

工業用計算機のソフトウェア開発支援オンラインシステム - H O N E S T -

小山 深 田中 裕平
(富士ファコム制御株式会社)

1. はじめに

ハードウェアの技術革新は目覚ましく、計算機の性能および記憶容量は飛躍的に増大し、かつ大巾なコストダウンがなされてきた。これにともない計算機に要求される機能はますます増大していくこととなり、開発すべきソフトウェアの量は急激に増大してきて、この問題は単に量的な問題にとどまらず、品質保証、保守など様々な問題をはらんでいる。これらに対処していくためのソフトウェアエンジニアリングの確立が強く望まれている。

今回、筆者らは工業用計算機の特徴を考慮したうえで、ソフトウェア開発過程における人の思考活動および製作への展開といかにしたか効率よく進めることができるかという観点からこの問題に取り組み、その支援環境として掲題のHONEST(In House On-Line Environment for Software Development)を開発した。これまでHONESTの個々について発表(参考文献参照)してきたが、本論では使用例とともにその全容を報告する。

2. 開発の背景とわらい

工業用計算機システムは複雑にからみあった多様なソフトと多種多様な機器で構成されていて、その信頼性を高め納入先での早期立ち上げをはかるには、ターゲットシステムでの検証は不可欠である。一方、その計算機システムは、制御を主目的とする装置の一端と考えられ、検証や修正のためのツールと格納する余地が少ないことや各種の周辺機器も不十分である。そのためプログラムのテストや修正、結果の確認が効率よく行われていないのが現状

である。さらに開発すべきソフトの増大にともない計算機ネットワークという状況もある。また、開発用ホスト計算機は、オープンバッチ方式で、決して効率のよい使われ方ではなかった。そこで、ホスト計算機(以下ホストと略す。)とターゲット計算機(以下ターゲットと略す。)を有機的に結びつけ、さらに資源を有効に使うソフトウェア開発環境が望まれていた。

HONESTは、ホストとターゲットとネットワーク化し、情報のやりとりを容易にしたうえで、ホストでの作業(プログラムの製作、修正、単体テスト、ターゲットでのテストデータの作成および実行結果や障害情報の出力)とターゲットでの作業(結合テスト)の分担をはかった効率のよいソフトウェア開発環境を提供することとねらいとしている。

3. システム構成

センター計算機室のホスト及試験場に設置されているホスト、ターゲットとローカルターミナルで結合している。システム概要を図1に、ローカルターミナルを図2に示す。

1) ホスト計算機

センター計算機室のホストには、事務フロアに設置されるTSS端末が接続されている。また、ソフトウェア部品や開発中のプログラム、テストデータ、テスト結果、ジョブカタログなどが保存されるデータベースがある。このホストは、プログラムの作成、修正に便利な会話処理機能をもったTSSで構成され、単体テストツール、障害解

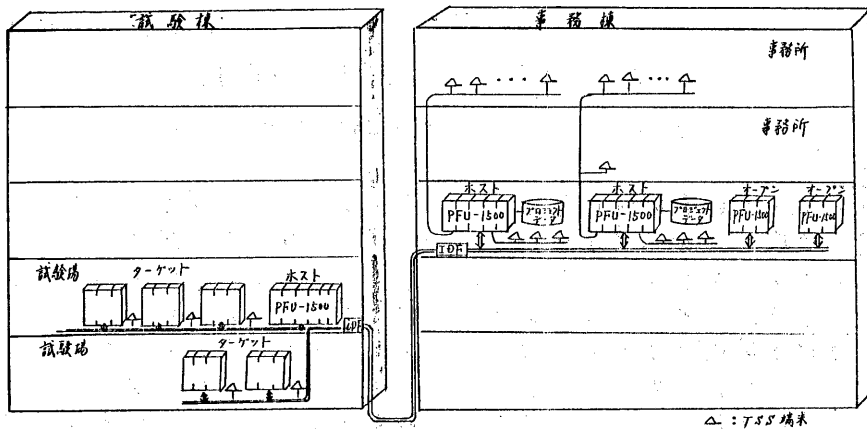


図1. システム概要

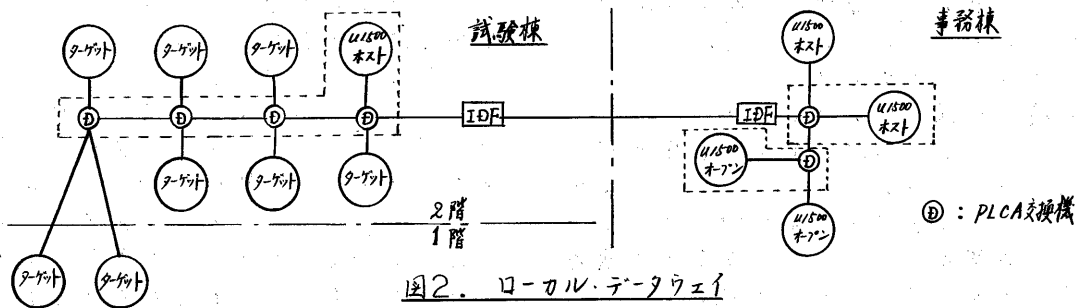


図2. ローカル・データウェイ

新支援ツール、結合テスト支援ツールなども備えている。

試験場のホストには、ターゲットの近くに設置されるTSS端末が接続されている。

2) ターゲット計算機

ターゲットは、それぞれ独自のOSをもっている。実行支援機能として、テスト経過のトレース、障害解析情報の収集と備えている。これらの情報はホストに集められる。

3) ローカルデータウェイ

ローカルデータウェイは、並列回線(PLCA)で構成され、最大400KByte/sの高速伝送が可能である。また、PLCAの交換接続オプションにより最大16台まで接続でき計算機間で任意の対向通信を行うことができる。とともに取り付けが容易である。

4. 機能

HONESTで支援する機能には、次のものがある。そのソフトウェア構成を図3に示す。

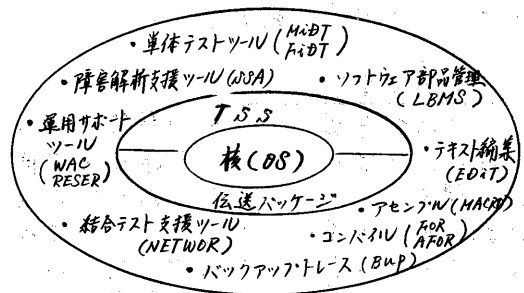


図3. ソフトウェア構成図

1) 単体テストツール

事務フロアに設置された会話端末よりプログラムのテストを行うものである。

る。TSS環境下で動作するため、プログラムの製作、テスト、修正が連続して行える。

単体テストツールには、マクロアセンブラ・プログラムテスト系とフォートラン・プログラムテスト系とがある。

2) 結合テスト支援ツール

ホスト・ターゲット間の情報伝送を行うものである。外部媒体を介さずに直接伝送が行えるため、ターゲットの有効利用、作業の連続化がはかれる。

3) 障害解析支援ツール

ターゲット計算機上でテスト中に発生した障害の解析をホスト上で会話端末から効率よく行うものである。

4) バックアップ・リリース

会話端末上での各種入出カメッセージとファイリングし、後刻の表示、印字と可能にしたものである。また、ファイリングされた情報よりジョブ・カタログも生成することもできる。

4.1 単体テスト環境

1) マクロアセンブラ・プログラムテスト系

本ツールは、アセンブラ言語で作成されたプログラムを会話端末からテストすることを目的とするソースレベルテストツール(MiDT: Macro interactive debugging tool)であり、フリコンバイラ部と実行時動作部から構成されている。概要を図4に示す。

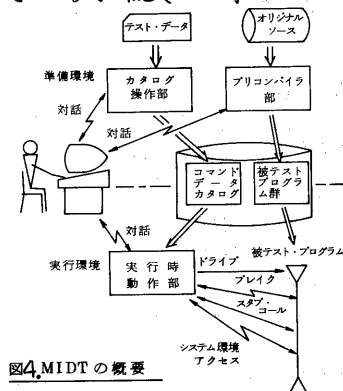


図4. MiDTの概要

① フリコンバイラ部: (iDPHAC)

ブレイクポイントの初期設定、システム環境インタフェース部検出と置換、ドライバ条件設定、スタブ定義生成。

② 実行時動作部: (MiDTRC)

ドライバ、スタブ機能。ブレイクポイントの変更およびブレイクポイントにおける対話型操作機能(データの読込、表示機能等)。バックアップ・リリース。

MiDTでは、全ての操作が原始プログラムの合致用ラベル名やデータ領域名でデバグすることができ、また、実行ルート中の部分的無効化や実行の中断を自由に設定でき、かつプログラムの完成度に応じて段階的にテストすることができる。MiDT (iDPHAC, MiDTRC)の使用例を図5.1、5.2に示す。

00010	TITLE	*TEST PROGRAM*
00020	PROG	TEST,X*07C00*
00030	VECT	X*0000*,A1
00040	DSUB	SYSIN,X*03A0*
00050	DSJB	SYSLST,X*0350*
00060	ENTRY	TEST
00070	*STOP	EQU X*4*
00080	TEST	EQU *
00090	LEA	AREA2,R2
00100	LOOP1	EQU *
00110	CALS	SYS2(R1)
00120	MV	SYSLST
00130	LOOP2	DS TEST
00140	AREA1	DS 2F
00150	AREA2	DS 68F
00340	END	

被テストプログラム

```

0076 //IDPHAC
0077 @ DATASET ID? MIDT.SVCSYS
0078 @ PASSWORD?
0079 S TX DATASET HOLD :831110? :MIDT.SVCSYS
0080 @ (GOOD,NOGOOD)?
0081 @ LABEL TRACE(YES,NO)?
0082 S SUBROUTINE (SYSIN) EXCHANGE DUMY
0083 S SUBROUTINE (SYSLST) EXCHANGE DUMY
0084 =SET AREA2=20,C7(ABCDEFH)J*
0085 =SET R0,A/AREA2+20
0086 =AT 15
0087 =END

```

図5.1 iDPHAC

```

0138 //MIDT
0139 @ TARGET PROGRAM? TEST
0140 @ PROGRAM RUN MODE(BC,EC)? E
0141 @ BACKUP TRACE MODE(GENERAL,NORMAL,DETAIL)?
0142 @ PRINTOUT LUN0?
0143 *** TRACE START : TNO. = 7E16 ***
0144 SYMBOL MODE STNO. C.C ARO AR1 AR2
0145 TEST BREAK 8 Z F9BD6 F0000 F0000
0146 =TY AREA2/X*20
0147 (F9BC2) 0000 0000 0000 0000 0000 0000 0000 0000
0148 (F9BD6) C1C2 C3C4 C5C6 C7C8 C9D1 0000 0000 0000
0149 =SET R1,A/AREA2+20
0150 =G
0151 SYMBOL MODE STNO. C.C ARO AR1 AR2
0152 LOOP1 BREAK 10 Z F9BD6 F9BD6 F9BD6
0153 =G
0154 SYMBOL MODE STNO. C.C ARO AR1 AR2
0155 SYSIN DUMY 11 Z F9BD6 F9BD6 F9BD6
0156 =G
0157 SYMBOL MODE STNO. C.C ARO AR1 AR2
0158 LOOP2 BREAK 13 F F0000 F9BD6 F9BD6

```

図5.2 MiDTRC

2) フォートランプログラムテスト系
 本ツールは、チームリーダーが総合的なプロセス環境の設定を行ない、チームに属するプログラマーはその環境の下で個別なテストケースを設定してプログラムのテストを会話型で行うことと目的としたツール（FIDT: Fortran interactive debugging tool）であり、ファイル模擬系とプロセス入出力模擬系、スタブ模擬系により構成さ

れている。各系は、サブルーチン部とデータ部とサブルーチンの機能やデータの構造と定義する定義部からなり、シミュレータ系全体として木構造としたライブラリを構成している。データ構成を図6に示す。

なお本ツールで使用するプロセスデータの多くは、ターゲットでのデータとしても使用できる。

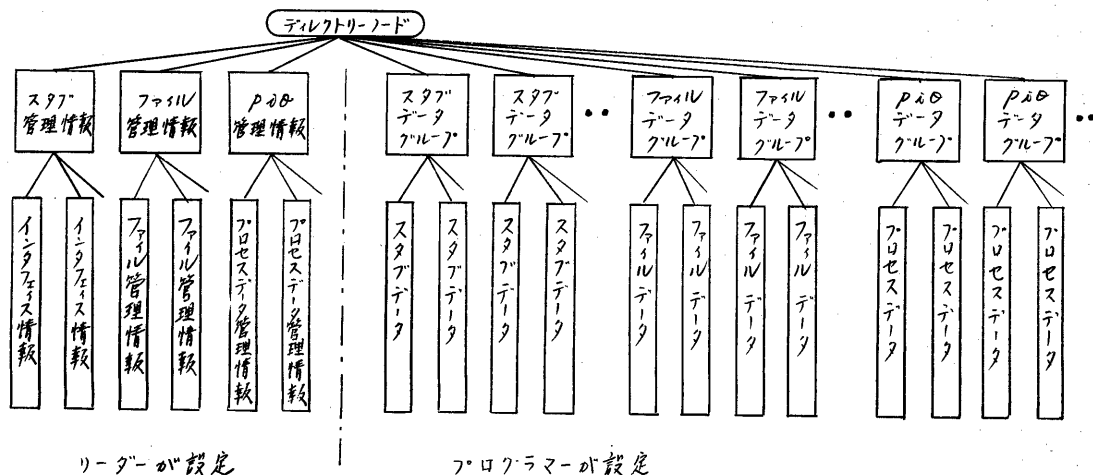


図6. データ構造概略図

次にスタブ模擬系の使用手順、使用例を示す。

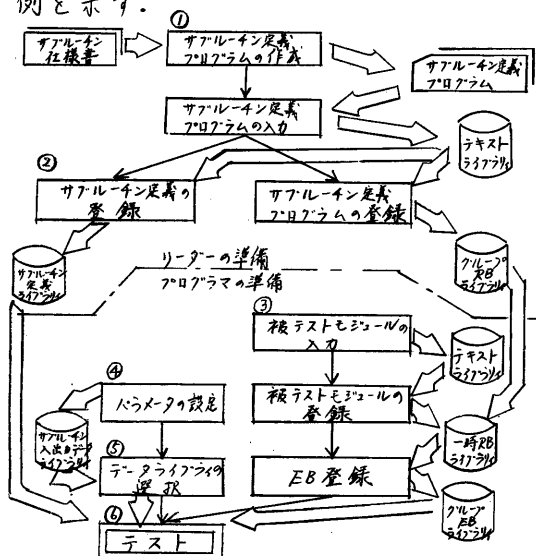


図7.1 スタブテスト手順概要図

```

①
010 C STUB DIVISION
020 C STB111 = * STUB TEST OF INTEGER*2 *
030 C IN1 = IN * IN PARA. OF VARIABLE *
040 C IN2 = IN * IN PARA. OF VARIABLE *
050 C JOUT1 = OUT * OUT PARA. OF VARIABLE *
060 C END STUB
070 C
080 SUBROUTINE STB111( IN1 , IN2 , JOUT1 )
090 CALL STUB ( 'STB111' )
100 RETURN
110 END
  
```

```

②
0109 //LSTUB
0110 @ LEADER PASSWORD?
0111 /DEF
0112 @ STUB ID? TSTB.STB111
0113 @ PASSWORD?
0114 S STUB COMPILE COMPLETE : STB111
  
```

```

③
010 PROGRAM MST111
020 DATA IN1 / 10 /
030 DATA IN2 /100/
040 C
050 DO 100 I=1,3
060 C
070 CALL STB111( IN1 , IN2 , JOUT1 )
080 C
090 100 CONTINUE
100 C
110 STOP
120 END
  
```

```

0012 //PSTUB
0013 /SET
0014 @ GROUP.CASE ID? ACSTB.CASE10
0015 @ PASSWORD?
0016 =QUALIFY STB111
0017 =PRINT IN1
0018 =PRINT IN2
0019 =PRINT JOUT1
0020 =TRACE LIST=DETAIL
0021 =TRACE POINT=ALL
0022 =CNO = ALL
0023 =TRACE
0024 LIST = DETAIL
0025 POINT = ALL
0026 CNO = ALL
0027 1 IN1 (IN)=16, NONCHECK
0028 2 IN2 (IN)=16, NONCHECK
0029 3 JOUT1 (OUT)=16, NONCHECK
0030 =MEMO 1./NO.1 SET -PSTUB /
0031 =SET 1.JOUT1,0
0034 =MEMO 2./NO.2 EXEC /
0035 =END
0036 /END
0037 S RETURN 0000
0038 //SAS
0039 /STB
0040 @ GROUP.CASE ID? ACSTB.CASE10
0041 @ PASSWORD?
0042 S STUB GROUP HOLD : ACSTB CASE10
0043 @ STUB_ASSIGN.(GOOD,NOGOOD)?
0044 S ASSIGN COMPLETE: PSTUB
0045 /END
0046 S RETURN 0000
0047 //TEST
0048 @ PROGRAM ID? ESTBS.MST111
0049 @ PARTITION SIZE (KB)?
0050 /AT 90
0051 /G
0052 * STBS(STUB ) * (STB111) CNO = 1 (ENTRY )
0053 (CNO MEMO) NO.1 SET -PSTUB
0054
0055 ** STUB DUMMY ARGUMENT LIST **
0057 (SUBROUTINE) (DUMMY ARGUMENT)
0058
0059 STB111 ( IN1 , IN2 , JOUT1 )
0060
0061 (MEMO) STUB TEST OF INTEGER=2
0062 (DATE) 83.11.10
0063
0064 * DUMMY ARGUMENT LIST *
0065
0066 (NO.) (NAME) (KIND) (I/O) (TYPE) (DIMENSION)
0067
0068 1 IN1 VARI IN I=2
0069 (MEMO) IN PARA OF VARIABLE
0070
0071 2 IN2 VARI IN I=2
0072 (MEMO) IN PARA OF VARIABLE
0073
0074 3 JOUT1 VARI OUT I=2
0075 (MEMO) OUT PARA OF VARIABLE
0076
0077 (DATA)
0078 IN1 (IN)=10
0079 IN2 (IN)=100
0080 JOUT1 (OUT)=0
0081 * STBS(STUB ) * (STB111) CNO = 1 (EXIT )
0082 (CNO MEMO) NO.1 SET -PSTUB
0083
0084 ** STUB DUMMY ARGUMENT LIST **
0085
0086 (DATA)
0087 IN1 (IN)=10
0088 IN2 (IN)=100
0089 JOUT1 (OUT)=0
0090 =SET IN1,20
0091 =SET IN2,200
0092 =SET JOUT1,1
0093 =PRINT IN1
0094 IN1 (IN)=20
0095 =PRINT IN2
0096 IN2 (IN)=200
0097 =PRINT JOUT1
0098 JOUT1 (OUT)=1
0099 =END
0100 @ (CONTINUE,KILL)? C
0101 S AT 00090 100 IN MST111
0102 /G
0103 * STBS(STUB ) * (STB111) CNO = 2 (ENTRY )
0104 (CNO MEMO) NO.2 EXEC
0105
0106 ** STUB DUMMY ARGUMENT LIST **
0107
0108 (SUBROUTINE) (DUMMY ARGUMENT)
0109
0110 ( IN1 , IN2 , JOUT1 )
0111
0112 (MEMO) STUB TEST OF INTEGER=2
0113 (DATE) 83.11.10
0114
0115 * DUMMY ARGUMENT LIST *
0116
0117 (NO.) (NAME) (KIND) (I/O) (TYPE) (DIMENSION)
0118
0119 1 IN1 VARI IN I=2
0120 (MEMO) IN PARA OF VARIABLE
0121
0122 2 IN2 VARI IN I=2
0123 (MEMO) IN PARA OF VARIABLE
0124
0125 3 JOUT1 VARI OUT I=2
0126 (MEMO) OUT PARA OF VARIABLE
0127
0128 (DATA)
0129 IN1 (IN)=10
0130 IN2 (IN)=100
0131 JOUT1 (OUT)=0

```

```

0132 IN1 (IN)=20
0133 IN2 (IN)=200
0134 JOUT1 (OUT)=1
0135 * STBS(STUB ) * (STB111) CNO = 2 (EXIT )
0136 (CNO MEMO) NO.2 EXEC
0137
0138 ** STUB DUMMY ARGUMENT LIST **
0139
0140 (SUBROUTINE) (DUMMY ARGUMENT)
0141
0142 STB111 ( IN1 , IN2 , JOUT1 )
0143
0144 (MEMO) STUB TEST OF INTEGER=2
0145 (DATE) 83.11.10
0146
0147 * DUMMY ARGUMENT LIST *
0148
0149 (NO.) (NAME) (KIND) (I/O) (TYPE) (DIMENSION)
0150
0151 1 IN1 VARI IN I=2
0152 (MEMO) IN PARA OF VARIABLE
0153
0154 2 IN2 VARI IN I=2
0155 (MEMO) IN PARA OF VARIABLE
0156
0157 3 JOUT1 VARI OUT I=2
0158 (MEMO) OUT PARA OF VARIABLE
0159
0160 (DATA)
0161 IN1 (IN)=10
0162 IN2 (IN)=100
0163 JOUT1 (OUT)=0

```

図7.2 スタッフ使用例

4.2 結合テスト支援ツール

伝送処理概要を図8に示す。

伝送方式の特徴として次の点がある。

- 1) 送信伝文ファイル、受信伝文ファイルは、本構造で、書き込み、削除に対して伸縮性がある。
- 2) 伝文は、優先レベル(コマンドレベル、テキストレベル)もっている。
- 3) 受信伝文ファイルは、タスクごとに持っている。このファイルは、伝文を受けた時に生成され、読み込みが完了した時に削除される。
- 4) 伝送要求には、タスク番号指定とそうでない場合がある。前者の時は受信伝文ファイルに格納後、受信要求SUBとPOSTし、後者の時は、パラメータとして渡される項目番号に対応づけられたプログラムをタスクとして割り付け、受信伝文ファイルに格納する。

次にホストからターゲットへのテキスト伝送例を示す。

```

0200 //NET
0201 /TRA
0202 @ JOB NAME? TESTS
0203 @ JOB MEMO?
0204 @ LB DATASET TYPE(TX,RS,SF)? TX
0205 @ SOURCE CPU NAME? HST1
0206 @ DATASET ID? JOB.COMREG
0207 @ PASSWORD?
0208 @ DESTINATION CPU NAME? KSCEC
0209 @ PASSWORD? DEAD
0210 @ GROUP ID? SOC:C
0211 S NETWORK REQUEST ACCEPT
0212 /END
0213 S RETURN 0000

```

図9. 伝送使用例

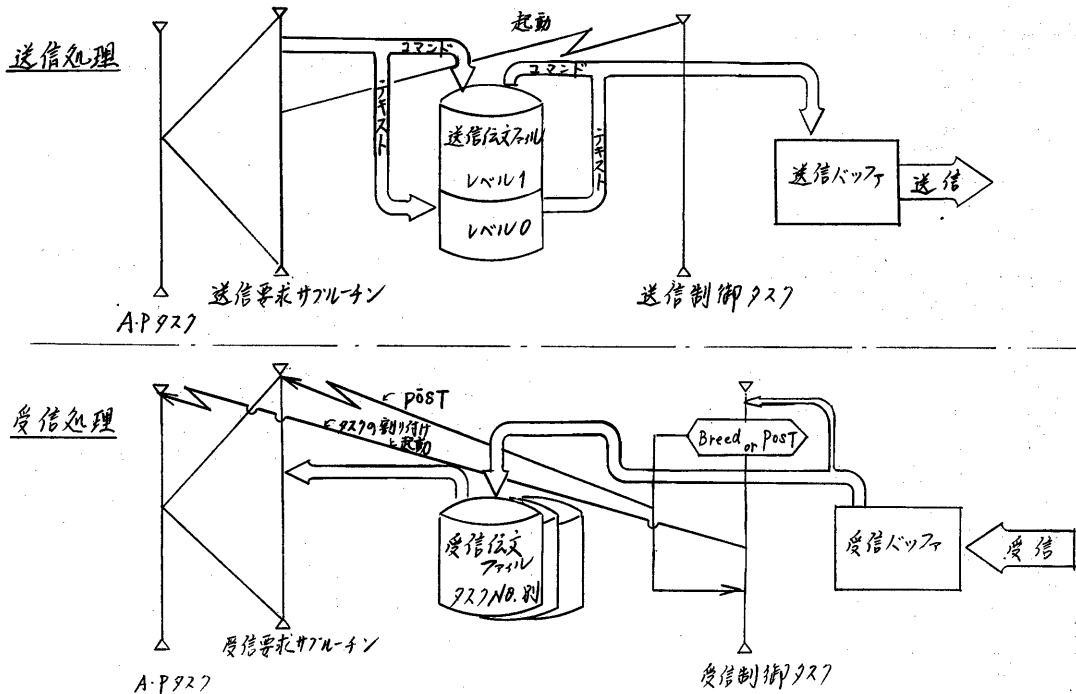


図8. 伝送処理概要図

4.3 障害解析支援ツール

本ツールは、メモリ退避処理（ターゲットの機能）とリンクージ処理（結合テスト支援ツール）、解析支援処理により構成されている。処理構成を図10に示す。

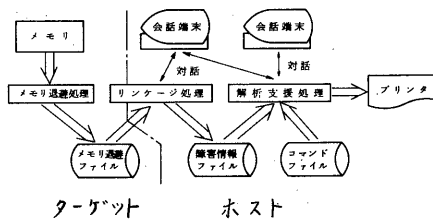


図10. 処理構成図

解析支援処理コマンドは、10種類の基本コマンド（データのリンクと造る、データを表示するなど）と、それらを組み合わせたコマンド列で構成される複合コマンドとからなり、ソフトウェアの複雑な構成情報や障害情報のノ

ウハウを複合コマンドとして蓄積できるようにしている。図11.に使用例を示す。

```
0004. /ISSA
0005 @ FILE ID? YUKO
0006 S FILE HOLD : YUKO ; UAS2(E03) DATE=83.07.
0007 S
0008 @ (G000,NOG00D)?
0009 @ PRINTOUT _UND?
0010 @ TROUBLE DATA(REG,HARD,IC,CMAP,DMAP,OLDS,ZLOG)?
0011 @ TROUBLE DATA(REG,HARD,IC,CMAP,DMAP,OLDS,ZLOG)?
0012 ==CBCSVT,T14
0013 010F4 - 10F4R 1676 1694 16B6 16D8 0000 0000 00
0014 =Z1214,T
0015 01214 - 1214R 1F5E
0016 ==TRBSVT,=JUL,THA?
0017 01DF4 - 1DF4R 1E18 2000 0364 F37F 0000 0000
0018 01E18 - 1E18R 2268 3010 0365 06AF 0000 0000
0019 02258 - 2258R 0000 6006 0390 7303 0036 EE80
0020 =a4POPST.POPSVT
0021 <<< POPSVT >>>
0022 04000 - 4000R 4600 4060 40A0 0000 41CC 0000 00
0023 04010 - 4020R 0000 F4D2 0000 0000 0000 0000 00
0024 04020 - 4030R 0000 F402 0000 0000 0000 F01E 00
0025 04030 - 4040R FE25 0000 0000 0000 0000 0000 40
0026 04040 - 4050R 0000 0000 417A 0000 0000 FF08 F
0027 04050 - 4060R 0000 0000 0000 0000 0000 0000 00
0028 =a4OST.UCBTSFC
0029 UCBT
0030 0171A - 171AR 0003
0031 0171C - 0000R 17EE 0400
0032 01720 - 0004R 1844 0440
0033 01724 - 0008R 1844 0448
0034 UCB
0035 017EC - 00CCR 003E 00A0 8F18 0000 486C 0000 EC
0036 017FC - 00CCR 00FF 1020 1001 8200 0000 9000 00
0037 017FC - 00CCR 00FF 1020 1001 8200 0000 9000 00
```

図11. 障害解析ツール使用例

4.4 バックアップ・トレース

本ツールは、バックアップ・トレース処理（a.バックアップ用ファイルの確保と削除、b.トレース情報の印字、c.入出カメッセのファイリング、d.トレース情報よりジョブカタログ生成）とバックアップ・トレースの開始、停止処理より構成されている。処理概要を図12に示す。

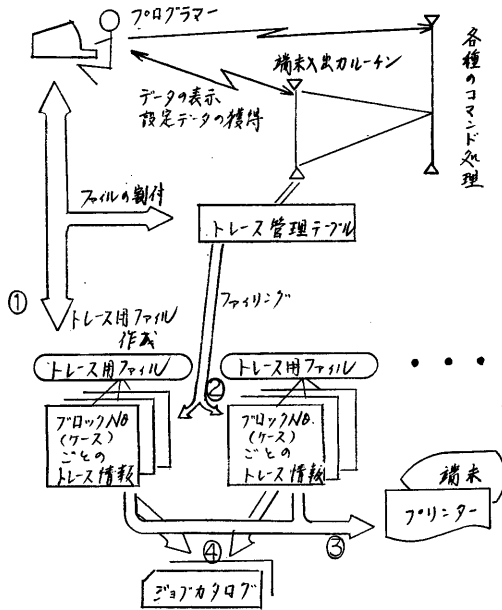


図12. 処理概要図

トレース情報を格納するファイルは、利用者ごとにライブラリとして持つことができ、各ケースごとに格納することができる。バックアップ・トレースの使用例を図13に示す。なお本稿で引用している使用例は、バックアップ・トレースを利用して出力したものである。

①

```

7/PSUP
/REG
@ GROUP ID? SUPTEST
@ PASSWORD?
@ BLOCK MAX?
@ GROUP MEMO? PSUP DEMO
@ BLOCK ID? BUP1
@ BLOCK MEMO? SUP NO.1
S BACK UP FILE REGISTER COMPLETE: SUPTEST :PSUP DEMO
FIRST BLOCK :BUP1 :SUP NO.1
/END
S RETURN 0000

```

②

```

/TSAS
/SUP
@ GROUP ID? BUPTEST
@ PASSWORD?
S ASSIGN COMPLETE: PSUP
/END
S RETURN 0000

```

③

```

//PSUP
/PRINT
@ PRINTOUT LUNO?
@ GROUP ID? SUPTEST
@ PASSWORD?
@ BLOCK ID? BUP1
@ START POINT?
@ BLOCK ID?
@ GROUP ID?
S BACK UP FILE PRINT COMPLETE
/END
S RETURN 0000

```

④

```

//PSUP
/CAT
@ GROUP ID? BUCKUP
@ PASSWORD?
@ BLOCK ID? BLOCK1
@ START POINT? 28
@ END POINT? 34
S JOBCATALOG MAKE COMPLETE
@ BLOCK ID?
@ GROUP ID?
/END
S RETURN 0000

```

図13. バックアップ・トレース使用例

5. 今後のネットワーク計画

現在のHONESTは、ミニコンであるUシリーズを対象としたネットワークシステムであるが、今後は、大型計算機の持つデータベースの有効利用やマイコン、パソコンなどのソフトウェア開発支援環境としての総合的なネットワークシステムにしていく考えである。また、ソフトウェア開発の共通的なOSであるUNIXを搭載したホストの接続を検討している。

6. おわりに

以上述べてきたように、HONESTシステムは、ホストとターゲット間でのプログラミングから結合テストまでの作業工程間の情報の流れと円滑にし、ターゲット上での作業をホスト上で効率よくサポートする各種機能を提供している。これらの機能は、ターゲットシステムの生産性、信頼性の向上に大きく役立ち、利用者からも高い評価が得られてきている。今後さらに、

性能および使い勝手と評価し、操作性のよいシステムにしていくと同時に、各機能の拡充により総合的なソフトウェア開発支援システムへと強化をはかっていく考えである。

最後に、HONESTシステムの開発に際して多大な御指導、御協力を頂いた関係各位に感謝の意を表わす次第である。

〔参考文献〕

- 1) 鶴岡他：工業用計算機のソフトウェア開発支援オンラインシステム—HONESTの開発思想
昭58情学全大7J-4
- 2) 木下他：HONEST単体テスト環境 昭58情学全大5J-2
- 3) 塩谷他：HONEST結合テスト支援環境
昭58情学全大5J-3
- 4) 大星他：対話型ソースレベルテストツール(MiDT)
昭57電学全大1241
- 5) 杉浦他：工業用ソフトウェア障害解析会話型支援ツール
昭58電学全大1350