CASE実施例

—富士通のケース—

藪田　和夫，森　　偉作

1．まえがき

近年コンピュータが金融機関やメーカなどの業務にも使われるようになるにつれ，ソフトウェアの需要が急速に拡大している．ところが他の工業製品と違い，ソフトウェア開発は自動化・機械化が容易ではないため，供給が需要に追いつかないという事態にまでなっている．そこでソフトウェア開発の生産性向上の努力がされてきたわけだが，いわゆるCASEツールはそうした背景から生まれてきた．

企業レベルのシステム(以下，業務基幹系システム)は規模が非常に大きく，数千から数万という膨大なデータ項目を扱うため，冗長で保守の困難なシステムが作り出されることが往々にしてある．従来の基幹系システム開発には以下のような問題が挙げられる．

・データ項目の冗長性：

従来の方法はプロセス中心アプローチと言われ，システムを機能ごとに分割し分析・設計をしていた．機能を中心にして分析・設計作業を進めるためシステム全体にわたってデータ項目定義の標準化や規格化がされず，重複のあるファイルや画面が各機能ごとにばらばらに作成されがちになり，保守性の低下を招いた．

・業務に用いる用語の曖昧さ：

どの業種においても言えるが，普段から慣用的に使われている業務用語でも，システム分析においてあらためてその意味・用法を問いただすと，案外はっきりと定義されずに用いられていることがよくある．業務用語の曖昧さは要求仕様獲得の困難さ，冗長なデータ項目の定義をもたらし，結果として作成されたシステムも保守性は低くなる．

・業務仕様保守の問題：

本来業務の仕様は，システムが稼働するOSやデータベースのアクセス方法などコンピュータの処理にかかわる部分とは独立である．ところが従来の技法ではプログラムにおいてこの2つが分離できないために，業務仕様による変更やデータベースシステムの変更に伴う保守は困難であった．

本稿ではこういった従来のシステム開発が抱える問題を解決するためのデータ中心のシステム開発技法「アプリケーション・アーキテクチャ」(以下AA)について説明し，次にそのAAにもとづくSDAS統合CASE環境，さらにSDAS統合CASEの実際の適用事例とその事例におけるAAとCASEの効果と課題について説明する．

2．AAの分析・設計の方法論

AAを整備する以前，富士通では上流の分析工程のみサポートするC-NAP IIというシステム開発技法を用い，実績を上げてきた．C-NAP II自体はデータ中心アプローチであったが，下流工程への一貫性，保守性において限界があった．そこで，これを全工程をカバーし，ニーズ分析，システム分析，業務仕様設計，実現設計に関する技法からなる一貫性のある方法論として拡張したのがAAである．

ここでは，AAが，

・データ中心による分析・設計
・データの標準化
・業務仕様とコンピュータ制御の分離

という特徴を持ち，上のまえがきで述べた従来のシステム開発の持っている問題を解決できることを示し，さらに，AAで用いられる厳密な仕様記述法であるBRSPEC (Business Rules SPECification) と，後の(3)で述べるが，業務仕様設計とは別に進められる実現設計について簡単に説明する．

(1) データ中心による分析・設計

従来のプロセス中心アプローチによるシステム開発

やぶた かずお，もり いさく　富士通㈱
〒261 千葉市美浜区中瀬1-9-3

では，まえがきで述べたように，システム全体で一貫性のあるデータ項目定義ができず，重複のあるファイルやデータがプログラムごとにばらばらに作成され，保守性の低下につながる．したがって，データ項目を一元管理して冗長性を取り去ることは非常に重要である．

AAでとられているデータ中心アプローチは「データはある1つのプログラムの所有物ではなく，企業全体の共有資源である」という考え方が基礎となっている方法論で，管理対象分析および正規化という方法を用いて，データの重複の最小化，データの格納効率の最適化をする．正規化はリレーショナル・データベースの設計にも使われる技法で，繰り返しデータ項目の分離，分離されたレコードへの識別キーの設定という操作を繰り返しながら，データの重複を最小化していく．

このように正規化により得られたモデルは「概念データモデル」といい，

・繰り返し構造は持たない．
・重複したデータ項目は持たない．
・識別キーに対して完全関数従属するデータ項目で構成される．

といった特徴を持つ．概念データモデルではデータを通じた業務のルールを明確とするため，データの論理的な側面だけを取り扱い，データベース設計において「物理的にどこにどのように格納されているのか」，「どうやって取り出すのか」といったことは考えないということに注意する必要がある．そして概念データモデルは概念ファイルから構成され，正規化されたレコードに対応している．

(2) データの標準化

まえがきで要求仕様獲得の困難さ，冗長なデータ項目の定義によるシステムの保守性の低下についても述べたが，データ項目の名称，意味，形式を標準化し一元管理することは情報システムの開発において最も基本的な事柄で，そのためにはデータの意味を確定するモデルが必要である．a）で述べた概念データモデルはその1つであり，概念データモデルを構成する概念ファイルはデータベースの論理的な構造を表わし，いくつかのデータ項目によって構成される．

AAでは，さらにいくつかの概念を用意しており，その1つに「ドメイン」という概念がある．これは，概念ファイルを構成するそれぞれのデータ項目が共通に持つ「型」で，データの取り得る値の単位，範囲などを整理したものである．たとえば，製品の重量の単位が，各製品で統一できないという場合，重量の単位コードと重量の数値からなる製品重量というドメインを構成する．製品1と製品2の重量単位が違う場合は，製品重量1の単位で2つの製品の合計などを計算する．

多くの業務アプリケーションは，プログラム中でデータの範囲をチェックし，チェックの結果に応じてさまざまな処理をしている．従来の方法ではチェック処理があるたびにチェック内容の詳細を記述する必要がある．しかしチェック対象となる範囲をドメインとして定義しておけば，プログラム中のチェック処理ごとに詳細なチェック内容を記述する必要がなくなる．

またドメインの他に「コード」と「業務用語」および「業務関数」という概念がある．コードとは業務で使われる言葉の意味をコンピュータが扱いやすいように番号体系などで表わした置き換えの仕組みである．この番号体系と業務における意味の対応は，個々のデータ項目ごとに管理するのではなく，ドメインに関連した情報のコードとして共通に管理する．たとえば，性別を表わすときは，「男」に対しては「1」，「女」に対しては「2」，「不明」に対しては「0」というように性別区分のコードを用いる．ただし業務仕様の表現には「0」「1」「2」ではなく，「不明」「男」「女」という業務の言葉を使う．

業務用語は，プログラムの中で現われる複雑な条件や処理を使い慣れて理解しやすい言葉として登録されるもので，仕様の読みやすさを向上させるために使う．たとえば，出荷取消処理をするための条件などを業務用語として別途定義することにより，業務の仕様から複雑な部分を減らし，業務仕様を読みやすくすることができる．

しかし仕様によっては業務用語などにより宣言的な方法で仕様を表現することがあり，複雑な手続きでしか正確に表現できないことがある．業務関数はそのような複雑な手続きを定義したものである．プロセスの仕様からこの業務関数を参照する．

なお，データの標準化を実施するには，データ分析の技術だけでなく，データ管理も重要である．つまりデータ管理者を置いて，標準としてのデータの形式・名称の維持，新たに発生したデータ項目がデータ管理者により承認されなければならない仕組みを作る必要がある．またこのような仕組みを実装するCASEツールも要求される．

(3) 業務仕様とコンピュータ制御の分離

一般に業務アプリケーションは大きく2つの要素からなりたっている．1つは業務仕様設計といい，入力データ項目に対するチェック条件や計算方法などデータの取り扱いルールに関する部分で，純粋に業務側から設計が行なわれる．もう1つは実現設計といい，業務仕様設計で得たルールを実際のコンピュータ上でプログラムとして動かすための制御部分で，データベースの編成法やOSなどの実現手段によって異なるため，実現設計と呼ぶ．

まえがきで業務仕様による変更やデータベースシステムの変更などに伴う保守の困難さについても述べたが，業務仕様とコンピュータ制御とを明確に分離することにより，コンピュータ制御部分の技術的な内容設計を知らなくても，業務仕様だけを直接変更可能にすることができ，またシステムが稼働するOSやデータベースのアクセス方法なども業務仕様とは独立に変更が可能であり，システムの保守性を大きく向上させることができる．

(4) BRSPECについて

BRSPECは業務の仕様を表形式のわかりやすい形で厳密に定義するために開発された仕様記述法で，データ項目に対して処理内容を定義するというデータ中心の考え方にもとづいている．AAの業務仕様設計において各種ドキュメントを作成するが，その一部でBRSPECを用いる．

従来の方法では手続きの中でデータの値が変化していくため，データの変化を追跡しないと個々の手続きの仕様が理解できなかった．これに対しBRSPECはデータ中心アプローチにもとづいてデータに対して仕様を記述する．したがって，たとえば，あるデータがどういう条件下でどういう値を取りうるのか，またあるチェックでエラーが発生するのはどういう条件の場合か，といったことがわかりやすく記述できる．

(5) 実現設計について

上の(3)でも述べたが，AAでは特定の実現手段は前提にせずにシステム分析と業務仕様設計を行なう．実現設計では，使用するハードウェア，OS，データベースシステムなどの条件に合わせて，技術的・物理的な情報を決めていく．

具体的には，

・入出力設計
・プロセス内の制御設計
・論理ファイル設計

を決定する．入出力設計では，対話手順の設計と，入出力情報の定義から画面や帳票の物理的な形状（レイアウト）の設計を行なう．

プロセス内の制御設計では，業務仕様設計で業務的な要件が決められた処理をいかに実行するかをプロセスフローをもちいて精密に設計する．また，制御スケルトンと呼ばれる，プロセス内の制御の標準的な型を決めて，それにより制御設計を行なっており，設計の効率化だけでなく，プログラムの保守性にも大きな効果が得られる．対話制御，データベースへのアクセスの制御もこの制御スケルトンが受け持ち，概念ファイルと実際のデータベースの差を吸収するように設計される．

論理ファイル設計とは，実現するファイル編成やデータベース管理システムに合わせたデータベース設計であり，アプリケーション・プログラムから見えるデータベースのモデルである．実際のデータの格納設計は物理ファイル設計と呼ばれる．(1)でも述べたが，概念モデルは正規化により設計され，同じ正規化を設計基準としているリレーショナル・データベースでこの概念モデルを実現する場合は論理ファイル設計は不要となる．その他のデータベースでも正規化のレコードは多くの場合そのままの形で論理ファイル設計に生かすことができる．

3．AAにもとづいた統合CASE環境

AAをサポートするCASEはSDAS統合CASEとして提供されており，すべてMS-Windows上で動作する．AAにおいて，業務仕様とシステムの実現手段は分離されていることは前章で述べたが，ツールもこれに呼応して，業務モデルの作成・保守を支援する業務モデリングツール“AA／BRMODELLER”と，機種，OSなどのターゲットシステム環境に合わせたプログラムを生成する構築ツール群の2つの体系に分れる．後者の構築ツール群には，“YPS／BLDR”，“FORM”などが提供されている．

AA／BRMODELLERによりシステム分析，業務仕様設計を行ない，構築ツール群によりプロセス内の制御設計などの実現設計およびプログラムの自動生成を行なう．SDAS統合CASEの体系を図1に示す．

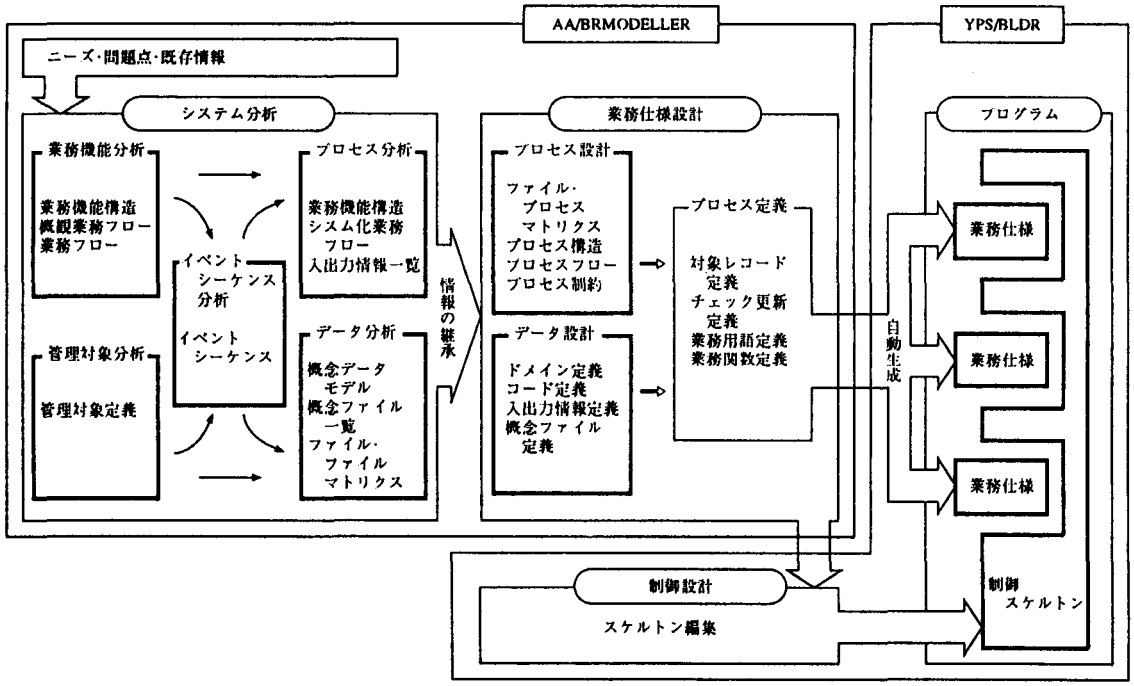

図1　SDAS統合CASEの体系

(1) 業務モデリングツール

AA／BRMODELLERは，業務仕様を表わすさまざまなドキュメント（以下CASEドキュメント）の作成支援，および論理的な整合性のチェック機能を持っている．これらCASEドキュメントはプロセス，データ，イベントの3つの観点から業務を表現し，ツールにはそのためのエディタ（図形エディタ，表エディタ，マトリクスエディタの3種類）および標準的な図・記号・作成ルールが与えられている．前章で述べたBRSPECもその1つである．AA／BRMODELLERのこの他の機能を以下に示す．

- 関連ドキュメントを呼び出す：すべてのCASEドキュメント間で業務分析で得られたプロセス，データなどの相互参照が可能である．
- マトリクス生成：システム分析で作成したドキュメントからファイル間の親和性，ファイルとプロセスの関係を表わすマトリクスを自動生成する機能．この自動生成されたマトリクスにより，プロセスやデータの漏れ，冗長性をチェックできる．
- パレット：すでに定義された用語は一覧画面から取り出して記述することができ，作業を簡単にしている．
- 単体チェック：1つのドキュメントにおいて多重した定義，矛盾した定義などを発見する機能．
- 相関チェック：複数のドキュメントにおいて多重した定義，矛盾した定義などを発見する機能．
- クロスリファレンス：ドキュメントの一部を修正した場合に，影響がおよぶ範囲を検索する機能．

(2) オープンインターフェース

一般には業務モデリングツールと構築ツールの連携にはオープン・インターフェースと呼ばれるファイルが用いられる．現在AA／BRMODELLERと連携できるのはYPS／BLDRだけであるが，このインターフェースは仕様を公開しており，YPS以外のプログラミング言語との連携，他社ツールへの連携も検討している．

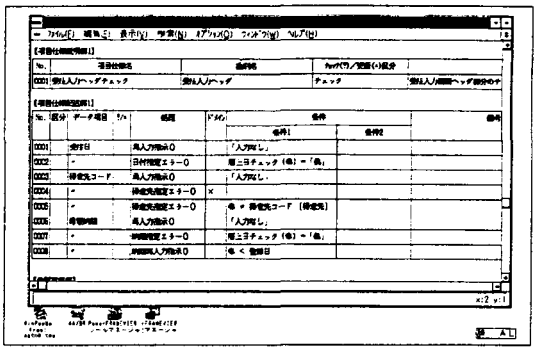

図2　AA／BRMODELLERの画面例

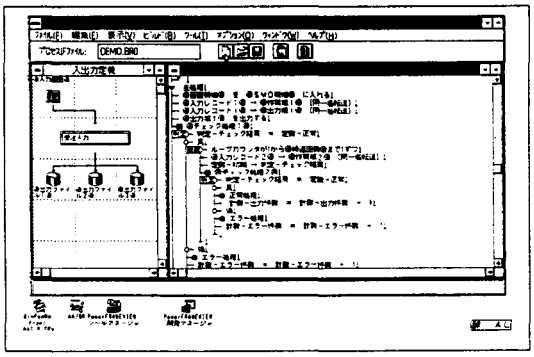

図3　YPS／BLDRの画面例

(3) 構築ツール

AAでは業務仕様は実現手段とは切り離されていることは何度も触れているが，当然業務モデルだけではシステムは完成せず，構築ツールにより制御ロジックに業務の仕様を埋め込んで，プログラムを生成しなければならない．

前章の(5)で述べたように制御ロジックの骨格を制御スケルトンと呼び，現在提供しているYPS/BLDRはYPSで記述された制御スケルトンを対象としている．構築ツールの画面例を図3に示す．

また一般には業務アプリケーションの保守作業の多くは業務仕様の変更によるものである．業務仕様の変更後に構築ツールでプログラムを自動生成するので，そのシステムは常に最新の業務仕様を反映していることになる．

前章の(5)で述べた入出力設計を支援するツールとして“FORM”という構築ツールを利用する．業務アプリケーションではところどころ人間が手作業でディスプレイからデータを入出力する場面がある．そのディスプレイの画面のレイアウトを画面・帳票定義体と呼ぶ．FORMはその画面・帳票定義体を入出力設計にもとづき作成するためのツールである．さらに作成された画面・帳票定義体とアプリケーション・プログラム間のデータの受け渡しを“MeFt”というツールが行なっている．

4．SDAS統合CASEの実施例

これまで富士通の業務基幹系システムの開発方法論であるAAと，AAにもとづくSDAS統合CASE環境について述べてきた．現在AAをシステム開発で適用しているユーザ数は約120ユーザである．この章では，実際のプロジェクトでどのようにSDAS統合CASEが適用され，どのような効果が得られているか，またどのような課題があるかについて実施例を2つあげて述べる．

(1) 実施例1

第1の例としてある貿易会社のプロジェクトをあげる．このプロジェクトは約1.7メガステップの大規模プロジェクトで開発要員は約80人である．開発期間は2年半を用意している．

開発環境はパソコン約80台に，UNIXサーバ機4台を設置し，全機がSDAS統合CASEを利用した開発を行なっている．AAに関する専門SEが2，3人で約80名のフィールドSEにAA適用の指導を行なっている．各開発者は共有資産であるデータ・ディクショナリ，制御スケルトンを専門のグループが設計・管理し，個々の開発者はそれを参照して業務仕様通りのプログラム生成を実施できる．

効果は以下のようになる．

- スケルトンの十分な標準化により，各開発者が担当するプログラムの業務の特殊性に専念できるため，生産性の向上が期待できる．
- CASEによる自動生成により，プログラムの品質が向上し，後に続くテスト工程の短縮が期待できる．

またAAおよびSDAS統合CASEを適用するにあたって以下のような問題が浮かび上がった．

- フィールドSEのAAおよびCASEの理解が不十分．
- 設計された業務仕様をユーザがレビューするが，レビュアーはAAに関する知識があまりなく，CASEドキュメントのレビューが十分できない．
- 大人数の開発のため資産の同期保障（設計書・CASEドキュメントなどの変更をリアルタイムな反映）が難しい．

これらの問題から，CASE適用にあたって，CASEに関する教育の重要性がわかる．優れた技法を開発してもSEやユーザが使いこなせないと十分その技法の効果が出ない．この他CASEツールの操作性などもまだまだ改善の余地があると指摘されている．

(2) 実施例2

第2の例としてある共済業務の例をあげる．このプロジェクトは約140 kステップで，開発期間8カ月，ピーク時で25人が参加する．SDAS統合CASEツール

の適用は参加者全員が初めてであるが，開発作業と平行して勉強会を開くなどしているため効果は十分出ており，

・開発期間は従来よりも25％短縮した．
・テスト工程におけるバグ発生率はほぼ半分以下，レビュー工数率も大幅に減っている．
・従来手法で開発した場合に比べて上流工程の工数の占める割合が高く，全体として平準化された．

といったことがあげられる．

これはCASEツール，特にYPS／BLDRなどの構築ツールにより，BRドキュメントや制御スケルトンなどからプログラムの自動生成ができるようになったためである．

5．むすび

最後にAAおよびSDAS統合CASEを実施するにあたっての利点と問題を整理し，今後の展望を述べたい．

AAではデータ中心による分析・設計により，データの重複の最小化およびデータの格納効率の最適化をする．さらにデータの標準化および業務仕様とコンピュータ制御の分離により，システムの保守性を高めることができる．ここで得られた業務モデルはプラットホームには依存せず，再利用性がある．実現設計においては標準的な制御スケルトンを用意することにより，生産性を高めている．

また業務モデリングツールは業務仕様を表わすドキュメントの作成支援，および論理的な整合性のチェック機能を持っており，構築ツールは業務仕様書とプログラムの整合性を保つ働きがある．このようにSDAS統合CASEはAAにもとづくシステム開発をサポートする．

ただし上で述べたように，実際に適用する際にSEやユーザはAAに対する十分な理解がなければ，効果は出ず，また業務知識そのものも重要であることは言うまでもない．

今後AAにもとづくCASE環境は，ツール間コミュニケーション，ワークフロー・システムなどのグループウェア導入による開発プロセス支援機能，一般文書とCASEドキュメントの統合管理機能などの導入により，いっそう生産性・保守性を高めることが期待される．さらにオブジェクト指向を適用すれば，データと機能をオブジェクトとして一緒にモデル化でき，また業務知識の簡潔な表記により，業務の理解を深めることができる．

参考文献

1）吉岡ほか：業務モデル再利用の実際．ソフトウェア再利用シンポジウム，1992．
2）橋本：システム要求分析技法．情報システムの計画と設計チュートリアル資料，1990．
3）橋本：ソフトウェア開発の自動化技術：CASE．FUJITSU, 44, 2, pp. 153-157 (1993)
4）橋本ほか：SDAS統合CASEによるシステム開発：FUJITSU, 45, 3, pp. 204-210 (1994)