特集論文

大規模な現行システムを効率的に見える化する技術

堀田朋子* 川口正高*
朱雀 健* 松田昇平*
小俣正樹*

Technology to Visualize Efficiently Current Large-scale Systems

Tomoko Horita, Ken Sujaku, Masaki Omata, Masataka Kawaguchi, Shohei Matsuda

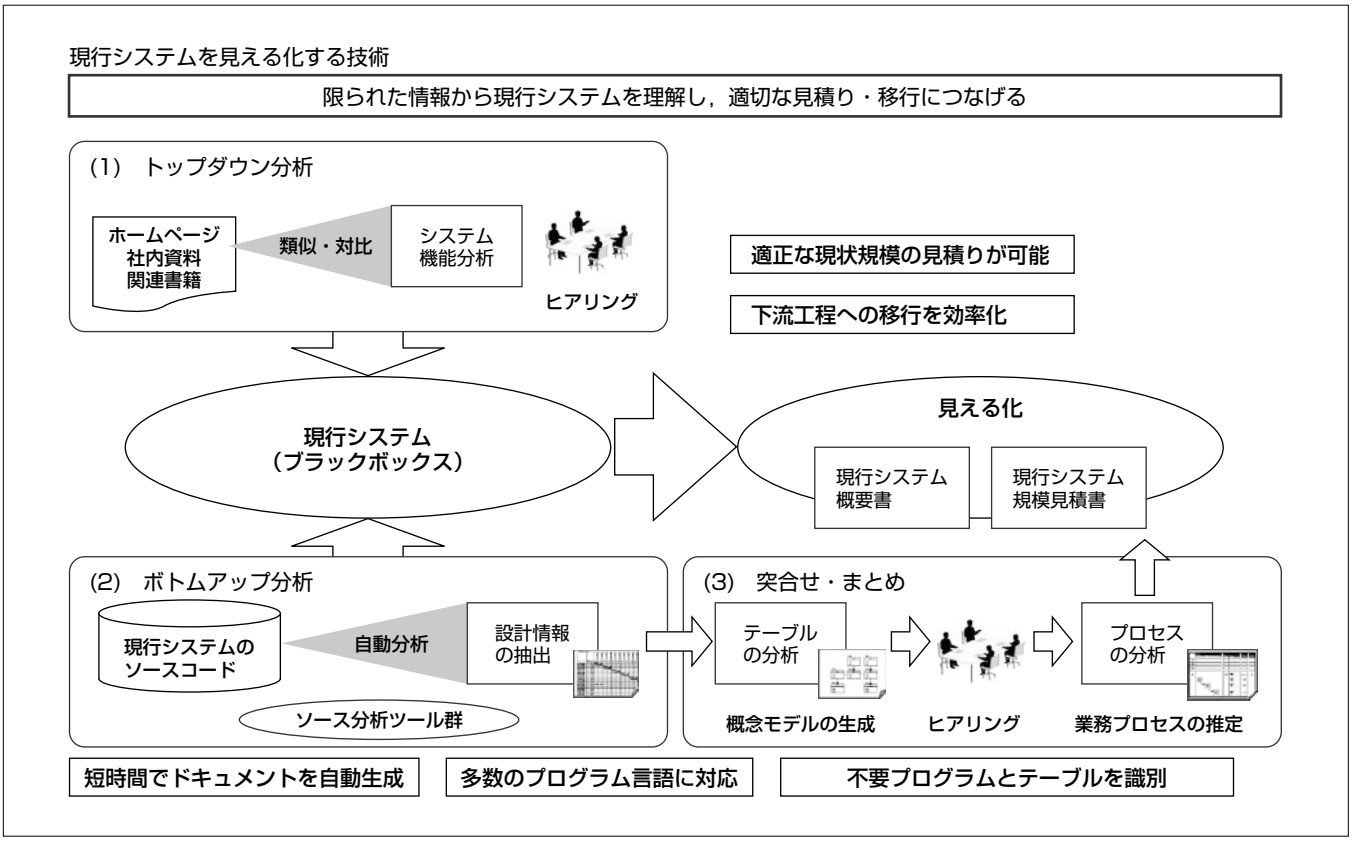
要 旨

既に稼働している大規模なITシステムの再構築や機能追加などの移行開発では，基となる現行システムの“仕様書がない”“開発当時の設計者がいない”“顧客も仕様を知らない”というシステムのブラックボックス化が，適正な規模見積りを得る上での非常に大きな課題となっている。

三菱電機インフォメーションシステムズ㈱(MDIS)では，この課題に対応するため，現行システムからシステムの機能，構造，設計情報を把握し，適切な見積りを得る“現行システムを見える化する技術”を開発した。

この技術は，3つのフェーズで構成する。最初に，対象企業の公開情報や関連性の高い業務を分類・整理した業務参照モデルを収集し，類推・対比しながら業務とシステム機能の関連を把握する“トップダウン分析”を行う。次に，現行システムのソースコードやデータベースを自動的に分析して見積りに必要な設計情報(プログラム間の関連など)を短期間に得る“ボトムアップ分析”を行う。最後に，2つのフェーズの分析結果である機能とプログラムを突き合わせた結果から，FP(Function Point)概算法を用いて機能単位に規模を算出する“突合せ・まとめ”を行う。

この技術を活用することによって，ブラックボックス化した大規模なITシステムであっても，不要なプログラムを識別して見積り対象から除外すること，根拠が明確な機能ごとの規模見積りを短期間で得ることが可能になり，予算に応じた開発対象範囲を特定することができる。



現行システムを見える化する技術

“現行システムを見える化する技術”には，“トップダウン分析”“ボトムアップ分析”，その結果の突合せを行う“突合せ・まとめ”の3つのフェーズがある。この3つのフェーズから，適切な見積り・移行につなげる“現行システム概要書”“現行システム規模見積書”を完成させることができる。

*三菱電機インフォメーションシステムズ㈱

1. まえがき

既に稼働している大規模なITシステムの再構築や機能追加等の開発では，基となる現行システムの“システムの仕様書がない”“開発当時の設計者がいない”“顧客も仕様を知らない”といったブラックボックス化が，適正な規模見積りを得る上での非常に大きな課題となっている。この課題を解決するために，現行システムからシステムの機能，構造，設計情報を把握し，適切な見積りを得る技術を開発した。

本稿では，ブラックボックス化してしまった大規模なITシステムを短期間で効率的に把握し，FP概算法を用いて機能単位に規模を算出する“現行システムを見える化する技術”とその適用事例・効果について述べる。

2. 現行システムを見える化する技術

現行システムを見える化する技術は，システムの機能，構造，設計情報を把握し，適切な見積りを得る技術である。

この技術は，次のことを目的としている。

⑴ 適正な現状規模を見積もる

FP概算法を使った規模や工数の見積りを実施するために必要となる設計情報を，プログラムからリバースエンジニアリングによって作成する。

⑵ 現行システムのプログラムの棚卸し

現行システムのプログラムを棚卸しすることによって，使用しているプログラムやテーブルを明確にして移行対象範囲を特定する。

この技術は，“トップダウン分析”“ボトムアップ分析”“突合せ・まとめ”の3つのフェーズから構成される。

2.1 トップダウン分析

トップダウン分析では，現行システムからは得られない情報を事前収集する。顧客へのヒアリングを最小限にするために，対象システムの業務，機能，及びデータがどのようなものか，事前に見当をつける。

次の手順でトップダウン分析を行う。

⑴ 業務・機能分析の準備

①分析対象となる企業情報，業務に関する情報を収集する。具体的には，分析対象企業のホームページ，業務関連書籍，業務パッケージソフトのドキュメント，類似開発の社内成果物，業務参照モデル[1][2]等を参考にする。なお，業務参照モデルとは，関心対象領域の構造や関係を体系的に分類・整理したモデルである。

②不明点について，分析対象の領域に関する知見がある社内有識者と質疑応答を行い，理解する。

③収集した情報を体系化する。具体的には，会社基本情報，ビジネスモデル，業務関連図，機能モデル，初回ヒアリングシートを作成する。

⑵ 業務・機能の分析

現行システムのドキュメントや分析結果から，現行システムの業務・機能を把握する。

①バッチスケジュール表，概略レベルの処理フロー，操作説明書等のドキュメントを入手する。

②現行システムのテスト環境を使って，メニュー画面構成を分析する。

③収集した情報を基に， 現行システムの業務・機能一覧を作成する。

2.2 ボトムアップ分析

ボトムアップ分析では，ソース分析ツールを利用して，プログラムから自動的に見積りに必要な設計情報を把握するアウトプットを生成する。アウトプットには，プログラムの呼出し関係やプログラムがアクセスしているテーブルの関係等がある。

次の手順でボトムアップ分析を行う。

⑴ 事前準備(アーキテクチャ(構成，関連，役割)の分析)

プログラムのフォルダ構成，ファイルの種類，命名規則等から，プログラムの役割や相互の関連等を分析する。

⑵ ソース分析

ソース分析ツールを利用してプログラムから得られる情報を自動的に抽出し，画面呼出し関連，プログラムがアクセスするテーブル情報などのアウトプットを作成する。代表的なアウトプットを**表1**に示す。

⑶ 不要プログラム，画面，テーブルの棚卸し

適正な見積りを行うために，どのプログラムからも呼ばれないプログラム，画面，テーブルを自動的に抽出し，不要プログラム，画面，テーブルの棚卸しを行う。

2.3 突合せ・まとめ

トップダウン分析とボトムアップ分析で作成した機能分析やプログラム構造，設計情報の結果の突合せを行い，FP概算法を用いて機能単位に規模を算出し，現行システム規模見積書を作成する。さらに，システムの設計情報を

表1．ソース分析ツールの代表的なアウトプット

分類	ソース分析ツールのアウトプット
プログラム規模・関連調査ツール	プログラムライン数一覧
	プログラム呼出し一覧
	プログラム呼出し構成図
	未使用プログラム候補一覧
	ソース流用度一覧
	ソース流用度マトリックス表
画面関連ツール	画面一覧
	画面項目定義書
	画面呼出し関連図
	未使用画面候補一覧
帳票関連ツール	帳票一覧
	帳票項目定義書
テーブル関連ツール	プログラムデータベース関係表
	CRUDマトリックス表
	未使用テーブル候補一覧

CRUD：Create，Read，Update，Delete

まとめた現行システム概要書も作成する。

次の手順で突合せ・まとめを行う。

⑴ テーブルの分析

テーブルの分析では，次工程のシステム化業務フローの分析で行う顧客へのヒアリングや顧客との合意形成がスムーズに進むように，データモデルとして表されているシステムのビジネスルールや顧客特有の概念を理解する。

具体的には，ボトムアップ分析で得られたテーブル名を単語単位に分割して修飾語＋主要語＋分類語の“主要語”に当たる単語をエンティティとして抜き出し，主要エンティティ間の関係をモデリング[3]の表記方法で表す(概念モデル)。

⑵ システム化業務フローの分析

これまでの作業(**2. 1節**⑴～**2. 3節**⑴)を通じて得られた結果から理解したシステムの内容をシステム化業務フロー(**図1**)として表す。システム化業務フローでは，概念モデルで登場した業務のイベントや管理対象を表す言葉を使うことによって，粒度を合わせる。また，縦軸は時間軸，横軸は業務，業務に対応するプロセス，業務プロセスに対応する画面，画面に対応する入力・出力情報のように列を分割して，内容を整理する。

①機能別にプログラムを分類

トップダウン分析で作成した機能一覧から，各プログラムやテーブルがどの機能に属するかを分類する。

②現行システム規模見積書の作成

ボトムアップ分析で作成した未使用プログラム一覧の結果から，不要プログラムを省いたライン数，FP数を算出し，機能ごとに規模を見積った現行システム規模見積書を作成する。

③現行システム概要書の作成

現行システムの概要を把握するため，トップダウン分析，ボトムアップ分析，突合せ・まとめで得られた結果を整理して，現行システム概要書を作成する。これらは，下流工程であるシステム設計書の元資料とすることができる。

3. システムを見える化するソース分析ツール

3.1 ソース分析ツールの概要

ソースプログラムが1ML以上ある大規模なITシステムを分析するには，人手と時間がかかるとともに，抜け・漏

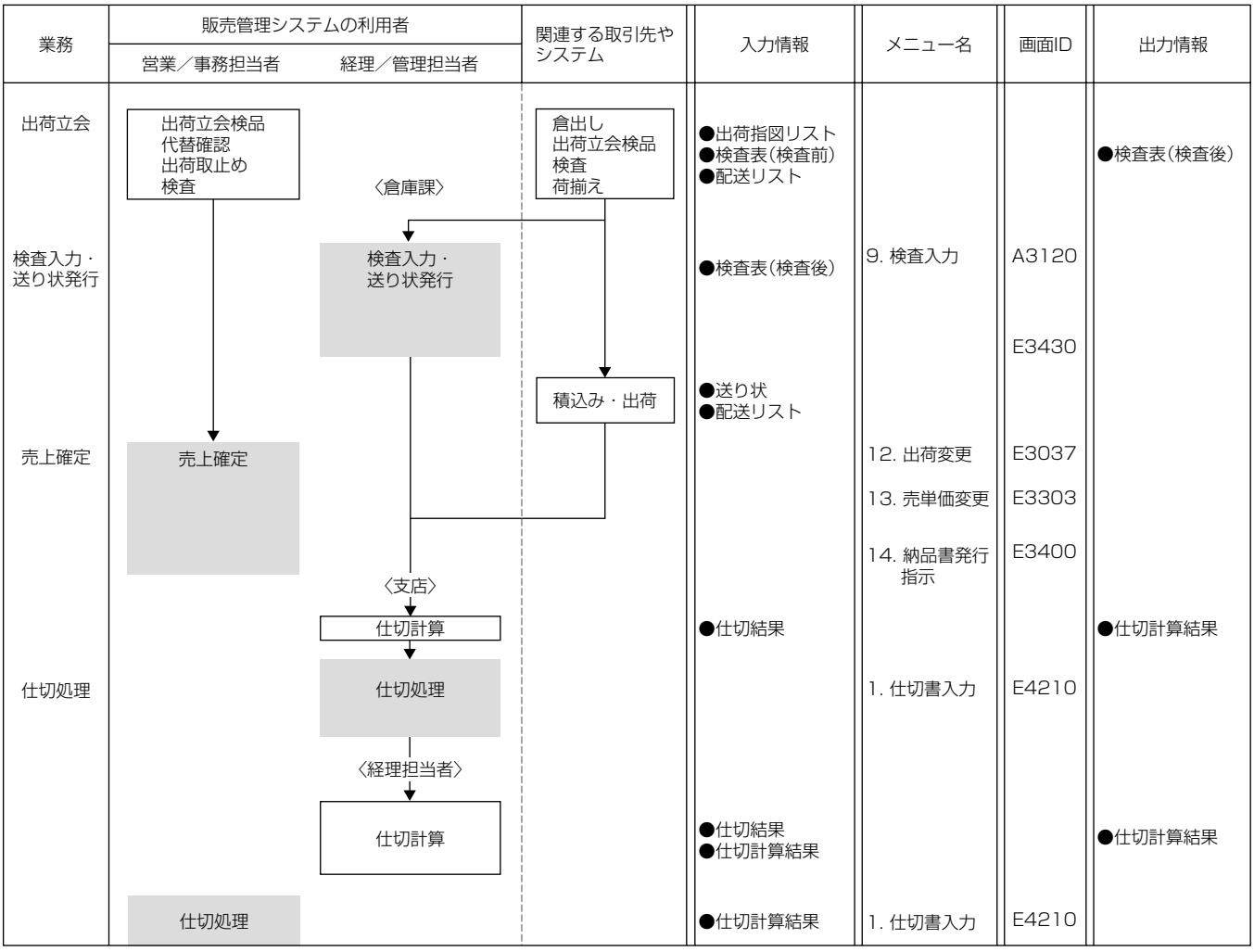


図1．システム化業務フローの例

れが発生しやすい。

そこで，ボトムアップ分析で効率的かつ網羅的にシステムを分析するために，ソース分析ツールを開発した。このツールを活用することで，プログラムの呼出し関係の確認や不要なプログラム，画面，テーブルの棚卸し等の分析を短期間で行うことが可能である。

このツールは大きく次のように分類される。

⑴ 分析準備用ツール

次の⑵～⑸の分析を行うための準備として，プログラムの記述コードや改行コードの統一，プログラム内のコメント削除等を行う。

⑵ プログラム全体の規模や関連を分析するツール

プログラムの規模や関連，重複しているプログラムの分析を行う。例えば，プログラムのライン数，プログラムの呼出し関係(**図 2**)，未使用プログラムの抽出がある。

⑶ 画面関連ツール

画面項目定義(**図 3**)，画面呼出し関係，未使用画面の抽出等の画面関連の分析を行う。

⑷ 帳票関連ツール

帳票一覧と帳票項目定義書を作成し，帳票についての分析を行う。

⑸ テーブル関連ツール

プログラムが利用しているテーブルの関係や未使用テーブルの抽出等の分析を行う。

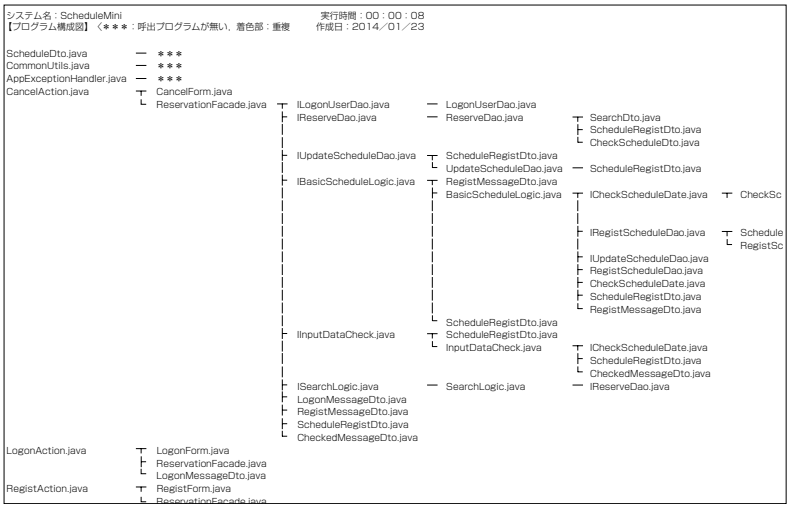


図 2．プログラム呼出し構成図の例

システム名		局面	画面仕様書 画面項目定義書	作成者	作成日	ページ
vb_Sample		現状システム分析				

ファイル名　D:¥画面項目定義書¥テストソース¥CreateOptionForm.designer.vb

画面ID　CreateOptionForm　画面名　生成オプション

No	コントロール名(Name)	表示名(Text)	コントロールタイプ	位置	処理機能有無	備考
1	createVMTable	生成一覧	グループボックス	8.8	0	
2	serversListBox		リストボックス	3.15	0	
3	Panel1		パネル(グループ化)	0.261	0	
4	Panel2		パネル(グループ化)	0.0	0	
5	Panel3		パネル(グループ化)	220.0	0	
6	executeButton	生成	ボタン	74.6	1	
7	executeCancelButton	キャンセル	ボタン	155.6	1	
8	Panel4		パネル(グループ化)	0.0	0	
9	FlowLayoutPanel1		パネル(動的配置)	0.0	0	
10	optionList	生成オプション	グループボックス	297.8	0	
11	createOptionCheckedListBox		リストボックス(チェックボックス付)	3.15	0	

図 3．画面項目定義の例

3.2 ソース分析ツールの特長

通常，ツールを使用して現行ソースを分析するには，対象システムのプログラミング言語に応じたツールの作成が必要で，多種類の設計情報を得るためにはツールの開発自体に時間がかかってしまうという課題があった。また，プログラムの棚卸しは，稼働ログを解析するのが一般的で，稼働ログが取れないシステムでは不要なプログラムやテーブルの候補を洗い出すのが困難である。

本稿で述べるソース分析ツールはこれらの課題を解決した次の 3 つの特長がある。

⑴ 多数のプログラミング言語に対応

ソース分析ツールは，情報を抽出するための単語(情報抽出単語)をキーとして，プログラムの設計情報を抽出している。

例えば，JavaScript(注1)言語で画面遷移先を指定する場合は，href(location.href)やaction(document.formName.action)等の命令文を使用する。このhrefやaction等の単語をソース分析ツールの設定データとして記述することで，画面遷移先をソース分析ツールが抽出する。

プログラム言語やアーキテクチャが異なる場合，この情報抽出単語も異なるためツールのカスタマイズが必要であるが，情報抽出単語をソース分析ツールの設定データとして記述することで，ツールはプログラム言語に依存しなくなる。そのため，ツールはJava(注1)，Visual Basic(注2)，Visual C＃(注2)，COBOL等現在19のプログラム言語に対応している。

システム移行を行っている企業にも現行システムを自動的に分析する技術がある。しかし，それら企業のソース分析ツールは構文解析などプログラム言語依存の処理をしているので，JavaやCOBOL等限られたプログラム言語にしか対応できていないのが実状である。

MDISが開発したソース分析ツールは，キーとなる単語を設定する仕組みであるため，今後更にプログラム言語を拡大することが可能である。

⑵ 多様な設計情報生成ツールを短期間で準備

このツールは，30種類以上の設計情報を生成するツール群から構成されている。これらのツールは，設計情報の特性に応じてパターン化した数種類のテンプレートをもとに少量のカスタマイズで作成することができる。そのため，短期間でツールの準備ができる。これらのツールを実行することによって，多様な設計情報を生成できる。

⑶ 不要なプログラム，画面，テーブルの棚卸し

稼働ログが採取できないシステムでも，ソー

ス分析ツールで生成される未使用プログラム候補一覧，未使用画面候補一覧，未使用テーブル候補一覧から不要なプログラム，画面，テーブルの棚卸しを行うことができる。

（注１） JavaScript，Javaは，Oracle Corp. の登録商標である。
（注２） Visual Basic，Visual C＃は，Microsoft Corp. の登録商標である。

4. 適用事例と効果

4.1 某企業向け受発注システムへの適用事例

現行システムを見える化する技術を適用したことで，当初4.5人月で想定していた作業を約1/3の工数で実現した事例について述べる。

⑴ 事例の概要

この事例は，サーバOSのサポート切れに伴いシステムを刷新する開発である。他社が開発したシステムで，担当者はシステムの中身を把握していない。さらに，現行システムの仕様書や資料もほとんどない中，現状分析作業は，4.5人月かかると想定され担当者１名で行わなければならなかった。

⑵ 適用効果

①分析ツールの活用で多彩なアウトプットを作成

当初担当者は，必要なアウトプット約15種類を作成する計画であったが，ソース分析ツールを利用した結果，約30種類のアウトプットを作成できた（**図４**の効果１）。また，人海戦術で分析を行った場合に比べ，短期間で網羅的にプログラムの分析を行うことができた。そのため，プログラムの構造的な問題点（同じ名前のプログラムが複数存在など）が分かり，新システムへの移行作業を行う上での注意点が明らかになった。

②短期間でより網羅的にプログラムの分析を実施

ソース分析ツールで効率的にシステムの分析を行ったため，当初4.5人月で想定していた作業を約35％の1.5人月で行うことができた（**図４**の効果２）。この事例の場合，ソース分析ツールの一部（主に画面系のツール）が対象プログラム言語やアーキテクチャに対応していなかったため，ツールのカスタマイズ作業が発生しており，作業にかかった工数のほとんどはカスタマイズ工数であった。

③不要プログラムの棚卸し

ソース分析ツールを利用して，どこからも呼ばれていないプログラムを自動的に抽出した。その結果，画面プログラムの約60％，ロジックプログラムの約70％が不要なプログラムであることが分かった（**図４**の効果３）。

4.2 某企業向け販売管理システムへの適用事例

販売管理システムを再構築する開発にこの技術を適用したことで， 短期間に重複プログラムが約30％あることや，当初再構築対象外であると想定していた必要プログラムを見つけ，適正な見積りができた事例について述べる。

⑴ 事例の概要

この事例は1990年代前半に他社が開発したオフコン上の基幹システムを，Windows(注3)システム化する開発である。新システムは現行機能を踏襲するが，メンテナンスされた現行システムの仕様書はない。また，顧客に仕様をヒアリングするのは体制や時間の制約上困難なため，現行プログラムを正として仕様書のリバース作成と新言語のプログラム作成を行った。

顧客から提供された再構築対象プログラムは1.5ML以上もあり，不要プログラムの棚卸しと棚卸し結果に基づく見積りを短期間に人海戦術で行うのは困難であった。

⑵ 適用効果

①重複プログラムを除いた見積りを提示

この事例は予算の大枠が決まっていたため，その範囲内で対応可能な再構築対象機能を絞り込む必要があった。

ソース分析ツールを利用して流用度分析を自動的に行

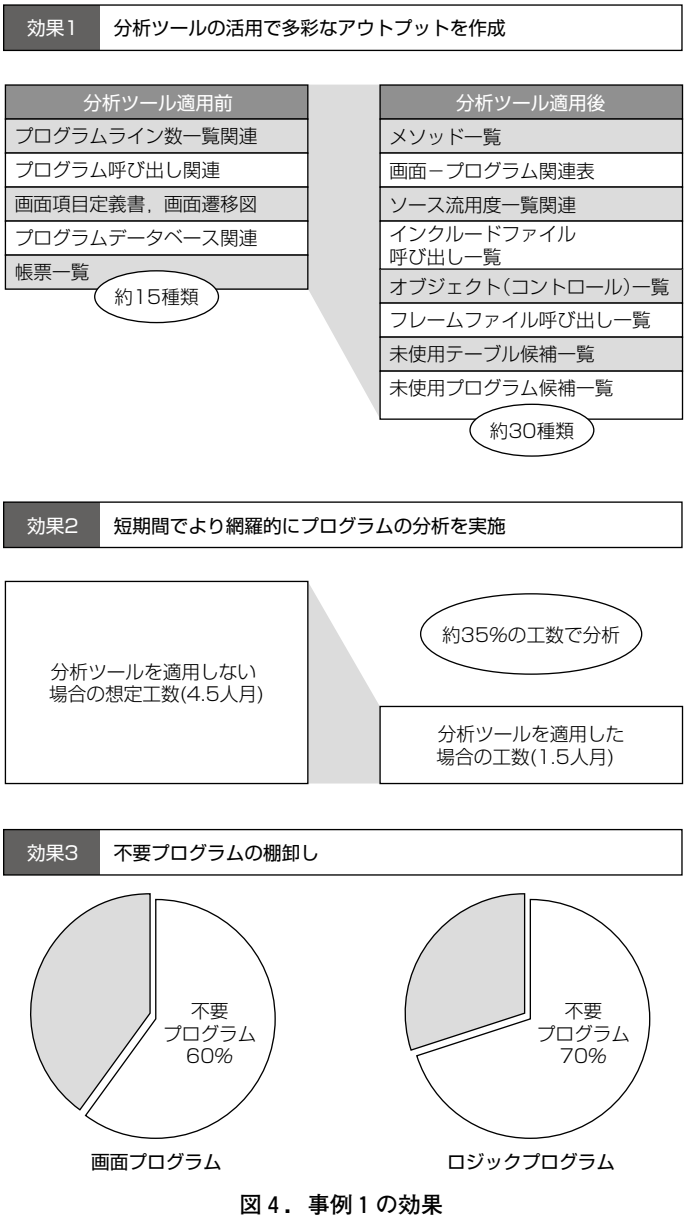


図４．事例１の効果

うことによって重複部分を除外すると，プログラムの規模は約70%になることが分かった(**図5**の効果1)。この分析結果を見積時に考慮することによって，予算枠に収めることができた。

②再構築対象プログラム間の整合性を確保

再構築対象プログラムの妥当性を，ソース分析ツールを利用して確認した。具体的には，再構築対象プログラムから再構築対象外プログラムを呼び出していないか，再構築対象プログラムはバッチスケジュールやトップ画面メニューから運用されているかについて分析した。妥当性が疑わしいプログラムは顧客に確認を取り，その確認結果から再度同じ分析作業を行って最終判断を行っていたため，複数回同じ作業を行う必要があった。手動で行うと，作業ミスや再分析に多大な工数がかかるが，この技術を利用することによって，1.0人月以下の工数で約20%の再構築対象外であると想定していた必要プログラムと再構築対象プログラムから呼ばれていない不要プログラムを見つけることができた（**図5**の効果2)。

③業務と再構築対象プログラムの整合性を確保

“再構築対象プログラムを使って従来と同様に業務を行うことができるか”という観点で，トップダウン分析で作成したシステム化業務フローに登場する画面や帳票と再構築対象プログラムを突合せる作業を，ソース分析ツールを用いて自動化した。その結果，システム化業務フローでは抽出し切れなかった画面・帳票漏れ(総数の約10%)を発見できた。これは，特殊業務で利用する画面や帳票であり，業務の漏れも合わせて発見できた(**図5**の効果3)。

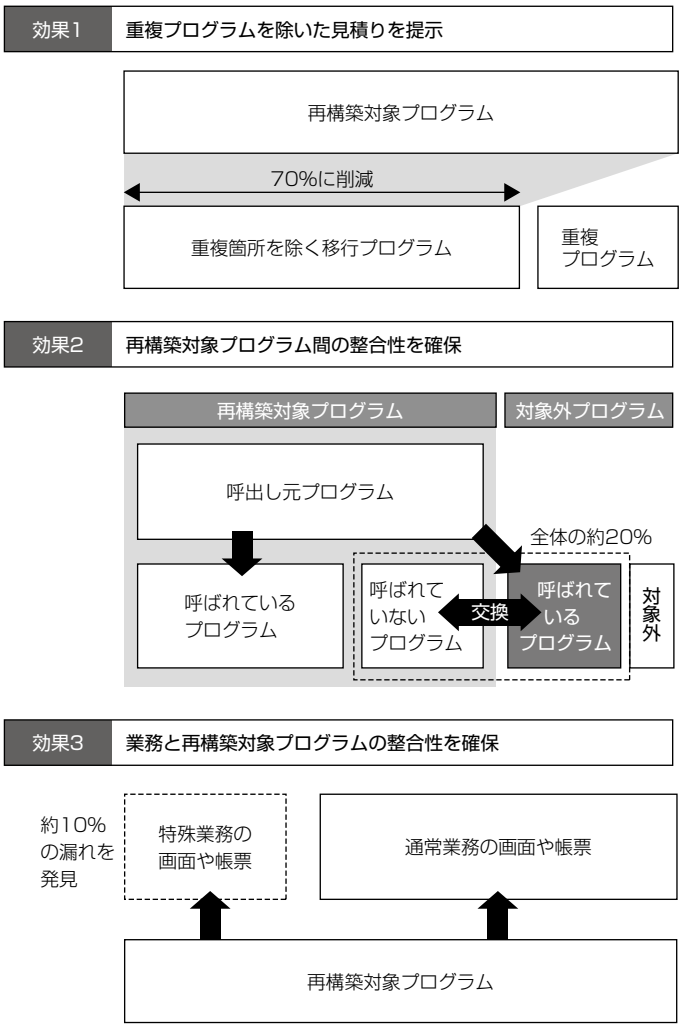


図5．事例2の効果

(注3) Windowsは，Microsoft Corp. の登録商標である。

5. むすび

大規模なITシステムを短期間で効率的に分析し，適切な見積りを行う技術と，この技術を利用した事例と効果について述べた。この技術によって，適正な現状規模を見積ること，現行システムのプログラムの棚卸しという2つの目的を達成した。

今後は，見積りのために抽出したシステムの設計情報を活用して下流工程の移行効率化につなげるための開発を実施予定である。この次期開発が完了すると，正確な現行システムの分析結果を活用して，その後の要件定義や開発工程をスムーズに遂行することができる。また，上流での漏れを防ぐことで，顧客の予算通りに開発を行うことができるようになる。

なお，プログラム言語に依存しないソース分析ツールを狙いとして開発したが，画面系などの一部のツールでは，アーキテクチャに依存する部分があり，ツールのカスタマイズが必要になる場合もあり得る。今後より多くの開発に適用することでカスタマイズなしでソース分析ツールが利用できるよう，更なる対象プログラム言語・アーキテクチャの拡大，ソース分析ツールの精度向上と拡張を行い，適用範囲を拡大していく予定である。

参考文献

(1) 隈　正雄：SEのための「経験則的」要件定義の極意，技術評論社（2009）
(2) バリューチェーンプロセス協議会：プロセス参照モデル
http://vcpc.org/modules/pico_3/index.php?content_id=9
(3) UMLモデリング推進協議会：モデル脳検定
http://www.umtp-japan.org/modules/modeno/