ソフトウェア工学 106－5
（1995．11．17）

通信サービス開発支援環境：ＩＣＡＲＯＳ

青山　幹雄
新潟工科大学　情報電子工学科

ＩＣＡＲＯＳ(Integrated Computer-Aided enviRonment for cOoperative Software development)の設計概念，アーキテクチャ，実現，適用経験を述べる。ＩＣＡＲＯＳは，通信ソフトウェア開発の上流工程である通信サービスの仕様化と設計を支援する統合ＣＡＳＥである。

大規模通信ソフトウェアは長寿命であるため，長期間にわたり機能追加が繰り返される。ＩＣＡＲＯＳは，状態遷移図によるサービス記述を支援する図形エディタとソースコードから状態遷移図を生成するリバースエンジニアリングをリポジトリを介して統合し，機能追加を主体とする開発形態を支援する。現在，大規模通信ソフトウェアの開発，保守に適用している。

本稿では，ＩＣＡＲＯＳの狙いとする開発モデルについて論じ，それを支援するＩＣＡＲＯＳのアーキテクチャを紹介する。特に，状態遷移図エディタとリバースエンジニアリング機能の開発における問題点とアプローチ，実現方法を述べる。最後に，ＩＣＡＲＯＳの実践から得た教訓について述べる。

ＩＣＡＲＯＳ：
An Integrated CASE for Forward and Reverse Engineering of Communication Services

Mikio　Aoyama
Department of Information and Electronics Engineering
Niigata Institute of Technology

This article reports the architecture and development experience of ICAROS (Integrated Computer-Aided enviRonment for cOoperative Software development) which is an integrated CASE(Computer-Aided Software Engineering) for specifications of communication services.

Telecommunication software is long-life and requires repetitive enhancements for years. Integrating the forward-engineering and reverse-engineering makes it possible to seamlessly support the iterative and evolutional development process.

This article first proposes a development model which ICAROS supports, and explains the architecture, features, implementation and evaluation of ICAROS. Finally, lessons learned from the development and practice of ICAROS will be explained.

1．ようこそＩＣＡＲＯＳの世界へ

ソフトウェア開発を支援する様々なＣＡＳＥ(Computer-Aided Software Engineering)が開発されている[Take94]。しかし，それらの多くはビジネスアプリケーションを対象としており，リアルタイムシステムを支援するＣＡＳＥは少ない。

とくに，大規模通信ソフトウェアは長寿命であるため，長期間にわたり機能追加が繰り返される。開発現場から，このような開発形態を支援するＣＡＳＥが求められていた。

通信サービス開発支援環境ＩＣＡＲＯＳ(Integrated Computer-Aided enviRonment for cOoperative Software development)は，このような開発形態を支援するために，状態遷移図によるサービス記述を支援する図形エディタとソースコードから状態遷移図を生成するリバースエンジニアリング[Chik90]を統合した。

本稿では，ＩＣＡＲＯＳの狙いとする開発モデルについて論じ，それを支援するＩＣＡＲＯＳのアーキテクチャを示す。とくに，状態遷移図を対象とするエディタとリバースエンジニアリング機能の開発における問題点とそのアプローチ，ならびに実現方法を述べる。最後に，ＩＣＡＲＯＳの評価と実践から得た教訓について述べる。

2．ＩＣＡＲＯＳのコンセプト

ＩＣＡＲＯＳの開発では，次のような開発モデルを想定した。

(1)成長型開発

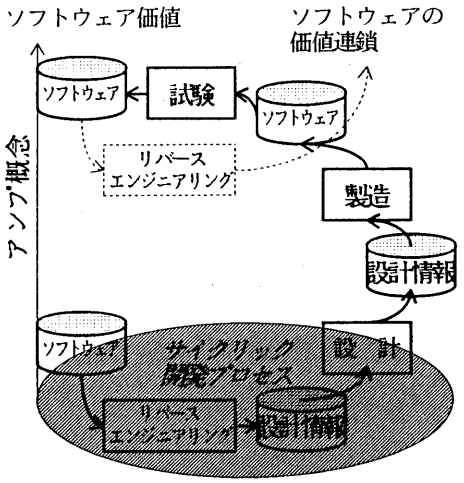
継続的に機能追加する開発形態を『成長型開発』と呼ぶ。成長型開発のモデルとして図－１に示すＣＳＥ(Concurrent, Spiral and Evolutional)開発モデルを提案した[Aoya92c]。このモデルは次の２つの要素から成る。

①アンプ概念[Aoya90]

ソフトウェア開発の主眼はソフトウェアの付加価値を高めることである。ＣＡＳＥは付加価値を生まない作業を担い，かつ付加価値を増大する『アンプ』として捉える。ＩＣＡＲＯＳの開発では自動化による省力化の視点ではなく，開発者の付加価値創造力を増幅する視点を重視した。たとえば，成長型開発では，設計情報の復元が付加価値を生まない作業の中で大きな比重を占めることから，リバースエンジニアリングを重視した。

②サイクリック開発プロセス

図－１に示すように，フォワードエンジニアリングとリバースエンジニアリングを統合した，サイクリックな開発プロセスで捉える。これは，アンプ概念に基づき，開発プロセスをプロダクトの価値連鎖(Value Chain)の面から見ている。ＣＳＥ開発モデルは，並行開発プロセス(Concurrent-Development Process)に組み入れている[Aoya93]。



図－１　ＣＳＥ開発モデル

(2)分散協調型開発

地域分散した複数の開発チームが広域分散処理環境上で協調的に開発を行うための環境として働く[Aoya92a, Aoya92b]。

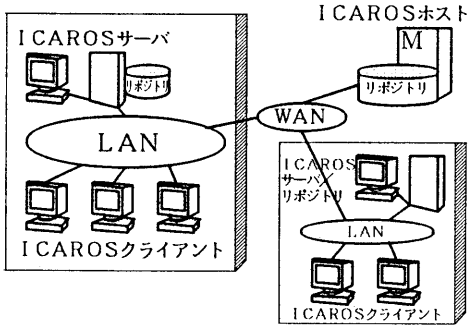
3．ＩＣＡＲＯＳのアーキテクチャ

ＩＣＡＲＯＳのアーキテクチャをオペレーション，ソフトウェア，設計情報の３つの観点から紹介する。

3.1 オペレーションアーキテクチャ

ＩＣＡＲＯＳは，図－２に示す広域クライアント／サーバによる分散処理アーキテクチャをとる。

ＩＣＡＲＯＳを構成する要素システムを表－１に示す。ホストは，旧来の開発環境で蓄積したレガシィデータのリポジトリとして利用している。



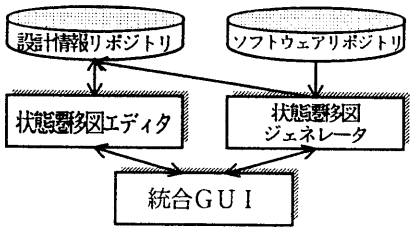
図－２　オペレーションアーキテクチャ

表－１　構成要素

構成要素	システム
クライアント	ＰＣ(DOS/Windows3.1)または ＷＳ(UNIX/X-Windows)
サーバ	ＷＳ(UNIX)
ホスト	メインフレーム

3.2 ソフトウェアアーキテクチャ

ＩＣＡＲＯＳのソフトウェアアーキテクチャを図－３に示す。設計情報を中心に各種ツールを統合した。



図－３　ソフトウェアアーキテクチャ

3.3 設計情報アーキテクチャ

交換サービスは拡張された状態遷移図で記述されている[Taka76]。これは，一つの交換サービスをそれに関わる端末とその間の通話パスなどのオブジェクト群が集団的に状態遷移を行うモデルで表現する。ＩＣＡＲＯＳの開発では，このようなオブジェクト群を階層的に再構造化し，表－２に示す３階層構造とした。内部表現として，テキスト形式の中間言語ＣＩＡＬ(Common Intermediate Abstract Language) を開発した。各階層のオブジェクト，すなわち，システム，サービス，タスクやその構成要素である，ネットワーク，端末とその属性情報はその間の関係と共に，ＣＩＡＬで定義される。

表－２　設計情報アーキテクチャ

情報階層	設計情報の内容	設計担当者
システム	サービスの組合せによるシステムの記述	システムエンジニア
サービス	状態遷移の組合せによるサービスの記述	ソフトウェアエンジニア
タスク	状態遷移処理論理の記述	プログラマ

4．提供機能

ＩＣＡＲＯＳの各サブシステムの狙いと機能を述べる。

(1)状態遷移図エディタ[Ebih90]

状態遷移図の作成と変更，構成管理と版数管理を行い，また，設計情報リポジトリへのアクセスインタフェースも提供する。

(2)状態遷移図ジェネレータ[Yone90]

ソースプログラムからＩＣＡＲＯＳが支援する状態遷移図をＣＩＡＬ形式で生成する。

(3)統合ＧＵＩ[Hash91]

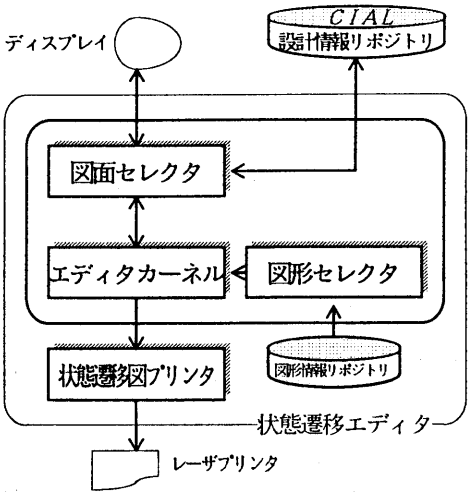
ＩＣＡＲＯＳの各サブシステムと分散したリポジトリを，統一的に操作できるＧＵＩを提供している。このＧＵＩを介して，設計者は必要な設計

情報を分散を意識することなく利用できる。また，関連するツールが自動的に起動されるので，個々のツールやツール間インタフェースを意識する必要がない。

5．状態遷移図エディタ

5.1 状態遷移図エディタのアーキテクチャ

状態遷移図エディタのアーキテクチャを図－4に示す。状態遷移図エディタは，状態遷移図の参照や編集を行うエディタカーネルと，図面セレクタ，図形セレクタ，状態遷移図プリンタから成る。状態遷移図エディタと他の要素間のインタフェースは，テキスト形式の中間言語であるＣＩＡＬを用いる。したがって，状態遷移図ジェネレータなどの他のＩＣＡＲＯＳサブシステムや，ＳＣＣＳなどの既存の開発支援ツールとも連携できる。



図－4　状態遷移図エディタのアーキテクチャ

5.2 状態遷移図エディタの実現

(1)エディタカーネル

状態遷移図の作成や編集を行う。ＩＣＡＲＯＳの対象とする状態遷移図は，各状態が端末などの個別に状態遷移する複数の図形要素から成る複雑な構造をとる。したがって，各要素とその間の関係を管理する必要があるため，要素間の意味的関連を中間言語で保持する。エディタは，要素群として意味のあるまとまりで，複写，移動，削除やラバーバインディングできる。この結果，論理的に矛盾する結果をもたらす編集操作を未然に防止できる。また，対象とする図形要素毎に，サブウインドゥを介して，その属性（名称，パラメータ，遷移の起動イベントなど）を記述できる。

本エディタは，Ｘウインドゥシステム上に独自の図形クラスライブラリをオブジェクト指向に基づきインプリメントしている。クラスライブラリの継承機能を利用して，仕様追加や図形定義の変更も可能である。

(2)図面セレクタ

状態遷移図の構成，変更管理を行う。設計情報リポジトリに登録されている状態遷移図の一覧を常時別ウィンドウ上に表示できるので，編集操作とは独立に名称，版数などの情報に基づき図面を選択できる。複数の図面を同時に開き，図面間でのカット&ペーストもできる。

(3)図形セレクタ

端末記号や分岐記号といった図形要素に関する情報（画面に表示する形状や属性項目）は，図形情報リポジトリで一元管理する。図形要素のセットを組み替えることにより，異なる記号にも対応する。図形セレクタは登録されている図形要素の一覧を表示する。図形入力はこの中から図形要素をマウスで選択し，状態遷移図エディタの編集画面上で位置決めするだけでよい。

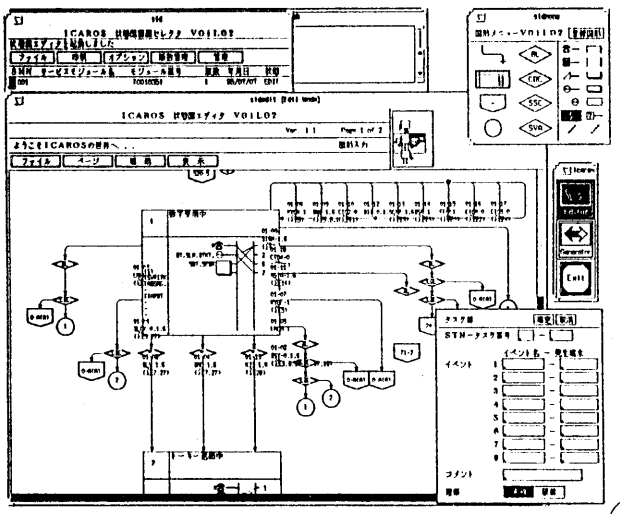
(4)状態遷移図プリンタ

状態遷移図プリンタは，ＣＩＡＬ形式の状態遷移図をPostScript形式に変換し，レーザープリンタで印刷する。

5.3 状態遷移図エディタの使用例

図－5は状態遷移図ジェネレータでソースプログラムから生成した状態遷移図をエディタで編集する画面の例である。中央にエディタのウィンドウがあり，左上隅に状態遷移図セレクタ，右上隅に図形セレクタ（図形メニュー）がある。右下のサブウィンドウは，タスク01-15 の属性を定義す

るためにポップアップしたところである。エディタの右肩の小ウィンドウは1ページ全体のレイアウトと現在表示している部分を示す。この小ウィンドウ上でクリックすると，その位置を中心とした部分がエディタに表示される。



図－5　状態遷移図エディタの使用画面例

6．状態遷移図ジェネレータ

6.1 状態遷移図ジェネレータのアプローチ

(1)設計情報の構造化

状態遷移図の生成に必要な情報を論理構成情報と図形構成情報に分離した。論理構成情報とは状態遷移論理を定義する情報であり，ソースプログラムから抽出できる。一方，図形構成情報は状態の配置などを規定し，生成時にヒューリステックな判断が必要である。これは，簡易に指定できるようなＧＵＩを提供し設計者が状態遷移図生成時に指定することとした。さらに，内部ではテキスト形式の中間言語ＣＩＡＬで定義されるので，通常のテキストエディタでも入力可能である。

(2)論理構成情報の階層化

論理構成情報は表－2の階層構造をとる。

(3)段階的な変換方法

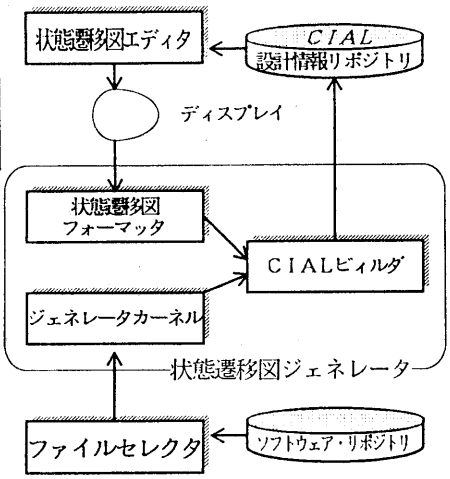
ＣＩＡＬ形式の状態遷移図を生成するために，次のような段階的生成方法をとった。

①ソースプログラムから論理構成情報を生成する。

②論理構成情報と図形構成情報からＣＩＡＬを生成する。

6.2 状態遷移図ジェネレータのアーキテクチャ

状態遷移図ジェネレータのアーキテクチャを図－6に示す。ソースプログラムから状態遷移図のＣＩＡＬ表現を生成する。生成したＣＩＡＬは，エディタを介して状態遷移図として表示すると共に，状態遷移図プリンタで印刷する。



図－6状態遷移図ジェネレータのアーキテクチャ

6.3 状態遷移図ジェネレータの実現

(1)ジェネレータカーネル

ソースプログラムを解析し，論理構成情報を生成する。ソースプログラムの解析部と解析結果から状態遷移情報を生成し状態遷移の解析を行う部分から成る。ソースプログラムの解析部はｙａｃｃ／ｌｅｘを利用している。

(2)状態遷移図フォーマッタ

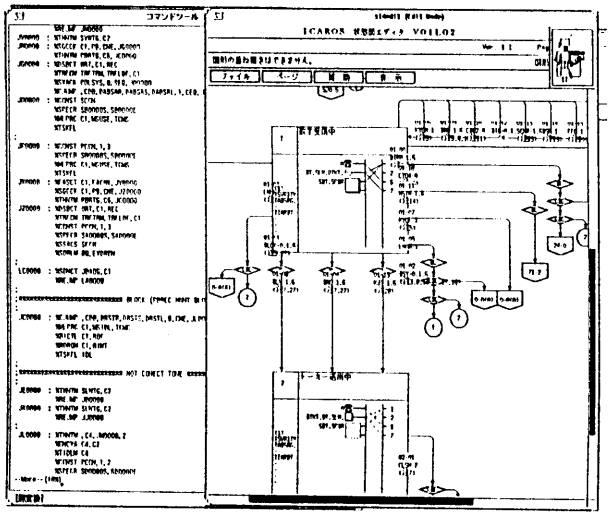
ＧＵＩを介して，設計者に状態遷移図のレイアウトなどの図形構成情報やオブジェクトの属性情報を入力し，ＣＩＡＬビィルダに渡す。

(3)ＣＩＡＬビィルダ

カーネルが生成した論理構成情報とフォーマッタから得られた図形構成情報に基づき，ＣＩＡＬ形式の中間言語を生成する。

6.4 状態遷移図ジェネレータの使用例

図－７に，ソースプログラムと生成した状態遷移図の例を示す。



図－７　状態遷移図ジェネレータの使用画面例

7．評　価

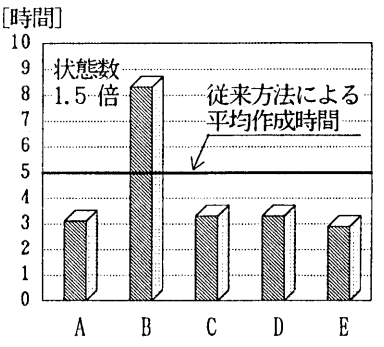
通信ソフトウェアの上流工程支援環境として，リバースエンジニアリングを統合したＣＡＳＥの開発，実践事例はあまり知られていない。この点から，本システムの開発は評価できる。

定量的評価として，状態遷移図エディタとジェネレータの機能，操作性，性能などを評価した。

7.1 状態遷移図エディタの評価

(1)状態遷移図作成効率

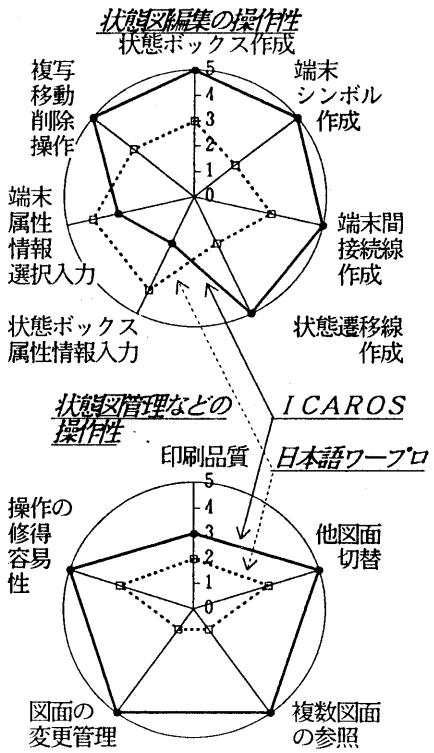
サービスの開発担当者５名（Ａ～Ｅ）が実際の状態遷移図７モジュールを作成した。作成所要時間の測定結果を図－８に示す。従来方法による平均作成時間を５とする。ＩＣＡＲＯＳを用いた場合，作成時間が平均30%短縮した。ただし，Ｂの作成したモジュールは状態数が平均値を50%上回り，かつ遷移構造が複雑であった。



図－８　状態遷移図作成時間の評価

(2)操作性評価

ＩＣＡＲＯＳの操作性と機能を，従来の日本語ワープロと比較し，５段階評価した平均値を図－９に示す。



図－９　状態遷移図エディタの操作性評価

端末シンボル，端末間接続線，状態遷移線の作成など，オブジェクト間の関係を保持した操作の場合，構造化エディタの機能が有効に働いていることが分かる。また，変更管理や複数画面間での参照の面でも設計支援機能の有効性が分かる。一方，属性情報入力に関しては，日本語ＦＥＰの使い勝手の面で評価が悪くなった。

7.2 状態遷移図ジェネレータの評価

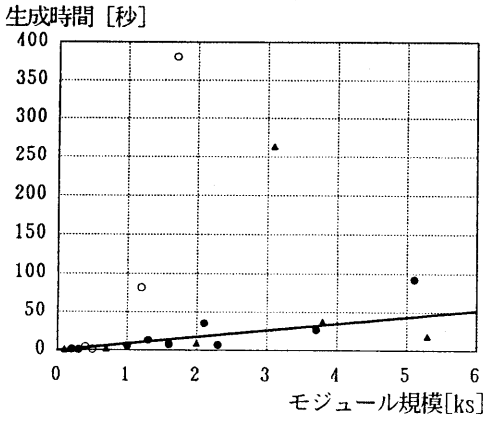
(1)生成率

数百モジュールあるソースプログラムの中で構造的に特に問題がある２モジュールを除き，全て生成可能である。また，生成した状態遷移図は状態とその要素，状態遷移などの項目に関して検証を行い，実用上問題がないと判断している。

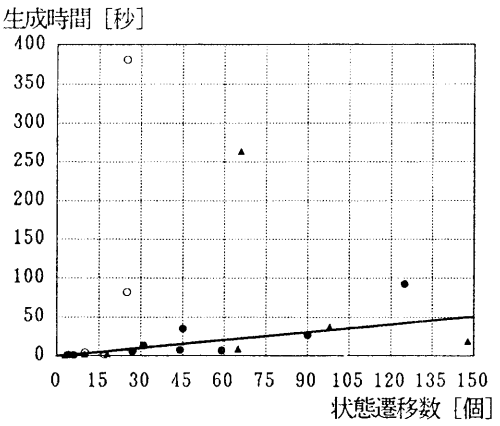
(2)性　能

ソースプログラム規模と状態遷移図の状態遷移数をパラメータとして，代表的モジュールの生成に要したＣＰＵ時間を図－11(a)，(b)に示す。実行マシンはＳＵＮ４／ＩＸである。

50秒未満で95%以上のモジュールの状態遷移図が生成可能であるので性能面でも実用可能と言える。また，ＣＰＵ時間は規模，状態遷移数に比例している。さらに，この傾向から大きく外れるモジュールも幾つかある。これらのモジュールを分析した結果，自己ループの多用やモジュール全体でのループ構造など構造面で特異性が見られた。



図－10(a)　ソースプログラム規模と生成時間



図－10(b)　状態遷移数と生成時間

8．実践からの教訓

ＩＣＡＲＯＳの開発ではリバースエンジニアリング機能の開発が必須要件であり，その開発が実践の成否を握る鍵となった。この経験から，リバースエンジニアリングの開発，適用のあり方について，いくつかの考察を述べよう。

(1)開　発

①アプローチ：知識ベースアプローチとアルゴリズミックアプローチ

ＩＣＡＲＯＳの開発に先立って，通信サービスの設計支援エキスパートシステムの開発を行った[Yama87]。そこでは，知識ベースの保守（追加，変更）が，大きな問題となった。ＩＣＡＲＯＳの開発では，この経験に基づき，むしろアルゴリズミックなアプローチをとった。これは，対象システムを記述するスクリプト言語が意味表現を可能とし，リバースエンジニアリングできる見通しがあったからである。

②表現媒体の物理的制約

ＩＣＡＲＯＳの情報モデルでは，状態や端末などの『物』を単位として情報をカプセル化している。このアプローチは，ポップアップウィンドウなどの表現方法がとれるウィンドウ上での表現に適しているが，紙への印刷はむしろ制約となる。

(2)適　用

①リバースエンジニアリングからリエンジニアリングへ

対象システムが高信頼性を要求されるリアルタイムシステムであるため，適用の鍵は正確な設計情報生成の成否に懸っている。一方，リバースエンジニアリングでは，一般に，ソースプログラムの悪構造が正確な情報の生成を阻害する。ＩＣＡＲＯＳの開発では，ある程度の悪構造は許容したが，信頼性の面でも問題となる悪構造は，ソースプログラムを修正した。結果として，リバースエンジニアリングがシステムのリエンジニアリングを促進した。現在では，ＩＣＡＲＯＳのリバースエンジニアリング機能を通らない構造は設計として認めない方針をとっている。リバースエンジニアリングは，このような設計品質の向上の点からもっと積極的に利用し，評価すべきである。

②開発対象と支援環境

開発対象と支援環境は相対的なものである。とくに，リバースエンジニアリング機能では，環境だけで解決できない問題もある。したがって，プロトタイピングによる事前評価が重要である点を強調しておきたい。ＩＣＡＲＯＳのリバースエンジニアリング機能の開発では，ｙａｃｃ／ｌｅｘでプロトタイピングした。また，対象システムのリエンジニアリングは一朝一夕にはできない。長期的な戦略が重要である。

９．まとめ

ＩＣＡＲＯＳのアーキテクチャとその開発，適用経験を述べた。ＩＣＡＲＯＳは，大規模通信ソフトウェアの保守に適用され，成果を得ている。

ＩＣＡＲＯＳの開発ではリバースエンジニアリング機能の開発が鍵であった。とくに，この開発を通して，リバースエンジニアリングが対象システムのリエンジニアリングへと発展した点を強調しておきたい。

最後に，開発と実践を進めるにあたり協力を頂いた，富士通㈱と富士通北海道通信システム㈱，ならびに適用に協力頂いた関係各位に感謝する。

参考文献

[Aoya90]青山幹雄ほか，“通信ソフトウェアＣＡＳＥ環境ＩＣＡＲＯＳ：構想”，情報処理学会第41回全国大会，Vol. 5, No. 6H-1, Sep. 1990.

[Aoya92a] 青山幹雄，“分散開発環境：新しい開発環境像を求めて”，情報処理，Vol. 33, No. 1, Jan. 1992, pp. 2-13.

[Aoya92b] 青山幹雄ほか，“分散並行開発における設計情報管理環境”，情報処理学会第44回全国大会，No. 1J-2, Mar. 1992.

[Aoya92c] M. Aoyama, et al., “A Distributed Cooperative CASE Environment for Communications Software”, Proc. IEEE COMPSAC '92, Sep. 1992, pp. 102-108.

[Aoya93] M. Aoyama, “Concurrent-Development Process Model”, IEEE Software, Vol. 10, No. 4, Jul. 1993, pp. 48-55.

[Chik90] E. Chikofsky, et al., “Reverse Engineering and Design Recovery”, IEEE Software, Vol. 7, No. 1, Jan. 1990, pp. 13-17.

[Ebih90]蛯原　純ほか，“通信ソフトウェアＣＡＳＥ環境ＩＣＡＲＯＳ：状態図エディタ”，情報処理学会第41回全国大会，Vol. 5, No. 6H-2, Sep. 1990.

[Hash91]橋爪隆典ほか，“通信ソフトウェア開発環境ＩＣＡＲＯＳの統合化”，平成３年度電気関係学会北海道支部連合大会，No. 221, Oct. 1991.

[Taka76]高村真司，川島　浩，電子交換プログラム入門，電子情報通信学会，1976.

[Take94]竹下　亨，ＣＡＳＥ決定版，共立出版，1994.

[Yama87]山崎準一ほか，“呼処理プログラム開発支援システム－プログラム生成”，電子情報通信学会交換研究会，No. SE86-136, Jan. 1987, pp. 49-54.

[Yone90]米津恵美子，青山幹雄，“通信ソフトウェアＣＡＳＥ環境ＩＣＡＲＯＳ：状態図ジェネレータ”，情報処理学会第41回全国大会，Vol. 5, No. 6H-3, Sep. 1990.