

自動設計・現寸システム開発の生産性向上のための手法

日本橋梁(株) 正会員 ○松田 正弘

〃 池田 秀夫

〃 羽田野 英明

まえがき

エネルギーを始めとする限られた物的、人的資源をいかに効率的に利用し、生産性を向上させていくかは、低成長時代の今日に対応していくために、どこでも真剣に取り組まれている重要な課題である。現在、どの業種の企業においても、自動化に対するニーズは依然として高く、そのために効果ある自動化システムが模索されている状況にある。また、製造業においては設計、生産の分野で自動化システムの構築が急務となってきている。しかし、こうした中で、開発要員の不足、コスト増などの問題があって、必ずしも要請に応じきれていないのが現状である。ところで、電算機の著しい技術の進歩に伴い、ハードウェアは低価格化してきたが、プログラムの開発費用は人件費の上昇に伴って、ハードウェア費用を大きく上まわってきており、この傾向は今後も続くものと予想される。特に、自動化システムの適用領域が拡大するにつれて、システムはますます大規模、複雑化している。それに伴い、システム開発に要する作業はもちろんのこと、保守作業の増加が大きくなっている。

したがって、このシステム開発と保守の生産性向上および効率化を図ることが一層重要な課題となってきた。

そこで、本文では、この課題に対処する目的で、システム開発におけるシステム設計、プログラミング、メンテナンスまでの一連の開発作業およびドキュメント作成体系の標準化を図ることを試みている。その適用例を橋梁の自動設計システムと自動現寸システムの開発について示すとともに、それによって得られた適用効果を紹介したいと思う。

1. プログラム開発上の問題点

製造業においては設計、生産への積極的な電算化への試みが進められており、適用領域の拡大への需要は今後、ますます増大することが予想される。と

ころで、プログラム開発の拡大と相まって、プログラムの保守や変換の量が増えることになり、新規開発業務に振り向けられるマンパワーは年とともに減少している。このため、プログラム開発の大幅なコストダウンを計らなければ開発需要に対応できなくなっている状況にある。我々が従来より行ってきたプログラム開発上の問題点として、次のような点が挙げられる。

- 1) 生産性が低い → プログラム開発を短期間で行えないか。
- 2) 保守性に欠ける → メンテナンスが容易に行えないか。

その他、開発からデバッグの進捗状況の適確な把握が困難などである。

これらの問題点は、次のような状況に起因しているものと考えられる。すなわち、プログラム開発作業における計画フェイズでシステム要件が明確に設定されていないため、出来上りソフトとのくい違い、あるいは開発途中での機能追加などの手戻り作業が出る。また、ドキュメントの統一的な表記方法が確立されていないため、プログラムの機能仕様の不備を生じやすいことなどである。

2. 解決方法

これらの問題点に対して、システム開発における生産性向上という観点から、指向したのがつぎのような方法である。

システム開発の効率化には、システムの分析計画→設計→作成→保守という段階を通して生産性・保守性を向上させる必要があり、したがって、我々はつぎのような開発作業の標準化を図ることを試みた。

- 1) システム要件の分析からテストまでの一連の作業内容の標準化を行い、開発作業をやり易くする。
- 2) ドキュメントの標準化を行い、メンテナンスおよび改善作業の効率化を図る。

システム開発における生産性向上のためには、標準化が不可欠であると考えられ、さらにユーザーの

多種多様な要求に対して容易に追加・修正が可能なように、メンテナンスおよび改善作業の効率化を目指したものである。

3. 適用例

開発業務の効率向上を目的とする標準化技法を橋梁の自動設計システムおよび自動現寸システムの開発において適用した例を示したい。

橋梁の自動設計システムは I 型桁の線形計算から応力解析、各部細部設計まで自動的に設計するシステムであり、また、自動現寸システムは I 型桁の設計段階の情報に製作時や載荷時のそりやたわみなどを付加して、所要寸法を展開計算し、製作に必要な定規、型板などを自動的に作り出すものである。

これらのシステムの開発に先立って、図-1に示すような開発工程を設定し、これにもとづいて各段階の開発工程を明確に区別して進めていった。そして、各工程の区切りでレビュー作業を設けて、要件あるいは仕様との整合性チェックを行った。

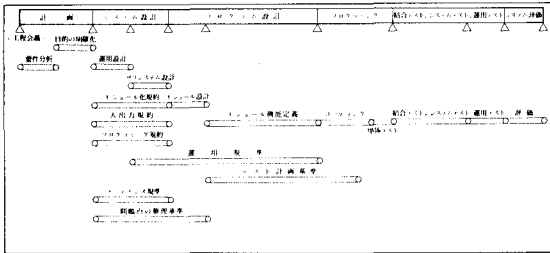


図-1. 開発工程

つぎに、図-2は自動設計および自動現寸システムの開発において、適用して記述したシステム設計書の一部分を示すものである。

従来のシステム仕様は、文章形式で記述されるのが通例であったが、ここで我々が使用している方法は、使い易さと実用性を重視して図形化記述方式を採り、機能表現を明確にし、機能概要の説明をパターン化し、それを用いて標準化している。

システム設計書類に不可欠な要素として、各機能の正確かつ詳細な記述と、もう一つはそれぞれの機能の有機的関連を示す全体の構造図を併せもつこととされている。

システム設計書の構成は、次のようになっている。

- (1) サブシステム関連図
- (2) サブシステム説明書
- (3) 問題解析書

- (4) ファイル設計書
- (5) 変数説明書
- (6) モジュール構造図
- (7) モジュール機能説明書

自動設計システムにおいては、各サブシステムとして設計 FILE 管理、構造解析、床版設計、線形データ処理・作画、主桁断面、対傾構・横構、分配横桁、スタッドジベル、補剛材、現場継手、中間支点上床版、主桁作用と床版作用、断面構成図作画の13のサブシステムから構成されている。

各サブシステムのそれぞれについて、先に記述した7種類のドキュメントを同じ記述形式で作成する。

また、自動現寸システムについては、各サブシステムとして入力処理、立体座標処理、キャンパー展開、参考図作成、対傾構処理、横桁、横構、主桁、材料管理、マーク図作成、型板作成、定規作成、帳票作成の13のサブシステムから構成されている。その各サブシステムについて、同様に7種類のドキュメントを同じ記述形式で作成している。

つぎに、ドキュメント作成にあたっての具体的な表記基準についてふれてみる。

サブシステム関連図においては、システム構成上必要となるすべての機能の洗い出しを行い、サブシステムに分割し、その各サブシステム間の機能的関連と処理の流れを明示する。

サブシステム説明書においては、サブシステムの入出力を含めた機能および目的の基本事項とその特長について記述する。

問題解析書においては、ユーザ要件とシステムの機能、性能は十分か、プログラム間のインタフェース仕様は明確かなどシステム設計書の整合性のチェックを記述する。

ファイル設計書においては、ファイルの概要およびそのデータ構造について明示する。

つぎに、プログラムを機能の最小単位であるモジュールに分解する方法としては機能分解を行い、モジュール構造図において、モジュール構造を表わし、その相互間の関連のインターフェイスを明らかにする。モジュール機能説明書において、その機能を明示する。これによって、プログラムの設計変更が小さな範囲内だけしか影響を与えないので、プログラム変更が容易になる。

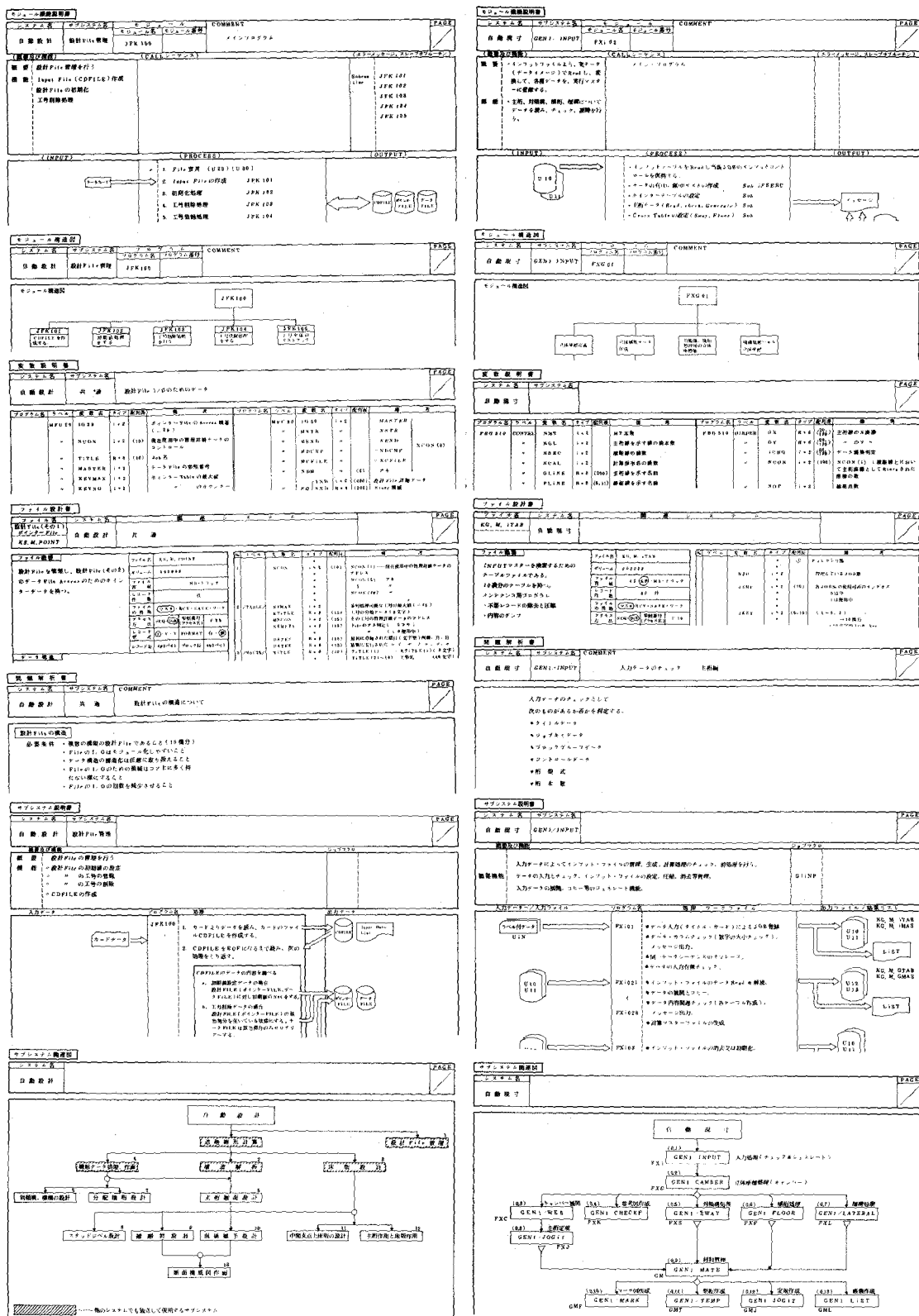


図-2. システム設計書

