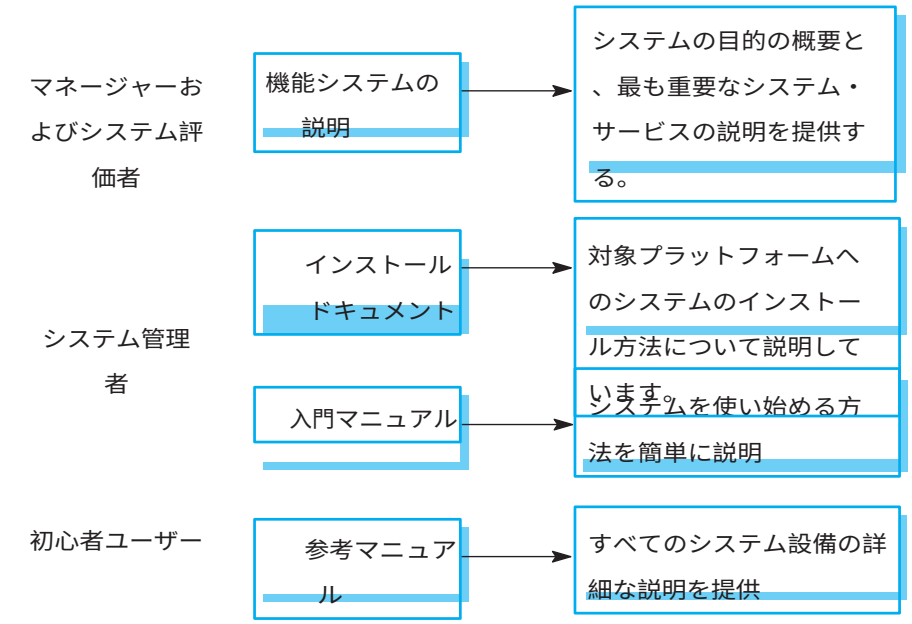

図30.1：文書の種類と文書利用者

このような異なるクラスのユーザーや、異なるレベルのユーザーの専門知識に対応するために、ソフトウェア・システムには複数の文書（あるいは1つの文書の章）を添付する必要がある（図30.1）。

システムの機能説明では、システム要件の概要を述べ、提供されるサービスを簡潔に説明する。この文書はシステムの概要を説明するものでなければならない。ユーザーは、入門マニュアルとともにこの文書を読み、そのシステムが自分にとって必要なものかどうかを判断できるようにする必要があります。

システムのインストールに関する文書は、システム管理者を対象としています。特定の環境にシステムをインストールする方法の詳細を提供する必要があります。システムに含まれるファイルと、必要最小限のハードウェア構成についての説明が含まれていなければなりません。確立しなければならない恒久的なファイル、システムの起動方法、特定のホストシステムに合わせてシステムを調整するために変更しなければならない設定依存ファイルについても記述する必要があります。PCソフトウェアの自動インストーラの使用により、この文書が不要と考えるサプライヤーもいる。実際、システム管理者がインストール に関する問題を発見し、修正するのに必要です。

入門マニュアルは、システムの「通常の」使い方を説明する、非公

式な入門書であるべきです。どのように使始め、どのようにエンドユーザーが般的なシステム機能を利用するか説明すべきである。また、例題をふんだんに盛り込むべきである。必然的に初心者は、その経歴や経験がどうであれ、間違いを犯すものです。このようなミスから回復し、有用な作業を再開する方法について簡単に発見できる情報は、この文書の不可欠な部分であるべきです。

システム・リファレンス・マニュアルは、システム設備とその使用法を説明し、エラーメッセージの完全なリストを提供し、検出されたエラーから回復する方法を説明すべきである。完全でなければならない。形式的な記述技法を用いてもよい。リファレンスマニュアルのスタイルは、以下のようなものであってはならない。

不必要に衒学的で冗長だが、完全性は読みやすさよりも重要だ。

指揮統制システムのようなある種のシステムには、より一般的なシステム管理者ガイドを提供すべきである。これには、システムが他のシステムと相互作用するときに生成されるメッセージと、これらのメッセージにどのように対応するかを説明すべきである。システムのハードウェアが関係している場合は、そのハードウェアを保守するオペレーターのタスクについても説明する。例えば、システムに新しいハードウェアを追加する方法や、ハードウェアの故障を診断する方法を説明することもある。

マニュアルだけでなく、他の使いやすい文書も提供されるかもしれない。利用可能なシステム設備とその使い方を列挙したクイック・リファレンス・カードは、経験豊富なシステム・ユーザーにとって特に便利である。これは、ユーザーの画面に常時表示できる短いオンライン文書として提供することができる。

ここ数年、ユーザーマニュアルを廃止し、オンラインヘルプシステムに置き換えることで、ドキュメンテーションのコストを削減する傾向がある。これらは、ユーザー情報をシステムの機能に関連した単純な塊に分解したものである。しかし残念ながら、これらのシステムには多くの問題がある：

1. ユーザーが特定の機能に興味を持ち、その機能の説明にすぐに行きたいと考えていることを想定しているという点で、「ブラウズ可能性」に欠けている。しかし、多くの場合、ユーザーは利用可能な一連の機能に興味があり、システムの一般的な概要を知りたいのである。これは、ユーザーマニュアルのような連続した文書から得ることができますが、リンクされた個別の情報の塊として実装されているシステムでは、はるかに困難です。

2. 問題指向ではなく、機能指向である。具体的な問題を抱えているユーザーが、その問題を解決するために使用する可能性のあるシステム機能を理解していなければ、その方法を見つけることはできない。

オンラインシステムはもちろん便利だが、よく書かれたユーザーマニュアルの代わりにはならない。

30.2.2 システム文書

システム文書には、要求仕様から最終的な受け入れテスト計画まで、システムそのものを記述するすべての文書が含まれる。プログラムを理解し維持するためには、システムの設計、実装、テストを記述した文書が不可欠です。ユーザー文書と同様に、システム文書も構造化されていることが重要であり、概要が読者をシステムの各側面のより正式で詳細な記述へと導く。

顧客の仕様に合わせて開発された大規模システムの場合、システム・ドキュメンテーションには以下を含めるべきである：

1. 要求文書と関連する根拠。
2. システム・アーキテクチャを説明した文書。
3. システム内の各プログラムについて、そのプログラムのアーキテクチャの説明。

4. システム内の各コンポーネントについて、その機能とインターフェースの説明。

5. プログラム・ソース・コードのリスト。コメントでコードの複雑な部分を説明し、使用したコーディング方法の根拠を示すべきである。意味のある名前が使われ、優れた構造化されたプログラミング・スタイルが使われていれば、コメントを追加しなくても、コードの多くは自己文書化できるはずである。このような情報は、現在では紙ではなく電子的に管理され、読者からの要求に応じて選択された情報が印刷されるのが普通である。

6. 各プログラムがどのように妥当性確認され、妥当性確認情報がどのように要求事項に関連するかを記述した妥当性確認文書。これらは、組織内の品質保証プロセスに要求されることがある。

7. システムメンテナンスガイドは、システムの既知の問題を説明し、システムのどの部分がハードウェアとソフトウェアに依存しているかを説明し、システムの進化が設計においてどのように考慮されているかを説明するものである。

システム・メンテナンスの一般的な問題は、システムが変更されたときに、すべての表現が確実に一致するようにすることである。これを支援するために、文書や文書の部分間の関係や依存関係を文書管理システムに記録する必要がある。

小規模なシステムや、ソフトウェア製品として開発されたシステムの場合、システム・ドキュメンテーションは通常、あまり包括的でない。開発者に対するスケジュールのプレッシャーは、単に文書が書かれなかったり、書かれたとしても最新の状態に保たれていなかったりすることを意味する。このようなプレッシャーは時に避けられないが、私の考えでは、少なくともシステムの仕様書、アーキテクチャ設計書、プログラムのソースコードは常に維持するように努めるべきである。

残念ながら、ドキュメンテーションのメンテナンスは軽視されがちである。ドキュメンテーションは、関連するソフトウェアと歩調を合わせることができなくなり、システムのユーザーと保守者の双方に問題を引き起こす可能性がある。他のドキュメントを後で修正するつもりでコードを修正することで、期限を守ろうとするのは自然な傾向である。

多くの場合、仕事のプレッシャーから、変更すべき箇所を見つけることが非常に困難になるまで、この修正は脇に置かれ続ける。この問題に対する最良の解決策は、文書の関係を記録し、ある文書の変更が別の文書に影響したときにソフトウェア・エンジニアに注意を促し、文書に矛盾がある可能性を記録するソフトウェア・ツールを使って、文書のメンテナンスをサポートすることである。このようなシステムは、Garg and Scacchi (1990)によって説明されている。

30.3　ドキュメントの品質

ドキュメントの質は、プログラムの質と同じくらい重要である。システムの使い方や理解方法に関する情報がなければ、そのシステムの有用性は低下する。

コンポーネント	説明
識別データ	肩書きや識別子などのデータで、以下を一意に識別するもの。 文書
目次	章／節名とページ番号
図版リスト	図番号とタイトル
はじめに	文書の目的と内容の簡単な要約を示す。
ドキュメントの使用に関する情報	文書を効果的に使用する方法について、さまざまな読者への提案。
オペレーションの概念	を使用する概念的な背景についての説明。 ソフトウェアである。
手順	ソフトウェアがサポートするように設計されているタスクを完了するためにソフトウェアを使用する方法に関する指示。
インフォメーション　ソフトウェアの　ソフトウェアコマンド	ソフトウェアがサポートする各コマンドの説明。
エラーメッセージと問題解決	報告される可能性のあるエラーと、これらのエラーから回復する方法についての説明。
用語集	専門用語の定義
関連情報源	追加情報を提供する他の文書への参照またはリンク
ナビゲーション機能	読者が現在地を検索できる機能 とドキュメント内を移動する。
索引	主要な用語と、その用語が参照されているページのリスト。
検索機能	電子文書において、文書内の特定の用語を検索する方法。

図30.2 ソフトウェア・ユーザー・ドキュメントの推奨構成要素

文書品質を達成するには、文書設計、標準、品質保証プロセスに対する管理者のコミットメントが必要である。多くのソフトウェア・エンジニアは、良質なドキュメントを作成することは、良質なプログラムを作成することよりも難しいと感じている。

残念なことに、多くのコンピューター・システム・ドキュメンテーションは、文章が下手で、理解しにくく、時代遅れであったり、不完全であったりする。状況は改善されつつあるが、多くの組織ではシステム・ドキュメントの作成に十分な注意を払っていない。

30.3.1　文書構造

文書構造とは、文書内の資料を章立てにし、その章立ての中でセクションやサブセクションを構成する方法のことです。文書構造は、読みやすさや使いやすさに大きな影響を与えるため、文書を作成する際には慎重に設計することが重要です。ソフトウェアシステムと同様、各部分ができるだけ独立するように文書構造を設計する必要があります。こうすることで、各部分を1つの項目として読むことができ、変更を加えなければならないときに相互参照の問題を減らすことができます。

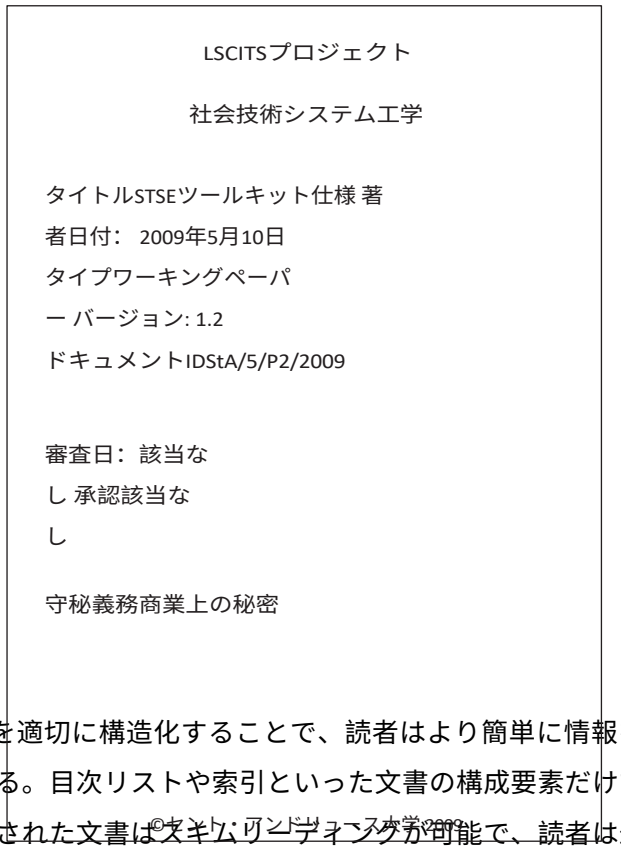

図30.3 文書表紙の例

文書を適切に構造化することで、読者はより簡単に情報を見つけることができる。目次リストや索引といった文書の構成要素だけでなく、適切に構造化された文書はスキムリーディングが可能で、読者は最も関心のあるセクションやサブセクションを素早く見つけることができる。

IEEE standard for user documentation (IEEE, 2001)は、文書の構造として、図30.2に示す構成要素を含むことを提案している。この規格は、これらが文書の望ましい、あるいは不可欠な機能であることを明確にしているが、これらの構成要素をどのように提供するかは、文書の設計者に依存することを明確にしている。いくつかの構成要素（目次など）は明確に独立したセクションであり、ナビゲーション機能などの他の構成要素は文書全体に存在する。

次節で述べるように、このIEEE規格は一般的な規格であり、この規格の使用を義務付けるのであれば、これらの構成要素をすべて含めなければならない。しかし、多くの組織は、この規格をガイドとして使用し、必ずしも図30.2に示すすべての構成要素を含めるとは限らない。そのような状況では、常に従うべきと私が考える最低限の構造化ガイドラインがある：

1. どんなに短い文書であっても、すべての文書には、プロジェクト、文書、作成者、作成日、文書の種類、構成管理および品質保証情報

、文書の意図される受信者、文書の機密性クラスを特定する表紙を付けるべきである。また、文書を検索するための情報 (

抄録またはキーワード）と著作権表示。図30.3は、考えられる表紙のフォーマットの例である。

2. 数ページ以上の文書は章立てにし、各章をセクションとサブセクションに分ける。これらの章、節、小節を列挙した目次ページを作成すべきである。チャプター、セクション、サブセクションには一貫した番号付けを行い、チャプターには個別にページ番号を付ける（ページ番号は*chapter-page*とする）。これにより、文書全体を再印刷することなく、個々の章を差し替えることができるため、文書の変更が簡単になる。

3. 文書に多くの詳細な参考情報が含まれている場合は、索引をつけるべきである。包括的な索引があれば、情報を簡単に発見することができ、書き損じの文書を使えるようにすることができる。索引がなければ、参考文献は事実上役に立たない。

4. ボキャブラリーの異なる幅広い読者を対象とする文書の場合、文書内で使用される専門用語や略語を定義した用語集を提供すべきである。

文書構造は多くの場合、事前に定義され、文書標準に定められている。これには一貫性という利点があるが、問題を引き起こすこともある。標準がすべての場合に適切とは限らないし、軽率に標準を押し付けると不自然な構造を使わざるを得なくなることもある。

30.3.2 文体

優れた文書を作成するためには、標準と品質評価が不可欠であるが、文書の品質は、基本的に、明確で簡潔な技術的文章を作成する ライターの能力に依存している。つまり、優れた文書には優れた文章が必要なのです。

文書を上手に書くのは簡単なことではないし、一段階のプロセスでもない。書いたものを読み、批判し、納得のいく文書ができるまで書き直さなければならない。テクニカル・ライティングは科学というよりむしろ技術であるが、上手に書くための大まかな指針は以下の通りである：

1. *受動態ではなく能動態を使う*「画面の左上に点滅するカーソルが表示されるはずです」ではなく、「画面の左上に点滅するカーソルが表示されるはずです」と言う方がよい。

2. *文法的に正しい構文と正しいスペルを使う* 不定詞を大胆に分割したり（このように）、単語のスペルを間違えたり（mispellのように）することは、多くの読者を苛立たせ、彼らの目から見た書き手の信頼性を低下させる。残念なことに、英語のスペルは統一されておらず、英米の読者は時として別のスペルを嫌う非合理的なところがある。

3. *複数の異なる事実を提示するような長い文章は使わないでください*。それぞれの文章は、次のような形で吸収することができる。

それ自身である。読者は、完全な文章を理解するために、いくつかの情報を一度に維持する必要はない。

4. *段落を短くする* 原則として、7文以上の段落は作らない。私たちが即時的な情報を保持できる容量は限られている。短い段落であれば、その段落に含まれるすべての概念を短期記憶に保持することができる。

5. *冗長であってはならない* 少ない言葉で何かを語れるなら、そうすること。長い説明が必ずしも深いとは限らない。量より質が大切だ。

6. *使用する用語は正確に定義すること* コンピューティング用語は流動的で、多くの用語は複数の意味を持つ。モジュールや*プロセスといった*用語を使用する場合は、その定義が明確であることを確認してください。用語集に定義をまとめましょう。

7. *説明が複雑な場合は、繰り返し説明する* 同じ事柄について、2つ以上の異なる言い回しによる説明を提示するのは、良いアイデアであることが多い。読者が1つの説明を完全に理解できなかった場合、同じことを別の言い回しで言ってもらうと、読者は助かるかもしれない。

8. *見出しと小見出しを活用する*章を、別々に読めるように分割する。常に一貫した番号の付け方をする。

9. *可能な限り事実を箇条書きにする* 事実は通常、文章で示すよりもリストで示す方が明確である。強調のために、テキストのハイライト（斜体や下線）を使用する。

10. *参照番号だけで情報を参照しない* 参照番号を示し、その参照番号が何を扱っているかを読者に知らせる。例えば、「セクション1.3では.....」と言うのではなく、「セクション1.3では、マネジメントプロセスモデルについて.....」と言うべきです。

文書もプログラムと同じように検査されるべきである。文書検査では、テキストが批判され、脱落が指摘され、文書の改善方法が提案される。この後者の点で、エラー修正メカニズムではなくエラー発見メカニズムであるコード検査とは異なる。

個人的な批評だけでなく、ワープロに組み込まれている文法チェッ

カーを使うこともできる。このようなチェッカーは、文法に則っていない、あるいは使い方が乱雑な単語を見つけ出します。長い文章や段落、能動態ではなく受動態の使い方などもチェックできます。これらのチェッカーは完璧ではなく、時代遅れのスタイル・ルールやある国特有のルールを使っていることもある。とはいえ、入力中にスタイルをチェックすることが多いため、改善すべきフレーズを特定するのに役立ちます。

30.3.3　ドキュメンテーション基準

文書規格は、文書の品質保証の基礎として機能する。適切な標準に従って作成された文書は、一貫した外観、構造、品質を持つ。ユーザー・ドキュメンテーションに関するIEEE規格については、すでに前節で紹介したので、この規格については、もう少し詳しく説明する。しかし、関連するのは文書化に焦点を当てた規格だけではない。ドキュメンテーション・プロセスで使用される可能性のある他の規格は、以下の通りである：

1. *プロセス標準* 高品質な文書作成のために従うべきプロセスを定義した標準。
2. *製品規格* 文書自体の構成や構造などを規定する規格。
3. *交換標準* 現在、事実上すべての文書は電子形式で保存されている。しかし、これらは異なるシステムを用いて異なる時期に開発される可能性があるため、文書のすべての電子コピーに互換性を持たせるためには、交換標準が必要となる。

規格はその性質上、あらゆるケースをカバーするように設計されているため、時として不必要に制限的に見えることがある。したがって、プロジェクトごとに適切な標準を選択し、そのプロジェクトに適するように修正することが重要である。比較的耐用年数の短いシステムを開発する小規模なプロジェクトと、10 年以上にわたってソフトウェアを維持しなければならない大規模なソフトウェアプロジェクトとでは、必要とされる標準が異なる。

プロセス基準

プロセス標準は、文書を作成する際のアプローチを定義するものである。これは一般に、文書作成に使用すべきソフトウェアツールを定義し、高品質の文書が作成されることを保証する品質保証手順を定義することを意味する。

文書プロセスの品質保証基準は柔軟でなければならず、あらゆる種類の文書に対応できなければならない。文書が単なる作業文書やメモである場合には、明示的な品質チェックは必要ない場合もある。しかし、文書が正式な文書である場合、すなわち、構成管理手順によって文書の進化が

管理される場合には、正式な品質プロセスを採用すべきである。図3は、可能なプロセスの一つを示している。

起草、チェック、改訂、再起草は、許容できる品質の文書ができあがるまで続けられる反復プロセスである。許容できる品質レベルは、文書の種類と、その文書の潜在的な読者によって異なります。

製品規格

製品標準は、ソフトウェア開発の過程で作成されるすべての文書に適用される。文書は一貫した外観を持ち、同じクラスの文書は一貫した構造を持つべきである。文書標準はプロジェクト固有であるが、より一般的な組織標準に基づくべきである。

開発されるべき製品規格の例としては、以下のようなものがある：

1. *文書の識別基準* 大規模プロジェクトでは通常数千の文書が作成されるため、各文書は一意に識別されなければならない。正式な文書の場合、この識別子は、コンフィギュレーションマネージャーが定義した正式な識別子とすることができる。非公式文書の場合は、プロジェクトマネージャが文書識別子のスタイルを定義する。

2. *文書構造標準* 前のセクションで説明したように、ソフトウェアプロジェクトで作成される文書の各クラスには、適切な構造がある。構造標準は、この構成を定義すべきである。また、ページ番号、ページのヘッダーとフッター情報、セクションとサブセクションの番号に使用される規約も規定する必要があります。

3. *文書体裁基準* 文書体裁基準は、文書の「ハウススタイル」を定義し、文書の一貫性に大きく寄与する。これには、文書で使用するフォントやスタイルの定義、ロゴや会社名の使用、文書構造を強調するための色の使用などが含まれる。

4. *文書の更新基準* システムの変更を反映するために文書が変更される場合、これらの変更を示す一貫した方法を使用すべきである。これには、新しい文書のバージョンを示すために異なる色の表紙を使ったり、変更または削除された段落を示すために変更バーを使ったりすることが含まれる。これらはほとんどのワープロシステムで自動的に追加することができる。文書へのアクセスを提供するウェブサイトは、すべてのバージョンにリンクし、どれが文書の最新バージョンであるかを明確に示すべきである。

文書標準は、すべてのプロジェクト文書とユーザー文書の初稿に適用される べきである。しかし、多くの場合、ユーザー・ドキュメントは、プロジェクトではなく、ユーザーに適した形で提示されなければならない

。

インターチェンジ規格

文書交換標準は、より多くの文書が紙だけでなく、あるいは紙の代わりに電子フォーマットで作成されるようになるにつれて重要になっている。文書を交換する際によく発生する問題は以下の通りである：

1. 異なる国で開発された文書は、異なる文字セットを使用することがある。
2. ワープロソフトのバージョンが異なると互換性がなく、新しい文書が旧バージョンのソフトで読めないことがある。機能の実装にわずかな違いがあり、その結果、既存の文書のレイアウトが変わってしまうことがあります。
3. 組織が合併した場合、それぞれの文書交換標準に互換性がなくなる可能性がある。
4. 補足プログラムやアドイン（書誌マネージャなど）は、組織全体で使用することはできません。

ソフトウェア・システムとともに提供される文書には、通常、Adobe Portable Document Format（PDF）が使用される。しかし、開発チームによって文書が交換され、ドラフトが組織内で回覧される場合、これらは多くの場合、使用されているワープロ（多くの場合Microsoft Wordだが、Open Office、Google Docs、Latexの場合もある）のフォーマットで作成される。

標準的なワープロとグラフィカルな編集システムの使用がプロセス標準で義務付けられていると仮定すると、交換標準はこれらのツールを使用するための規約を定義する。交換標準を使用することで、文書を電子的に転送し、元の形式で再作成することができる。

交換標準は、単に文書作成に共通のバージョンのシステムを使用するという合意以上のものである。交換標準の例としては、文書作成にテキストフォーマットシステムを使用する場合、合意された標準マクロセットを使用することや、ワードプロセッサーの標準スタイルシートを使用することなどがある。また、プリンターやディスプレイの機能が異なるため、交換標準によって使用するフォントやテキストスタイルが制限されることもある。

30.3.4 ユーザー・ドキュメントのIEEE標準

ユーザー・ドキュメンテーションに関する最初のIEEE規格（IEEE, 1987）は1987年に作成され、この原稿を書いている時点では、この規格の新しいドラフトが出版に向けて準備中である（IEEE, 2001）。

すべての標準がそうであるように、この標準もまた、ソフトウェア

文書に関す る知恵と経験を集約し、ユーザー文書の構造を提案している。この構造を基礎として、この標準はソフトウェアのユーザードキュメントの内容を議論し、これらのドキュメントのフォーマット標準を提案する。規格の最新版で提案されている文書構造についてはすでに述べた。この規格の書式に関するアドバイスを説明するために、現行のグッドプラクティス規格の草案からいくつか引用する：

> *文書は、視覚、聴覚、その他の身体的制約のある人々も利用できるような媒体や形式で提供されるべきである。*
>
> *電子文書の印刷方法については、電子文書と印刷文書の両方に記 載する。*

色の区別ができないユーザーもいるため、文書では赤や緑といった色だけで意味を伝えるのではなく、テキストによる合図を提供すべきである。

警告、注意、注記は、通常のテキストや指示ステップと容易に区別できる一貫した形式で表示しなければならない。

ユーザが入力するコマンドやコードの文書形式は、リテラル（示されたとおりに入力する）と変数（ユーザが選択する）を明確に区別しなければならない。

テキストに付随するイラストは、テキスト中の最初の参照箇所に隣接して表示し、関連するテキストとイラストを同時に見ることができるようにする。

これらから、この規格が規定的でなく有益であり、したがって異なる企業や組織で使用される異なる慣習に対応できることがわかる。

すべての規格がそうであるように、この文書規格も、それが使用される現地の状況に適合させなければならない。そのため、この規格の助言を現地の状況に合わせてインスタンス化し、使用すべき具体的な構造や書式を定義しなければならない。

30.4　ドキュメント制作

文書作成とは、文書を作成し、出版用にフォーマットするプロセスである。図30.4は、文書作成プロセスを次のように分けて示している。
文書作成、推敲、制作の3段階。最近のワープロ・システムは、このプロセスのすべての部分をサポートするソフトウェア・ツールの統合パッケージになっている。しかし、最高品質の文書を作成するためには、ワープロに内蔵されている機能ではなく、いくつかの準備プロセスには別のツールを使うのが最善であることに変わりはない。準備の3つの段階と、関連するサポート機能は以下の通り：

1. *文書作成* 文書の情報を最初に入力すること。ワードプロセッサやテキストフォーマッタ、表や数式プロセッサ、ドローイングやアートパッケージでサポートされている。

2. *文書の推敲 文書の*書き方や見せ方を改善し、より理解しやすく読みやすい文書にする作業です。スペルミス、句読点ミス、文法ミスの発見と削除、不明瞭な言い回しの検出、文章内の冗長な部分の削除などが含まれます。このプロセスは、オンライン辞書、スペルチェッカー、文法・スタイルチェッカー、スタイルチェッカーなどのツールによってサポートされる。

3. *ドキュメント制作*プロフェッショナルな印刷のためにドキュメントを準備するプロセスである。デスクトップパブリッシングパッケージ、アートワークパッケージ、タイプスタイリングプログラムによってサポートされる。

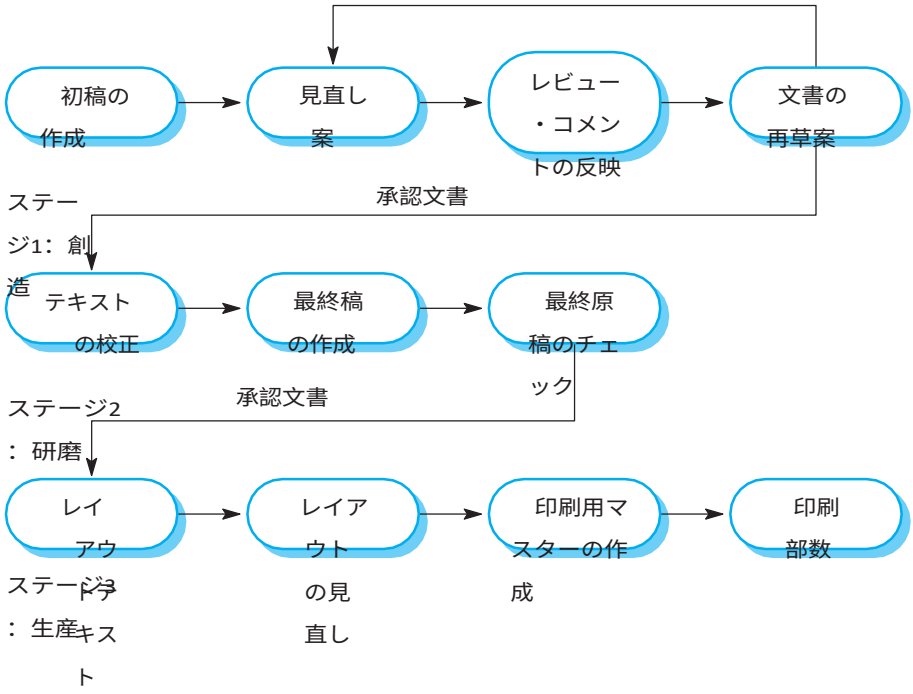

図30.4 文書作成

文書作成プロセスをサポートするこれらのツールだけでなく、構成管理システム、情報検索システム、ハイパーテキストシステムも、文書の保守、検索、管理をサポートするために使用されることがある。

最近のワープロ・システムは画面ベースで、テキスト編集と書式設定が組み合わされている。ユーザーの端末に表示される文書のイメージは、多かれ少なかれ、印刷された文書の最終形と同じである。完成したレイアウトはすぐにわかる。文書を印刷する前にエラーを修正し、レイアウトを改善することができる。しかし、すでにプログラムの準備にエディターを使っているプログラマーは、別のエディターやテキスト書式設定システムを使うことを好む場合がある。

Latexのようなテキスト整形システムは、文書作成者が指定したレイアウトプログラムを解釈する。レイアウトコマンド（多くの場合、標準的で定義可能なコマンドセットから選択される）は、文書のテキストに挟み込まれる。テキストフォーマッタは、これらのコマンドと関連するテキストを処理し、プログラマの指示に従って文書をレイアウトします。

テキストフォーマットシステムは、レイアウトされるテキストを先読みすることができるので、作業コンテキストがより限定されるワープロシステムよりも優れたレイアウト決定を行うことができる。また、Latexは数学や数式の扱いにおいてもワープロよりはるかに優れている。コマンドはまさにプログラミング言語であるため、プログラマーはワープロよりも

Latexを好むことが多いが、技術者以外のユーザーは通常、Latexの方が使いにくいと感じる。

テキスト・プロセッサーの主な欠点は、プログラミングをマスターしても、その出力がすぐに表示されないことである。

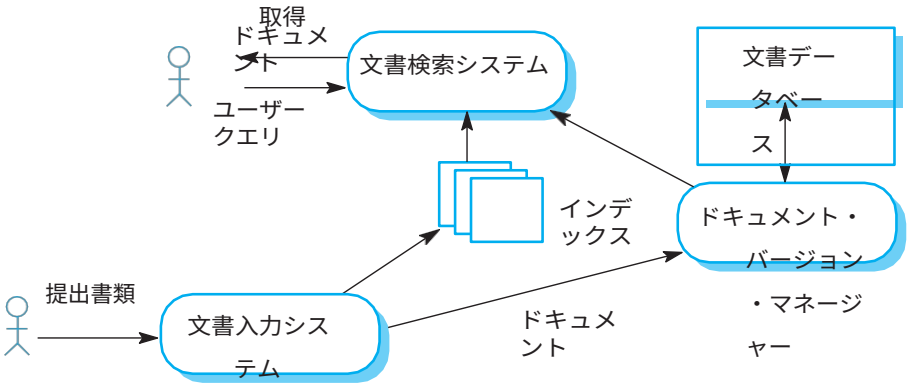

図30.5 文書管理システム

を生成する。ユーザーはテキストを処理し、プレビュー・パッケージを使って出力を表示しなければならない。フォーマットやレイアウトのエラーが発見された場合、すぐに修正することはできません。元のソースを修正し、プレビュープロセスを繰り返さなければならない。このように、テキストフォーマッタはより質の高い文書を作成できるものの、ほとんどのユーザーはワープロよりも不便だと感じている。

文書制作の最終段階は熟練を要する作業であり、印刷部数の多い文書については、プロの印刷業者に任せるべきである。しかし、Quark Xpress のようなデスクトップ・パブリッシング（DTP）システムや、写真やアートワークのスキャン・加工をサポートするグラフィック・システム（Photoshop）は、現在、普遍的に使用されている。DTPシステムは、テキストやグラフィックのレイアウトを部分的に自動化する。DTPシステムでは、文書のレイアウトや外観を非常に細かく制御できるため、エンジニアが完成したシステム文書を作成する際に使用できる。

パブリッシングシステムを使用する利点は、作成プロセスの一部の工程が省略されるため、高品質のドキュメントを作成するためのコストが削減されることです。少部数のドキュメントでも、高い水準で作成できます。デスクトップパブリッシングシステムを使用するデメリットは、グラフィックデザイナーのスキルを自動化できないことです。その魅惑的な使いやすさは、熟練していないユーザーが魅力的でない、ひどいデザインの文書を作成する可能性があることを意味します。

プロジェクトの過程では膨大な数の文書が作成され、必要なときに正しいバージョンの文書が利用できるように管理する必要があります。プロジェクトが分散している場合、文書のコピーはさまざまな場所で作成さ

れ、保管されます。各文書の確定版を含む文書の「マスターファイル」を管理することは非常に重要です。これは、文書の利用者が文書の最新バージョンで作業していないためにミスを犯すという、非常に一般的な問題を最小限に抑えるのに役立ちます。

各文書は一意なレコードを持つべきで、これは文書データベースのレコードのキーとして使うことができる。しかし、タイトルや著者などの他のフィールドによる検索もサポートされるべきである。

文書を保存するファイルシステムと、文書情報を管理するデータベース管理システムを使って文書を管理する場合の基本的な問題は、ユーザーがシステムの使い方を律しなければならないということである。ユーザーは

コンピュータ上のローカル・コピーや印刷したコピーを使用するのではなく、必要な都度、システムから文書のコピーをチェックアウトするようにする。実際には、このレベルの規律を達成することは困難であり、常にエラーが発生する可能性がある。

航空機用ソフトウェアのような非常に大規模なプロジェクトでは、文書の保管と文書情報の保守を統合した専用の文書管理システムを使用することができる（図30.5）。文書管理ソフトウェアでは、関連文書をリンクし、誰が文書をチェックアウトしたかの記録を管理し、文書テキストの圧縮と非圧縮をサポートし、文書を検索できるように索引付けと情報検索機能を提供することができる。また、文書管理システムにはバージョン管理機能があり、異なる文書のバージョンを維持することができる。

30.4.1 オンライン・ドキュメント

システムとともに提供されるオンライン・ドキュメントは、インタラクティブな ヘルプ・システムを通じてソフトウェアに関する非常に限定的な情報を提供 する単純な「read me」ファイルから、ユーザー・マニュアルやチュートリアルな どを含む完全なウェブベースのシステム・ドキュメント一式まで、さまざまなものが あります。しかし、ほとんどのアプリケーションでは、ハイパーテキストベースのヘルプシステムが、最も一般的に提供されるオンラインドキュメントです。これは、アプリケーションの一部として提供される組み込みのヘルプシステムであることもあります。しかし、より一般的には、ヘルプシステムはウェブベースのもので、ユーザーが情報にアクセスするにはインターネット接続が必要です。

オンライン・ドキュメンテーションの主な利点は、もちろんそのアクセシビリティである。ユーザーがマニュアルを探す必要はなく、古くなったドキュメントを拾い読みする可能性もなく、検索機能を使って素早く情報を見つけることができる。原則的に、ビルトインヘルプシステムは「コンテキスト認識」を持つことができ、どのようなシステム機能が使われているかを知ることができる。これは、現在のユーザー活動をサポートするヘルプを提供できることを意味する。ウェブベースのヘルプシステムは、潜在的なシステム購入者にシステムの機能に関する情報を提供するために使用することができる。

しかし、ビルトインやウェブベースのヘルプシステムにはいくつか

の欠点がある。以下はその例である：

1. ブラウズしやすさ」に欠けるため、読者は必要な情報を見つけるために簡単に目を通すことができない。私たちはしばしば、文書から欲しい情報を見つけ出すこ と は で き て も 、それを特徴づけることは難しいと感じる。ブラウジングは、このように私たちが検索する際に使用する重要なメカニズムである。ブラウジングはまた、未知のシステム設備を偶然発見する機会も提供してくれる。
2. スクリーンは紙よりも解像度が高いため、紙よりもスクリーンで文書を読む方が難しく、疲れる。
3. ユーザーがウェブベースのヘルプシステムで迷子になるのは非常に簡単で、その結果、目的の場所にナビゲートするのが難しくなる。

この原稿を書いている時点では、iPadのようなタブレット端末は導入されたばかりで、オンライン・ドキュメントへのアクセス方法にはまだ影響を与えていない。タブレットが導入されれば、紙文書とオンライン文書のバランスが変わり、紙文書が使われなくなる可能性はある。

画面ベースの文書を設計する際には、常にこれらの問題を念頭に置く必要があります。その結果、画面ベースの文書も紙ベースの文書もよく書かれたものであるべきですが、電子文書と紙文書では異なるデザインが必要になります。画面と紙の違いから、単にワープロ文書をウェブページのセットに変換しても、高品質のオンライン文書が作成されることはほとんどありません。

オンライン・ドキュメンテーションは、システム・ユーザーにヘルプとサポートを提供する、より一般的なユーザー・サポート・システムの一部である。これらについては、Dix *et al.*(2004) と Shneiderman and Plaisant(2004)が詳しく論じている。

キーポイント

ソフトウェア・ドキュメンテーションは、ユーザーやシステムの保守を担当するソフトウェア・エンジニアにシステムを説明するために使用される。

ユーザーのタイプに応じて、異なる詳細レベルの文書を作成すべきである。

アジャイルメソッドでは、ドキュメンテーションは古くなりがちで、ほとんど使われないため、排除しようとする。しかし、ほとんどすべてのシステムにおいて、ある程度のユーザードキュメントは必要であり、システム要件とアーキテクチャに関するドキュメントは、システムのメンテナンスに役立つ。

文書が有用であるためには、文書の質が非常に重要である。文書はよく構造化され、単純明快な言葉で書かれるべきである。

文書標準は、文書がどのように作成されるべきかを定義するプロセス標準、文書組織を定義する製品標準、または文書交換標準であるかもしれない。

文書管理システムは、文書が維持され、容易にアクセスできることを保証するために、非常に大規模なプロジェクトで使用されること

がある。

さらに読む

*The Art of Technical Documentation 2nd ed.*この本は、主にテクニカルライターとテクニカルライター志望者を対象としており、ソフトウェア文書に特化したものではない。しかし、ドキュメントのプレゼンテーションやスタイルに関する一般的なアドバイスも含まれています。(K. Haramundanis, Woburn-MA: Butterworth-Heinemann, 1998)

科学と工学のためのライティング：論文、プレゼンテーション、レポート。これも文書作成に関する一般的な本である。Haramundanisの本よりも詳細で、構成やスタイルに関する非常に具体的なアドバイスやチェックリストが含まれている。(H.シリン-ロバーツ. ウォバーン-マサチューセッツ州: バターワース-ハイネマン, 2000)

*リード・ミー・ファーストコンピュータ業界のためのスタイルガイド。*この本には、ソフトウェアとハードウェアの文書のスタイルに関する具体的な情報とアドバイスが含まれています。
国際的な読者のために書くことについての情報を含む。もともとは、Sunのソフトウェアやハードウェア・システムの文書作成に携わる人々のために開発されました。(3rd 版, Sun Technical Publications, 2010)

*ウェブのためのライティング*ニールセンは著名なユーザー・インターフェース・デザイン・コンサルタントであり、オンライン・ドキュメンテーションにも一般的に関連する、ウェブのためのライティングに関する多くのアドバイスを含む包括的なサイトをまとめている。(J. Neilsen.
http://www.useit.com/papers/webwriting/)

参考文献

Dix, A., Finley, J., Abowd, G. and Beale, R. 2004.*Human-Computer Interaction, 3rd ed.*London：Prentice-Hall.

Garg, P. K. and Scacchi, W. 1990.ソフトウェアのライフサイクル文書を管理するためのハイパーテキストシステム」。*IEEE Software*, **7** (3), 90-8.

IEEE, 1987.*IEEE Standard for Software User Documentation*, IEEE-Std1063- 1987. ニューヨーク：ニューヨーク: Institute of Electrical and Electronics Engineers.

IEEE, 2001. *ソフトウェア・ユーザー・ドキュメンテーションの標準草案*, IEEE- Std1063/D5.1.2001.ニューヨーク：ニューヨーク: Institute of Electrical and Electronics Engineers.

Shneiderman, B. and Plaisant, C. 2004.*ユーザーインターフェースのデザイン*. ボストン：Addison Wesley.

エクササイズ

30.1 システム要件とアーキテクチャに関する文書化がシステム保守にとって重要である理由を説明できる。

30.2 あなたがよく使うシステムのドキュメントを見てください。それが(a)初心者ユーザーと(b)経験豊富なユーザーにとってどの程度役に立つか、簡単に評価してください。

30.3 インドとカ ナ ダ に メンバーがいるチームがシステムを開発する場合、ドキュメントのやり取りでどのような問題が発生するでしょうか。

30.4 ここで述べた明確な文章を書くためのアドバイスを参考に、あなたが使っているいくつかの文書の文章の質を評価しましょう。これらの文書の一部をわかりやすく書き直しなさい。

30.5 非常に大規模なプロジェクトで文書管理システムを使用する必要がある理由を説明する。