SOFTWARE ENGINEERING LABORATORY (SEL) DATA BASE REPORTING SOFTWARE USER'S GUIDE AND SYSTEM DESCRIPTION
VOLUME 2: PROGRAM DESCRIPTIONS

(AUGUST 1983)

National Aeronautics and Space Administration

Goddard Space Flight Center
Greenbelt, Maryland 20771
FOREWORD

The Software Engineering Laboratory (SEL) is an organization sponsored by the National Aeronautics and Space Administration, Goddard Space Flight Center (NASA/GSFC) and created for the purpose of investigating the effectiveness of software engineering technologies when applied to the development of applications software. The SEL was created in 1977 and has three primary organizational members:

NASA/GSFC (Systems Development and Analysis Branch)
The University of Maryland (Computer Sciences Department)
Computer Sciences Corporation (Flight Systems Operation)

The goals of the SEL are (1) to understand the software development process in the GSFC environment; (2) to measure the effect of various methodologies, tools, and models on this process; and (3) to identify and then to apply successful development practices. The activities, findings, and recommendations of the SEL are recorded in the Software Engineering Laboratory Series, a continuing series of reports that includes this document. A version of this document was also issued as Computer Sciences Corporation document CSC/SD-82/6083-V1 and -V2.

The primary contributors to this document include

Pei-Shen Lo (Computer Sciences Corporation)
Suellen Eslinger (Computer Sciences Corporation)

Other contributors include

William Decker (Computer Sciences Corporation)

Single copies of this document can be obtained by writing to

Frank E. McGarry
Code 582.1
NASA/GSFC
Greenbelt, Maryland 20771
ABSTRACT

This two-volume document presents the Software Engineering Laboratory (SEL) data base reporting software user's guide and system description. The SEL data base reporting software programs provide formatted listings and summary reports of the SEL data base contents. This document is intended to serve as a reference or tool for the SEL data base administrator, librarians, and programmers and for managers and researchers involved in SEL data base activities. It describes the operating procedures and system information for 18 different reporting software programs.

Volume 1 contains an introduction summarizing the reporting software programs and detailed operating procedures for each program. Sample output reports from each program are also provided. Volume 2 contains descriptions of the structure and functions of each reporting software program. Baseline diagrams, module descriptions, and listings of program generation files are also included.
TABLE OF CONTENTS

VOLUME 1

Section 1 - Introduction .................................. 1-1

1.1 Document Organization ............................... 1-1
1.2 Relationship to the SEL Data Base and DBAM .......... 1-2
1.3 General Overview of SEL Data Base Reporting Software ........................................... 1-4
1.4 Relationship Between the Reporting Software and the SEL Data Base Files ...................... 1-8

Section 2 - User's Guide .................................. 2-1

2.1 Detailed Component Status Report Reporting Program (CS). ........................................... 2-2
  2.1.1 Introduction ....................................... 2-2
  2.1.2 Program Invocation ................................ 2-4
  2.1.3 Program Operation ................................. 2-10
  2.1.4 Sample Output ...................................... 2-12

2.2 Profile Report Program (PF) ............................ 2-24
  2.2.1 Introduction ....................................... 2-24
  2.2.2 Program Invocation ................................ 2-26
  2.2.3 Program Operation ................................. 2-34
  2.2.4 Sample Output ...................................... 2-35

2.3 Resource Utilization Report Program (RU) ............. 2-45
  2.3.1 Introduction ....................................... 2-45
  2.3.2 Program Invocation ................................ 2-46
  2.3.3 Program Operation ................................. 2-48
  2.3.4 Sample Output ...................................... 2-49

2.4 Weekly Hour and Form Count Report Program (WK) ...... 2-54
  2.4.1 Introduction ....................................... 2-54
  2.4.2 Program Invocation ................................ 2-57
  2.4.3 Program Operation ................................. 2-57
  2.4.4 Sample Output ...................................... 2-58

2.5 Component Information Report by Function Type Program (REP4) and Its Preprocessor, the Change and Error Accumulation Program (CG) .... 2-102
  2.5.1 Introduction ....................................... 2-102
  2.5.2 Program Invocation ................................ 2-104
## TABLE OF CONTENTS (Cont'd)

### Section 2 (Cont'd).

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5.3</td>
<td>Program Operation</td>
<td>2-105</td>
</tr>
<tr>
<td>2.5.4</td>
<td>Sample Output</td>
<td>2-106</td>
</tr>
<tr>
<td>2.6</td>
<td>Component Information Report Program (REP5)</td>
<td>2-127</td>
</tr>
<tr>
<td>2.6.1</td>
<td>Introduction</td>
<td>2-127</td>
</tr>
<tr>
<td>2.6.2</td>
<td>Program Invocation</td>
<td>2-128</td>
</tr>
<tr>
<td>2.6.3</td>
<td>Program Operation</td>
<td>2-129</td>
</tr>
<tr>
<td>2.6.4</td>
<td>Sample Output</td>
<td>2-129</td>
</tr>
<tr>
<td>2.7</td>
<td>Graphing Program (GQ)</td>
<td>2-144</td>
</tr>
<tr>
<td>2.7.1</td>
<td>Introduction</td>
<td>2-144</td>
</tr>
<tr>
<td>2.7.2</td>
<td>Program Invocation</td>
<td>2-146</td>
</tr>
<tr>
<td>2.7.3</td>
<td>Program Operation</td>
<td>2-153</td>
</tr>
<tr>
<td>2.7.4</td>
<td>Sample Output</td>
<td>2-153</td>
</tr>
<tr>
<td>2.8</td>
<td>Form Counter Program (NF)</td>
<td>2-158</td>
</tr>
<tr>
<td>2.8.1</td>
<td>Introduction</td>
<td>2-158</td>
</tr>
<tr>
<td>2.8.2</td>
<td>Program Invocation</td>
<td>2-160</td>
</tr>
<tr>
<td>2.8.3</td>
<td>Program Operation</td>
<td>2-160</td>
</tr>
<tr>
<td>2.8.4</td>
<td>Sample Output</td>
<td>2-160</td>
</tr>
<tr>
<td>2.9</td>
<td>SEL Data Base Listing Program (LISTDB)</td>
<td>2-162</td>
</tr>
<tr>
<td>2.9.1</td>
<td>Introduction</td>
<td>2-162</td>
</tr>
<tr>
<td>2.9.2</td>
<td>Program Invocation</td>
<td>2-164</td>
</tr>
<tr>
<td>2.9.3</td>
<td>Program Operation</td>
<td>2-164</td>
</tr>
<tr>
<td>2.9.4</td>
<td>Sample Output</td>
<td>2-166</td>
</tr>
<tr>
<td>2.10</td>
<td>SEL Data Base Recent Activity Report Program (RC)</td>
<td>2-188</td>
</tr>
<tr>
<td>2.10.1</td>
<td>Introduction</td>
<td>2-188</td>
</tr>
<tr>
<td>2.10.2</td>
<td>Program Invocation</td>
<td>2-190</td>
</tr>
<tr>
<td>2.10.3</td>
<td>Program Operation</td>
<td>2-190</td>
</tr>
<tr>
<td>2.10.4</td>
<td>Sample Output</td>
<td>2-190</td>
</tr>
<tr>
<td>2.11</td>
<td>SEL Data Base Record Counting Report Program (RPSTSCTR)</td>
<td>2-193</td>
</tr>
<tr>
<td>2.11.1</td>
<td>Introduction</td>
<td>2-193</td>
</tr>
<tr>
<td>2.11.2</td>
<td>Program Invocation</td>
<td>2-194</td>
</tr>
<tr>
<td>2.11.3</td>
<td>Program Operation</td>
<td>2-195</td>
</tr>
<tr>
<td>2.11.4</td>
<td>Sample Output</td>
<td>2-195</td>
</tr>
</tbody>
</table>
### TABLE OF CONTENTS (Cont'd)

#### Section 2 (Cont'd)

- **2.12 Component Name Report Generator Program**
  - (RPCOMPNM) ........................................ 2-198
    - 2.12.1 Introduction .......................... 2-198
    - 2.12.2 Program Invocation ................... 2-199
    - 2.12.3 Program Operation .................... 2-199
    - 2.12.4 Sample Output .......................... 2-200

- **2.13 Subjective Evaluations File Listing Program**
  - (DBRPTSEF) ........................................ 2-203
    - 2.13.1 Introduction .......................... 2-203
    - 2.13.2 Program Invocation ................... 2-204
    - 2.13.3 Program Operation .................... 2-204
    - 2.13.4 Sample Output .......................... 2-206

- **2.14 Subjective Evaluations Directory File Listing Procedure (DBRPTDIR)** ........................................ 2-211
    - 2.14.1 Introduction .......................... 2-211
    - 2.14.2 Procedure Invocation ................. 2-211
    - 2.14.3 Procedure Operation ................. 2-212
    - 2.14.4 Sample Output .......................... 2-212

- **2.15 Encoding Dictionary Listing Procedure**
  - (DBRPTENC) ........................................ 2-226
    - 2.15.1 Introduction .......................... 2-226
    - 2.15.2 Procedure Invocation ................. 2-226
    - 2.15.3 Procedure Operation ................. 2-226
    - 2.15.4 Sample Output .......................... 2-227

- **2.16 Phase Dates File Listing Procedure (DBRPTHDR)** ........ 2-238
    - 2.16.1 Introduction .......................... 2-238
    - 2.16.2 Procedure Invocation ................. 2-238
    - 2.16.3 Procedure Operation ................. 2-238
    - 2.16.4 Sample Output .......................... 2-239

- **2.17 File Name and Status File Listing Procedure**
  - (DBRPTSTS) ........................................ 2-241
    - 2.17.1 Introduction .......................... 2-241
    - 2.17.2 Procedure Invocation ................. 2-241
    - 2.17.3 Procedure Operation ................. 2-241
    - 2.17.4 Sample Output .......................... 2-242
TABLE OF CONTENTS (Cont'd)

Section 2 (Cont'd)

2.18 Estimated Statistics File Listing Procedure
   (DBRPTEST) ................................................. 2-251
   2.18.1 Introduction ........................................... 2-251
   2.18.2 Procedure Invocation ................................... 2-251
   2.18.3 Procedure Operation .................................... 2-252
   2.18.4 Sample Output ......................................... 2-252

VOLUME 2

Section 3 - System Descriptions ................................. 3-1

3.1 Detailed Component Status Report Reporting
   Program (CS) ................................................. 3-2
      3.1.1 Introduction ......................................... 3-2
      3.1.2 Program Structure .................................... 3-2
      3.1.3 Subroutine/Subsystem Description ...................... 3-5
      3.1.4 Task Build Procedure .................................. 3-19

3.2 Profile Report Program (PF) ................................. 3-28
   3.2.1 Introduction ........................................... 3-28
   3.2.2 Program Structure ....................................... 3-28
   3.2.3 Subroutine/Subsystem Description ....................... 3-29
   3.2.4 Task Build Procedure .................................... 3-40

3.3 Resource Utilization Report Program (RU) .................... 3-49
   3.3.1 Introduction ........................................... 3-49
   3.3.2 Program Structure ....................................... 3-49
   3.3.3 Subroutine/Subsystem Description ....................... 3-50
   3.3.4 Task Build Procedure .................................... 3-66

3.4 Weekly Hour and Form Count Program (WK) .................... 3-73
   3.4.1 Introduction ........................................... 3-73
   3.4.2 Program Structure ....................................... 3-73
   3.4.3 Subroutine/Subsystem Description ....................... 3-74
   3.4.4 Task Build Procedure .................................... 3-92

3.5 Component Information Report by Function Type
   Program (REP4) and Its Preprocessor (CG) .................... 3-102
      3.5.1 Introduction ......................................... 3-102
      3.5.2 Program Structure ..................................... 3-102
# TABLE OF CONTENTS (Cont'd)

Section 3 (Cont'd)

<table>
<thead>
<tr>
<th>3.5.3</th>
<th>Subroutine/Subsystem Description</th>
<th>3-103</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5.4</td>
<td>Task Build Procedure</td>
<td>3-111</td>
</tr>
<tr>
<td>3.6</td>
<td>Component Information Report Program (REP5)</td>
<td>3-121</td>
</tr>
<tr>
<td>3.6.1</td>
<td>Introduction</td>
<td>3-121</td>
</tr>
<tr>
<td>3.6.2</td>
<td>Program Structure</td>
<td>3-121</td>
</tr>
<tr>
<td>3.6.3</td>
<td>Subroutine/Subsystem Description</td>
<td>3-123</td>
</tr>
<tr>
<td>3.6.4</td>
<td>Task Build Procedure</td>
<td>3-127</td>
</tr>
<tr>
<td>3.7</td>
<td>Graphing Program (GQ)</td>
<td>3-133</td>
</tr>
<tr>
<td>3.7.1</td>
<td>Introduction</td>
<td>3-133</td>
</tr>
<tr>
<td>3.7.2</td>
<td>Program Structure</td>
<td>3-133</td>
</tr>
<tr>
<td>3.7.3</td>
<td>Subroutine/Subsystem Description</td>
<td>3-136</td>
</tr>
<tr>
<td>3.7.4</td>
<td>Task Build Procedure</td>
<td>3-149</td>
</tr>
<tr>
<td>3.8</td>
<td>Form Counter Program (NF)</td>
<td>3-158</td>
</tr>
<tr>
<td>3.8.1</td>
<td>Introduction</td>
<td>3-158</td>
</tr>
<tr>
<td>3.8.2</td>
<td>Program Structure</td>
<td>3-158</td>
</tr>
<tr>
<td>3.8.3</td>
<td>Subroutine/Subsystem Description</td>
<td>3-160</td>
</tr>
<tr>
<td>3.8.4</td>
<td>Task Build Procedure</td>
<td>3-169</td>
</tr>
<tr>
<td>3.9</td>
<td>SEL Data Base Listing Program (LISTDB)</td>
<td>3-175</td>
</tr>
<tr>
<td>3.9.1</td>
<td>Introduction</td>
<td>3-175</td>
</tr>
<tr>
<td>3.9.2</td>
<td>Program Structure</td>
<td>3-175</td>
</tr>
<tr>
<td>3.9.3</td>
<td>Subroutine/Subsystem Description</td>
<td>3-176</td>
</tr>
<tr>
<td>3.9.4</td>
<td>Task Build Procedure</td>
<td>3-193</td>
</tr>
<tr>
<td>3.10</td>
<td>SEL Data Base Recent Activity Report Program (RC)</td>
<td>3-199</td>
</tr>
<tr>
<td>3.10.1</td>
<td>Introduction</td>
<td>3-199</td>
</tr>
<tr>
<td>3.10.2</td>
<td>Program Structure</td>
<td>3-199</td>
</tr>
<tr>
<td>3.10.3</td>
<td>Subroutine/Subsystem Description</td>
<td>3-200</td>
</tr>
<tr>
<td>3.10.4</td>
<td>Task Build Procedure</td>
<td>3-206</td>
</tr>
<tr>
<td>3.11</td>
<td>SEL Data Base Record Counting Report Program (RPSTSCTR)</td>
<td>3-212</td>
</tr>
<tr>
<td>3.11.1</td>
<td>Introduction</td>
<td>3-212</td>
</tr>
<tr>
<td>3.11.2</td>
<td>Program Structure</td>
<td>3-212</td>
</tr>
<tr>
<td>3.11.3</td>
<td>Subroutine/Subsystem Description</td>
<td>3-213</td>
</tr>
<tr>
<td>3.11.4</td>
<td>Task Build Procedure</td>
<td>3-215</td>
</tr>
</tbody>
</table>
### TABLE OF CONTENTS (Cont'd)

#### Section 3 (Cont'd)

<table>
<thead>
<tr>
<th>Section</th>
<th>Component Name Report Generator Program (RPCOMPNM)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.12.1</td>
<td>Introduction</td>
<td>3-219</td>
</tr>
<tr>
<td>3.12.2</td>
<td>Program Structure</td>
<td>3-219</td>
</tr>
<tr>
<td>3.12.3</td>
<td>Subroutine/Subsystem Description</td>
<td>3-219</td>
</tr>
<tr>
<td>3.12.4</td>
<td>Task Build Procedure</td>
<td>3-221</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section</th>
<th>Subjective Evaluations File Listing Program (DBRPTSEF)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.13.1</td>
<td>Introduction</td>
<td>3-225</td>
</tr>
<tr>
<td>3.13.2</td>
<td>Program Structure</td>
<td>3-225</td>
</tr>
<tr>
<td>3.13.3</td>
<td>Subroutine/Subsystem Description</td>
<td>3-225</td>
</tr>
<tr>
<td>3.13.4</td>
<td>Task Build Procedure</td>
<td>3-236</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section</th>
<th>Subjective Evaluations Directory File Listing Procedure (DBRPTDIR)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.14.1</td>
<td>Introduction</td>
<td>3-242</td>
</tr>
<tr>
<td>3.14.2</td>
<td>Files Accessed</td>
<td>3-242</td>
</tr>
<tr>
<td>3.14.3</td>
<td>DATATRIEVE Command File</td>
<td>3-242</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section</th>
<th>Encoding Dictionary Listing Procedure (DBRPTENC)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.15.1</td>
<td>Introduction</td>
<td>3-244</td>
</tr>
<tr>
<td>3.15.2</td>
<td>Files Accessed</td>
<td>3-244</td>
</tr>
<tr>
<td>3.15.3</td>
<td>DATATRIEVE Command File</td>
<td>3-244</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section</th>
<th>Phase Dates File Listing Procedure (DBRPTHDR)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.16.1</td>
<td>Introduction</td>
<td>3-246</td>
</tr>
<tr>
<td>3.16.2</td>
<td>Files Accessed</td>
<td>3-246</td>
</tr>
<tr>
<td>3.16.3</td>
<td>DATATRIEVE Command File</td>
<td>3-246</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Section</th>
<th>File Name and Status File Listing Procedure (DBRPTSTS)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.17.1</td>
<td>Introduction</td>
<td>3-248</td>
</tr>
<tr>
<td>3.17.2</td>
<td>Files Accessed</td>
<td>3-248</td>
</tr>
<tr>
<td>3.17.3</td>
<td>DATATRIEVE Command File</td>
<td>3-248</td>
</tr>
</tbody>
</table>

xii
TABLE OF CONTENTS (Cont'd)

Section 3 (Cont'd)

3.18 Estimated Statistics File Listing Procedure
   (DBRPTEST) ............................................. 3-250
   3.18.1 Introduction .................................... 3-250
   3.18.2 Files Accessed ................................. 3-250
   3.18.3 DATATRIEVE Command File ..................... 3-250

References

Bibliography of SEL Literature
# LIST OF ILLUSTRATIONS

## Figure

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>CSR Activity Keywords File.</td>
<td>2-5</td>
</tr>
<tr>
<td>2-2</td>
<td>CS Parameters File.</td>
<td>2-9</td>
</tr>
<tr>
<td>2-3</td>
<td>CS Summary Report.</td>
<td>2-13</td>
</tr>
<tr>
<td>2-4</td>
<td>PF Description File for CIF Profile Report.</td>
<td>2-27</td>
</tr>
<tr>
<td>2-5</td>
<td>PF Description File for CRF Profile Report.</td>
<td>2-28</td>
</tr>
<tr>
<td>2-6</td>
<td>PF Description File for CSF Profile Report.</td>
<td>2-30</td>
</tr>
<tr>
<td>2-7</td>
<td>PF Description File for RAF Profile Report.</td>
<td>2-32</td>
</tr>
<tr>
<td>2-8</td>
<td>CIF Profile Report Program (PF) Output.</td>
<td>2-37</td>
</tr>
<tr>
<td>2-9</td>
<td>CRF Profile Report Program (PF) Output.</td>
<td>2-39</td>
</tr>
<tr>
<td>2-10</td>
<td>CSF Profile Report Program (PF) Output.</td>
<td>2-41</td>
</tr>
<tr>
<td>2-11</td>
<td>RAF Profile Report Program (PF) Output.</td>
<td>2-44</td>
</tr>
<tr>
<td>2-12</td>
<td>RU Input Parameters File.</td>
<td>2-47</td>
</tr>
<tr>
<td>2-13</td>
<td>Resource Utilization Report Program (RU) Output.</td>
<td>2-50</td>
</tr>
<tr>
<td>2-14</td>
<td>Accounting Information Run Count by Week (XW1).</td>
<td>2-60</td>
</tr>
<tr>
<td>2-15</td>
<td>Accounting Information CPU Plus I/O (IBM S/360-95) Hours by Week (XW2).</td>
<td>2-63</td>
</tr>
<tr>
<td>2-16</td>
<td>Accounting Information CPU Plus I/O (IBM S/360-75) Hours by Week (XW3).</td>
<td>2-66</td>
</tr>
<tr>
<td>2-17</td>
<td>Change Report by Week (HW).</td>
<td>2-69</td>
</tr>
<tr>
<td>2-18</td>
<td>Component Status Form Count by Week (TW).</td>
<td>2-72</td>
</tr>
<tr>
<td>2-19</td>
<td>Component Status Hours by Week (TH).</td>
<td>2-75</td>
</tr>
<tr>
<td>2-20</td>
<td>Component Summary Form Count by Week (MW).</td>
<td>2-78</td>
</tr>
<tr>
<td>2-21</td>
<td>Resource Summary (Programmer) Hours by Week (RH1).</td>
<td>2-81</td>
</tr>
<tr>
<td>2-22</td>
<td>Resource Summary (Other) Hours by Week (RH2).</td>
<td>2-84</td>
</tr>
<tr>
<td>2-23</td>
<td>Resource Summary (Computer) Hours by Week (RH3).</td>
<td>2-87</td>
</tr>
<tr>
<td>2-24</td>
<td>Resource Summary Person Count by Week (RP).</td>
<td>2-90</td>
</tr>
<tr>
<td>2-25</td>
<td>Resource Summary Run Count by Week (RR).</td>
<td>2-93</td>
</tr>
<tr>
<td>2-26</td>
<td>Run Analysis Form Count by Week (AW1).</td>
<td>2-96</td>
</tr>
<tr>
<td>2-27</td>
<td>Run Analysis Run Count by Week (AW2).</td>
<td>2-99</td>
</tr>
<tr>
<td>2-28</td>
<td>CG Intermediate File for Project AEM.</td>
<td>2-107</td>
</tr>
<tr>
<td>2-29</td>
<td>Component Information Report by Function Type Program (REP4) Output for Project AEM.</td>
<td>2-110</td>
</tr>
<tr>
<td>2-30</td>
<td>Component Information Report Program (REP5) Output for Project AEM.</td>
<td>2-131</td>
</tr>
<tr>
<td>2-31</td>
<td>External Data File Input to the GQ Program.</td>
<td>2-148</td>
</tr>
<tr>
<td>2-32</td>
<td>GQ Input Parameters File.</td>
<td>2-149</td>
</tr>
<tr>
<td>2-33</td>
<td>GQ Program Output Report.</td>
<td>2-154</td>
</tr>
<tr>
<td>2-34</td>
<td>NF Program Output Report.</td>
<td>2-161</td>
</tr>
<tr>
<td>2-35</td>
<td>CIF LISTDB Report.</td>
<td>2-167</td>
</tr>
<tr>
<td>2-36</td>
<td>CRF File LISTDB Report.</td>
<td>2-168</td>
</tr>
</tbody>
</table>
### LIST OF ILLUSTRATIONS (Cont'd)

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-37</td>
<td>CSF File LISTDB Report</td>
<td>2-170</td>
</tr>
<tr>
<td>2-38</td>
<td>CSR File LISTDB Report</td>
<td>2-177</td>
</tr>
<tr>
<td>2-39</td>
<td>HIS File LISTDB Report</td>
<td>2-179</td>
</tr>
<tr>
<td>2-40</td>
<td>RAF File LISTDB Report</td>
<td>2-180</td>
</tr>
<tr>
<td>2-41</td>
<td>RSF File LISTDB Report</td>
<td>2-183</td>
</tr>
<tr>
<td>2-42</td>
<td>Recent Activity Report Program (RC) Output</td>
<td>2-191</td>
</tr>
<tr>
<td>2-43</td>
<td>SEL Data Base Record Counting Report Program (RPSTSCCTR) Output</td>
<td>2-197</td>
</tr>
<tr>
<td>2-44</td>
<td>RPCOMPNM Program Output Report</td>
<td>2-201</td>
</tr>
<tr>
<td>2-45</td>
<td>Subjective Evaluations File Report Program (DBRPTSEF) Output</td>
<td>2-207</td>
</tr>
<tr>
<td>2-46</td>
<td>Subjective Evaluations Directory File Report Program (DBRPTDIR) Output</td>
<td>2-213</td>
</tr>
<tr>
<td>2-47</td>
<td>Encoding Dictionary File Report Program (DBRPTENC) Output</td>
<td>2-228</td>
</tr>
<tr>
<td>2-48</td>
<td>Phase Dates File Report Program (DBRPTHDR) Output</td>
<td>2-240</td>
</tr>
<tr>
<td>2-49</td>
<td>File Name and Status File Report Program (DBRPTSTS) Output</td>
<td>2-243</td>
</tr>
<tr>
<td>2-50</td>
<td>Estimated Statistics File Report Program (DBRPTEST) Output, Part 1</td>
<td>2-253</td>
</tr>
<tr>
<td>2-51</td>
<td>Estimated Statistics File Report Program (DBRPTEST) Output, Part 2</td>
<td>2-255</td>
</tr>
<tr>
<td>3-1</td>
<td>Baseline Diagram for the Detailed Component Status Report Reporting Program (CS)</td>
<td>3-3</td>
</tr>
<tr>
<td>3-2</td>
<td>CS Task Generation Command Procedure</td>
<td>3-20</td>
</tr>
<tr>
<td>3-3</td>
<td>CS Program Overlay Descriptor Language File</td>
<td>3-27</td>
</tr>
<tr>
<td>3-4</td>
<td>Baseline Diagram for the Profile Report Program (PF)</td>
<td>3-30</td>
</tr>
<tr>
<td>3-5</td>
<td>PF Task Generation Command Procedure</td>
<td>3-41</td>
</tr>
<tr>
<td>3-6</td>
<td>PF Program Overlay Descriptor Language File</td>
<td>3-48</td>
</tr>
<tr>
<td>3-7</td>
<td>Baseline Diagram for the Resource Utilization Report Program (RU)</td>
<td>3-51</td>
</tr>
<tr>
<td>3-8</td>
<td>RU Task Generation Command Procedure</td>
<td>3-67</td>
</tr>
<tr>
<td>3-9</td>
<td>RU Program Overlay Descriptor Language File</td>
<td>3-72</td>
</tr>
<tr>
<td>3-10</td>
<td>Baseline Diagram for the Weekly Hour and Form Count Report Program (WK)</td>
<td>3-75</td>
</tr>
<tr>
<td>3-11</td>
<td>WK Task Generation Command Procedure</td>
<td>3-93</td>
</tr>
<tr>
<td>3-12</td>
<td>WK Program Overlay Descriptor Language File</td>
<td>3-101</td>
</tr>
</tbody>
</table>
LIST OF ILLUSTRATIONS (Cont'd)

Figure

3-13 Baseline Diagram for the Change and Error Accumulation Program (CG) ............. 3-104
3-14 Baseline Diagram for the Component Information Report by Function Type Program (REP4) ................. 3-105
3-15 CG Task Generation Command Procedure .......... 3-112
3-16 REP4 Task Generation Command Procedure ......... 3-114
3-17 CG Program Overlay Descriptor Language File ........................................ 3-119
3-18 REP4 Program Overlay Descriptor Language File ........................................ 3-120
3-19 Baseline Diagram for the Component Information Report Program (REP5) ................. 3-122
3-20 Task Generation Command Procedure for the REP5 Program .................................. 3-128
3-21 REP5 Program Overlay Descriptor Language File ........................................ 3-132
3-22 Baseline Diagram for the Graphing Program (GQ) ........................................ 3-134
3-23 GQ Task Generation Command Procedure ............. 3-150
3-24 GQ Program Overlay Descriptor Language File ........................................ 3-157
3-25 Baseline Diagram for the Form Counter Program (NF) .................................... 3-159
3-26 NF Task Generation Command Procedure ............. 3-170
3-27 NF Program Overlay Descriptor Language File ........................................ 3-174
3-28 Baseline Diagram for the SEL Data Base Listing Program (LISTDB) ....................... 3-177
3-29 LISTDB Task Generation Command Procedure ....... 3-194
3-30 LISTDB Program Overlay Descriptor Language File ........................................ 3-198
3-31 Baseline Diagram for the SEL Data Base Recent Activity Report Program (RC) ........... 3-201
3-32 RC Task Generation Command Procedure ............. 3-207
LIST OF ILLUSTRATIONS (Cont’d)

Figure

3-33  RC Program Overlay Descriptor Language File  ................. 3-211
3-34  Baseline Diagram for the SEL Data Base Record Counting Report Program (RPSTSCTR) 3-214
3-35  RPSTSCTR Task Generation Command Procedure  3-216
3-36  RPSTSCTR Program Overlay Descriptor Language File  ............. 3-218
3-37  Baseline Diagram for the Component Name Report Generator Program (RPCOMPNM) 3-220
3-38  RPCOMPNM Task Generation Command Procedure  3-222
3-39  RPCOMPNM Program Overlay Descriptor Language File  ............. 3-224
3-40  Baseline Diagram for the Subjective Evaluations File Listing Program (DBRPTSEF) 3-226
3-41  DBRPTSEF Task Generation Command Procedure  3-237
3-42  DBRPTSEF Program Overlay Descriptor Language File  ............. 3-241
3-43  DBRPTDIR DATATRIEVE Command File  ............... 3-243
3-44  DBRPTENC DATATRIEVE Command File  ............... 3-245
3-45  DBRPTHDR DATATRIEVE Command File  ............... 3-247
3-46  DBRPTSTS DATATRIEVE Command File  ............... 3-249
3-47  DBRPTEST DATATRIEVE Command File  ............... 3-251

LIST OF TABLES

Table

1-1  Cross-Reference Between Reporting Programs and SEL Data Base Files  ...  1-6
1-2  Relationship Between Data Base Dump Utilities and SEL Data Base Files  ...  1-9
1-3  Cross-Reference Between Summary Reporting Programs and SEL Data Base Files  ...  1-10
SECTION 3 - SYSTEM DESCRIPTION

This section contains the system descriptions for the SEL database reporting programs. The function and structure of each program are presented. All accessed files are described, and, when applicable, baseline diagrams and descriptions of all routines in the program are provided. In addition, the task build procedure is described, including the command files, overlay structure, and required libraries.
3.1 DETAILED COMPONENT STATUS REPORT REPORTING PROGRAM (CS)

3.1.1 INTRODUCTION

The Detailed Component Status Report Reporting Program (CS) produces a report of the Component Status Report (CSR) file for a given project. The program provides a detailed breakdown of programmer hours as reported on the weekly CSR form for a given project (Section 2.1).

3.1.2 PROGRAM STRUCTURE

3.1.2.1 Files Accessed

The CS program accesses seven input files and two output files as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,6]CSR.NL</td>
<td>CS parameters file</td>
</tr>
<tr>
<td>[204,6]CSR.KEY</td>
<td>CSR activity keywords file</td>
</tr>
<tr>
<td>[204,1]ENCODE.HDR</td>
<td>Encoding Dictionary (ENC) file</td>
</tr>
<tr>
<td>[204,1]HEADER.HDR</td>
<td>Phase Dates (HDR) file</td>
</tr>
<tr>
<td>[204,1]EST.HDR</td>
<td>Estimated Statistics (EST) file</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSR</td>
<td>CSR file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CTF</td>
<td>Component Information File (CIF) for the given project</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;PRJNAM&gt;.CS</td>
<td>File containing the detailed CSR report</td>
</tr>
<tr>
<td>FOR010.DAT</td>
<td>File containing a list of all other activity names in file &lt;PRJNAM&gt;.CSR that did not match an activity subcategory name in file CSR.KEY</td>
</tr>
</tbody>
</table>

In these file names, <PRJNAM> is the name of the project selected by the user.

3.1.2.2 Baseline Diagram

Figure 3-1 is the baseline diagram for the CS program. The CSRRPT routine is the main driver. It displays the help
Figure 3-1. Baseline Diagram for the Detailed Component Status Report Reporting Program (CS) (1 of 2)
Figure 3-1. Baseline Diagram for the Detailed Component Status Report Reporting Program (CS) (2 of 2)
information, gets parameter values and other activity keyword values, obtains project and programmer names, and processes the selected CSR data. CSRRPT loops through this process until a^Z (control Z) is returned by the user in response to a prompt.

3.1.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines forming the CS program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major CS routines are described in Section 3.1.3.8. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the CS program also uses the following system routines: DATE, ERRSET, and TIME.

3.1.3.1 Process CSR Data and Compute Statistics

These six major routines read the CSR file and accumulate statistics for the selected project and/or programmers.

ROUTINE: ACC

FUNCTION: Accumulates CSR statistics for a selected programmer

CALLING SEQUENCE:

CALL ACC (ACAT, ANAME, CSRFIL, FCOD, FDES, FTST, KEY, MAXCMP, MAXOTH, NCAT, NL, NNAME, NPROG, PROGCO, PRJNAM, RANGES, SOURCE, TOTFLG, A, ACOL, AOTH, AOTHTT, AROW, ASUBTT, ATOT, C, CCOL, CNAMES, CROW, CSIZE, CSORTX, CTOT, ERROR)
ROUTINE: ASTAT

FUNCTION: Accumulates component and other activity statistics by reading the CSR file

CALLING SEQUENCE:

CALL ASTAT (ANAME, CSRFIL, FCOD, FDES, FRREQ, FTST, KEY, MAXCMP, MAXOTH, NL, PRJNAM, PROGCO, RANGES, SOURCE, TOTFLG, A, AOTH, C, CNAMES, CSIZE, CSORTX, ERROR)

ROUTINE: CSRRPT

FUNCTION: Main routine of the CS program, produces the detailed CSR report

CALLING SEQUENCE: None

ROUTINE: DOCSR

FUNCTION: Processes CSR data by obtaining statistics and writes the CSR output report file

CALLING SEQUENCE:

CALL DOCSR (ACAT, ANAME, CATNAM, CSRFIL, KEY, MAXPRG, NCAT, NL, NNAME, NPROG, PRGCOD, PRGNAM, RANGES, RPTFIL, SOURCE, SUMMARY)

ROUTINE: FRACT

FUNCTION: Reads the CSR file for the given project and computes the fraction of the design, code, and test phases for the given programmer

CALLING SEQUENCE:

CALL FRACT (CSRFIL, NL, PRJNAM, PROGCO, RANGES, TOTFLG, FCOD, FDES, FRREQ, FTST, TFRCOD, TFRDES, TFRTST, ERROR)

3-6

8818
ROUTINE:  SUMOTH

FUNCTION:  Adds the time for a given other name to the appropriate statistics

CALLING SEQUENCE:

CALL SUMOTH (ANAME, FDATE, FCOD, FDES, FRREQ, FTST, KEY, MAXOTH, NL, OTHNAM, OTHOUR, RANGES, SOURCE, A, AOTH, FOUND)

3.1.3.2 Write the CSR Report File

These four routines write the CSR output report file.

ROUTINE:  CMPRPT

FUNCTION:  Prints the report section containing alphabetized component names and corresponding accumulated hours

CALLING SEQUENCE:

CALL CMPRPT (C, CNAME, CSIZE, CSORTX, IPRG, MAXPRG, NL, PRGNAM, PRJNAM, RPTFIL, RPTITL, TOTFLG)

ROUTINE:  FSUMRY

FUNCTION:  Prints a six-line header summary with data from the Phase Dates and Estimated Statistics files

CALLING SEQUENCE:

CALL FSUMRY (IRPTF, PRJNAM)

ROUTINE:  HEADER

FUNCTION:  Prints a one-line title for each report page that includes the date and project name

CALLING SEQUENCE:

CALL HEADER (IRPTF, PRJNAM, RPTITL)
ROUTINE: OTHRPT

FUNCTION: Prints a report of other activity statistics

CALLING SEQUENCE:

CALL OTHRPT (A, ACAT, ACOL, ANAME, AOTH, AROW, ASUBTT, ATOT, CATNAM, IPRG, KEY, MAXOTH, MAXPRG, NCAT, NL, NNAME, PRGNAM, PRJNAM, RPTFIL, RPTITL, TOTFLG)

3.1.3.3 Obtain Data From Terminal or External File

These nine routines obtain information from a user's response to a terminal prompt or from an external file. This information includes input parameters, programmer names, other activity keyword names, and the project name.

ROUTINE: GETFLD

FUNCTION: Displays the given text on the terminal and prompts for a character string

CALLING SEQUENCE:

CALL GETFLD (TEXT, EXTFIL, FLDLEN, TERMNL, EOFTTY, ERROR, FIELD)

ROUTINE: GETNAM

FUNCTION: Gets all CSR programmer codes

CALLING SEQUENCE:

CALL GETNAM (CSRFIL, MAXPRG, PRJNAM, NPROG, PRGCOD, ERROR)

ROUTINE: GETNL

FUNCTION: Reads the sequential parameter file and fills the parameter array
CALLING SEQUENCE:

CALL GETNL (NLDSN, NLFIL, MAXNL, NL, ERROR)

ROUTINE: GETPRG

FUNCTION: Obtains programmer names from the user and converts them to programmer codes from the Encoding Dictionary

CALLING SEQUENCE:

CALL GETPRG (CSRFIL, EXTFIL, MAXPRG, NL, PRJNAM, TERMINL, NPROG, PRGCOD, PRGNAM, SUMARY, EOF, ERROR)

ROUTINE: GTKEYS

FUNCTION: Reads the sequential keywords file to obtain the necessary other activity names and keys for the detailed CSR report

CALLING SEQUENCE:

CALL GTKEYS (KEYFIL, MAXOTH, NL, ACAT, ANAME, CATNAM, KEY, NCAT, NNAME, SOURCE, ERROR)

ROUTINE: FENCA

FUNCTION: Finds the description field on the Encoding Dictionary corresponding to the given type and code

CALLING SEQUENCE:

CALL FENCA (IENCF, TYPE, CODE, NAME, REST, FOUND)

ROUTINE: FENC B

FUNCTION: Finds the description field on the Encoding Dictionary corresponding to the given type and name
CALLING SEQUENCE:

CALL FENCB (IENCF, TYPE, NAME, CODE, REST, FOUND)

ROUTEINE: HELP
FUNCTION: Prints help information to the terminal
CALLING SEQUENCE:

CALL HELP

ROUTEINE: NAME3
FUNCTION: Concatenates the given strings to form a complete file name
CALLING SEQUENCE:

CALL NAME3 (DISK, UIC, NAME, EXTENS, DSN)

3.1.3.4 Sort and Search Routines
These four routines provide some sort and search functions.

ROUTEINE: INSET
FUNCTION: Determines if the given eight-character name is in the given list of names
CALLING SEQUENCE:

CALL INSET (STRING, NAMES, MAXNAM, INDEX, FOUND)

ROUTEINE: PHSCH2
FUNCTION: Determines to which phase the given date belongs
CALLING SEQUENCE:

CALL PHSCH2 (FDATE, RANGES, PHNUM, INPHAS)
ROUTINE: SORTNM

FUNCTION: Produces an array of pointers pointing to the
given name array in alphabetical order

CALLING SEQUENCE:

    CALL SORTNUM (NAMES, NDIM, NUSED, NAMLEN,
                  SORTX)

ROUTINE: STACK2

FUNCTION: Determines whether the given name is in the given
name array, adds it if it is not, and returns the location
of the given name in the given name array

CALLING SEQUENCE:

    CALL STACK2 (ARYMAX, NAME, NAMLEN, NL,
                 ARY, ARYSIZ,
                 LOC, MAXERR)

3.1.3.5 File Open and Read Routines

These nine routines either open an indexed file or read rec-
ords from an indexed file.

ROUTINE: FCIF3

FUNCTION: Reads one record from the CIF using the tertiary
key (component code) and converts all data to internal format

CALLING SEQUENCE:

    CALL FCIF3 (ICIFF, CCODE,
                 PROJNO, CNAME, ICODE, PANV, MODFUN, SYSFUN,
                 ORIGIN, NEXEC, N LINES, NCOMNT, IETA1, IETA2,
                 NETA1, NETA2, NIOVAR, MCCABE, NFUNCT, NIO,
                 NASGN, NCALL, NFMT, STATUS, EOF, ERROR)
ROUTINE: FCSR

FUNCTION: Reads one record from the CSR file using a FORTRAN read

CALLING SEQUENCE:

CALL FCSR (ICSRF, FORMNO, SEQNO, PROJNO, PROGNO, FDATE, COMPCO, TIMES, OTHNAM, OTHOUR, ISTAT, PHASE, EOF, ERROR)

ROUTINE: FCSR3

FUNCTION: Reads one record from the CSR file using the tertiary key (programmer code)

CALLING SEQUENCE:

CALL FCSR3 (ICSRF, PROGCO, FORMNO, SEQNO, PROJNO, PROGNO, FDATE, COMPCO, TIMES, OTHNAM, OTHOUR, ISTAT, PHASE, EOF, ERROR)

ROUTINE: FEST

FUNCTION: Reads one record from the EST file using the secondary key (project name)

CALLING SEQUENCE:

CALL FEST (IESTF, NAME, PROJ, NCOMP, MODDEL, MODNEW, MODMOD, NRUNS, NCHANG, PAGDOC, LINDEL, LINNEW, LINMOD, TOTEXT, NEWEXT, MODEXT, PROGHR, MGMTHR, OTHRHR, HR95, HR75, OTHCMP, STATUS, ACTIVE, PRJCAT, FOUND, ERROR)

ROUTINE: FHDR

FUNCTION: Reads one record from the HDR file using the secondary key (project name)
CALLING SEQUENCE:

CALL FHDR (IHDRF, PRJNAM,
      PROJ, DEVCMP, TARG, ALIEN, RANGES, STATUS,
      ERROR)

ROUTINE: FOPEN
FUNCTION: Opens an indexed file

CALLING SEQUENCE:

CALL FOPEN (IUNIT, FILNAM,
      ERROR)

ROUTINE: FREAD
FUNCTION: Reads one indexed record

CALLING SEQUENCE:

CALL FREAD (IUNIT, KEYVAL, KEYLEN, LRECL,
      BUFFER, ERROR)

ROUTINE: OPENR
FUNCTION: Opens a sequential file for read only

CALLING SEQUENCE:

CALL OPENR (IUNIT, FILNAM, LEN,
      ERROR)

ROUTINE: RDSEQ
FUNCTIONS: Reads one record in a sequential file

CALLING SEQUENCE:

CALL RDSEQ (IUNIT, NCHAR,
      CHARS, EOF)
3.1.3.6 **Routines for String Movement or Comparison**

These eight routines deal with string movement or comparison.

**ROUTINE: BLANK**

**FUNCTION:** Initializes an array to blanks

**CALLING SEQUENCE:**

```
CALL BLANK (ARRAY, NUM)
```

**ROUTINE: CHARGT (LOGICAL FUNCTION)**

**FUNCTION:** Determines if the first string follows the second in alphabetical order

**CALLING SEQUENCE:**

```
CHARGT (STRNG1, STRNG2, LEN)
```

**ROUTINE: CHINT4**

**FUNCTION:** Converts the given character string to an I*4 integer

**CALLING SEQUENCE:**

```
CALL CHINT4 (CHARS, NCHAR, I4NUM, ERROR)
```

**ROUTINE: CHRINT**

**FUNCTION:** Converts the given character string to an I*2 integer

**CALLING SEQUENCE:**

```
CALL CHRINT (CHARS, NCHAR, I2NUM, ERROR)
```
ROUTINE: MATCHS (LOGICAL FUNCTION)
FUNCTION: Determines if the two input strings are the same
CALLING SEQUENCE:
    MATCHS (ARRAY1, ARRAY2, NBYTES)

ROUTINE: MOVE
FUNCTION: Moves a given number of bytes from one address to another
CALLING SEQUENCE:
    CALL MOVE (A, B, LEN)

ROUTINE: V2MOVE
FUNCTION: Copies bytes from one row of a virtual array to a nonvirtual character string
CALLING SEQUENCE:
    CALL V2MOVE (ARY2D, STRING, NROW, DIM1, DIM2)

ROUTINE: WHERE
FUNCTION: Finds the location of the given character in the given string
CALLING SEQUENCE:
    CALL WHERE (CHAR, STRING, LEN, LOC, FOUND)

3.1.3.7 Mathematical Functions
These two routines perform mathematical functions.

ROUTINE: RPCT (REAL FUNCTION)
FUNCTION: Computes a percentage
**CALLING SEQUENCE:**

```
RPCT (I, J)
```

**ROUTINE: SUMR4 (REAL FUNCTION)**

**FUNCTION:** Computes the sum of all numbers in a given array

**CALLING SEQUENCE:**

```
SUMR4 (ARRAY, N)
```

### 3.1.3.8 Variable Description

The variables in the calling sequences of major CS routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A(6,MAXOTH)</td>
<td>R*4</td>
<td>Array containing hours spent on other activities during each phase</td>
</tr>
<tr>
<td>ACAT(MAXOTH)</td>
<td>I*2</td>
<td>Activity name category array to indicate which category the given name belongs to</td>
</tr>
<tr>
<td>ACOL(6)</td>
<td>R*4</td>
<td>Array containing column total of each phase</td>
</tr>
<tr>
<td>ANAME(12,MAXOTH)</td>
<td>L*1</td>
<td>Array containing other activity names</td>
</tr>
<tr>
<td>AOTH(6)</td>
<td>R*4</td>
<td>Array containing hours spent on unknown activities that were not on the list of ANAME for each phase</td>
</tr>
<tr>
<td>AOTHTT</td>
<td>I*2</td>
<td>Not used</td>
</tr>
<tr>
<td>AROW(MAXOTH)</td>
<td>R*4</td>
<td>Array containing total hours spent on each activity</td>
</tr>
<tr>
<td>ARY(NAMLEN, ARYMAX)</td>
<td>L*1</td>
<td>Name array to be searched</td>
</tr>
<tr>
<td>ARYMAX</td>
<td>I*2</td>
<td>Maximum number of names in ARY</td>
</tr>
<tr>
<td>ARYSIZ</td>
<td>I*2</td>
<td>Actual number of names in ARY</td>
</tr>
<tr>
<td>ASUBTT(6,20)</td>
<td>R*4</td>
<td>Array containing total hours spent on each category for each phase</td>
</tr>
<tr>
<td>ATOT</td>
<td>I*2</td>
<td>Not used</td>
</tr>
<tr>
<td>C(9,MAXCMP)</td>
<td>R*4</td>
<td>Array containing hours spent on a component during different phases</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CATNAM(20,20)</td>
<td>L*1</td>
<td>Array containing category names for other activities</td>
</tr>
<tr>
<td>CCOL(3)</td>
<td>I*2</td>
<td>Not used</td>
</tr>
<tr>
<td>CNAMES(8,MAXCMP)</td>
<td>L*1</td>
<td>Array containing component names</td>
</tr>
<tr>
<td>CROW(MAXCMP)</td>
<td>I*2</td>
<td>Not used</td>
</tr>
<tr>
<td>CSIZE</td>
<td>I*2</td>
<td>Total number of components</td>
</tr>
<tr>
<td>CSORTX(MAXCMP)</td>
<td>I*2</td>
<td>Array containing index for sorted component names</td>
</tr>
<tr>
<td>CSRFIL</td>
<td>I*2</td>
<td>FORTRAN unit number for CSR file</td>
</tr>
<tr>
<td>CTOT</td>
<td>I*2</td>
<td>Not used</td>
</tr>
<tr>
<td>EOF</td>
<td>L*1</td>
<td>End-of-file flag</td>
</tr>
<tr>
<td>EOFTTY</td>
<td>L*1</td>
<td>Flag for end of file on terminal</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>EXTFIL</td>
<td>I*2</td>
<td>FORTRAN unit number for reading user input from terminal</td>
</tr>
<tr>
<td>FCOD</td>
<td>R*4</td>
<td>Fraction of time a given programmer spent on coding</td>
</tr>
<tr>
<td>FDATE(3)</td>
<td>I*2</td>
<td>Form date (YY,MM,DD)</td>
</tr>
<tr>
<td>FDES</td>
<td>R*4</td>
<td>Fraction of time a given programmer spent on design</td>
</tr>
<tr>
<td>FIELD(FLDLEN)</td>
<td>L*1</td>
<td>Field to be obtained</td>
</tr>
<tr>
<td>FLDLEN</td>
<td>I*2</td>
<td>Length of field</td>
</tr>
<tr>
<td>FOUND</td>
<td>L*1</td>
<td>Flag indicating a given name is found</td>
</tr>
<tr>
<td>FRREQ(7)</td>
<td>R*4</td>
<td>Fraction of time a given programmer spent on other activities during each phase</td>
</tr>
<tr>
<td>FTST</td>
<td>R*4</td>
<td>Fraction of time a given programmer spent on testing</td>
</tr>
<tr>
<td>INDEX</td>
<td>I*2</td>
<td>Location of a given name within an array of names</td>
</tr>
<tr>
<td>INPHAS</td>
<td>L*1</td>
<td>Flag indicating if a given form date is in any phase</td>
</tr>
<tr>
<td>IPRG</td>
<td>I*2</td>
<td>Current programmer number</td>
</tr>
<tr>
<td>IRPTF</td>
<td>I*2</td>
<td>FORTRAN unit number for CS output report file</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>KEY(MAXOTH)</td>
<td>I*2</td>
<td>Array containing keywords for other activity names</td>
</tr>
<tr>
<td>KEYFIL</td>
<td>I*2</td>
<td>FORTRAN unit number for data set CSR.KEY</td>
</tr>
<tr>
<td>LOC</td>
<td>I*2</td>
<td>Location of a given name in the given name array</td>
</tr>
<tr>
<td>MAXCMP</td>
<td>I*2</td>
<td>Maximum number of components</td>
</tr>
<tr>
<td>MAXERR</td>
<td>L*1</td>
<td>Flag indicating whether the maximum number of components is exceeded</td>
</tr>
<tr>
<td>MAXNAM</td>
<td>I*2</td>
<td>Maximum number of other activity names</td>
</tr>
<tr>
<td>MAXNL</td>
<td>I*2</td>
<td>Maximum number of input parameters</td>
</tr>
<tr>
<td>MAXOTH</td>
<td>I*2</td>
<td>Maximum number of other activity names</td>
</tr>
<tr>
<td>MAXPRG</td>
<td>I*2</td>
<td>Maximum number of programmers</td>
</tr>
<tr>
<td>NAME(NAMLEN)</td>
<td>L*1</td>
<td>Given name to be searched for</td>
</tr>
<tr>
<td>NAMES(NAMLEN, NDIM)</td>
<td>L*1</td>
<td>Name array to be sorted or to be searched</td>
</tr>
<tr>
<td>NAMLEN</td>
<td>I*2</td>
<td>Length of the name</td>
</tr>
<tr>
<td>NCAT</td>
<td>I*2</td>
<td>Number of name categories</td>
</tr>
<tr>
<td>NDIM</td>
<td>I*2</td>
<td>Maximum number of names</td>
</tr>
<tr>
<td>NL(MAXNL)</td>
<td>I*2</td>
<td>Array containing input parameter values</td>
</tr>
<tr>
<td>NLDSN(27)</td>
<td>L*1</td>
<td>Input parameter file name</td>
</tr>
<tr>
<td>NLFIL</td>
<td>I*2</td>
<td>FORTRAN unit number for the input parameter file (CSR.NL)</td>
</tr>
<tr>
<td>NNAME</td>
<td>I*2</td>
<td>Total number of other activity names</td>
</tr>
<tr>
<td>NPROG</td>
<td>I*2</td>
<td>Total number of programmers</td>
</tr>
<tr>
<td>NUSED</td>
<td>I*2</td>
<td>The actual number of names (fill size of NAMES)</td>
</tr>
<tr>
<td>OTHNAM(8)</td>
<td>L*1</td>
<td>Other activity name</td>
</tr>
<tr>
<td>OTHOUR</td>
<td>R*4</td>
<td>Other activity work hours</td>
</tr>
<tr>
<td>PHNUM</td>
<td>I*2</td>
<td>Number of phase containing date</td>
</tr>
<tr>
<td>PRGCOD(MAXPRG)</td>
<td>I*4</td>
<td>Array containing programmer's code</td>
</tr>
<tr>
<td>PRGNAM(8,MAXPRG)</td>
<td>L*1</td>
<td>Array containing programmer's name</td>
</tr>
</tbody>
</table>
### Name | Type | Description
--- | --- | ---
PRJNAM(8) | L*1 | Project name
PROGCO | I*4 | Given programmer's code
RANGES(3,2,7) | I*2 | Start and stop phase dates (YY,MM,DD)
RPTFIL | I*2 | FORTRAN unit number for CS output report file
RPTITL(40) | L*1 | Report title
SORTX(NDIM) | I*2 | Array containing index for sorted names
SOURCE(MAXOTH) | L*1 | Array containing keywords for other activity names
STRING(8) | L*1 | Name string
SUMARY | L*1 | Flag indicating whether a summary report is needed
TERMNL | L*1 | Flag indicating whether terminal or external file is to be read
TEXT(FLDLEN) | L*1 | Prompt text string
TFRCOD | R*4 | Fraction of total time spent on coding
TFRDES | R*4 | Fraction of total time spent on design
TFRTST | R*4 | Fraction of total time spent on testing
TOTFLG | L*1 | Flag indicating whether processing is for all programmers

### 3.1.4 TASK BUILD PROCEDURE

### 3.1.4.1 Command Procedures

The CS program can be generated from the source code by executing the command procedure CSGEN.CMD under UIC [204,6]. This command procedure references three command files--CSFPP.CMD, CSFOR.CMD, and CS.TKB--all under UIC [204,6]. Figure 3-2 is a listing of CSGEN.CMD, the command procedure to precompile, compile, and task build the CS program. The CS program is generated by entering the following command:

```
@[204,6]CSGEN
```
THIS COMMAND PROCEDURE GENERATES THE CS TASK FROM STRUCTURED FORTRAN SOURCE.

PRECOMPILE STRUCTURED FORTRAN SOURCE

This command procedure recompiles all routines which CS program uses. All routines are written in structured Fortran.

ALL ROUTINES WITH PREFIX CS

ROUTINE WITH PREFIX NF

ROUTINES WITH PREFIX UT

Figure 3-2. CS Task Generation Command Procedure (CSGEN.CMD) (1 of 3)
Figure 3-2. CS Task Generation Command Procedure (CSGEN.CMD) (2 of 3)
GENERATE THE CS TASK IMAGE

THIS COMMAND PROCEDURE BUILDS A TASK IMAGE FOR THE DETAILED
COMPONENT STATUS REPORT PROGRAM (CS).

Figure 3-2. CS Task Generation Command Procedure
(CSGEN.CMD) (3 of 3)
3.1.4.2 Overlay Structure

The CS program is overlaid to reduce the memory space requirement. Figure 3-3 is a listing of the Overlay Descriptor Language file, [204, 6]CS.ODL, needed to build the CS program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay.
THE OVERLAY STRUCTURE FOR THE DETAILED COMPONENT STATUS REPORT PROGRAM (CS)

.ROOT $TREE1.OTSALL.RMSALL
$TREE1: .FCTR $ROOT-RMSROT-OTSROT-*(HLP, $NL, $KEY, $PRG, $DO)
$ROOT4: .FCTR [204, 7]UTFMATCHS-$ROOT6
$ROOT6: .FCTR [204, 7]UTFOPENR-[204, 7]UTFREDO -$ROOT8
$HLP: .FCTR [204, 6]CSHELP
$NL: .FCTR [204, 6]CSCRTNL
$KEY: .FCTR [204, 6]CSSGKEYS-[204, 7]UTFCHRINT
$PRG: .FCTR [204, 6]CSSGTPRG-[204, 6]CSSGTPRNT-$PRG2
$PRG2: .FCTR [204, 7]UTFCHRINT
$DO: .FCTR [204, 6]CSSDOCSR-$DOCSR-$DOCSR3-$DOCSR-$DO2
$DO2: .FCTR ($HED, $FR, $ACC, $ORPT, $CRPT)
$HED: .FCTR [204, 7]UTFSUMRY-[204, 7]UTFEXT
$ACC: .FCTR [204, 6]CSSACC-($FR, $AS, $SUM)
$FR2: .FCTR [204, 7]UTFSUM4-[204, 7]UTFCHRINT4-$FR3
$FR3: .FCTR ((204, 7)UTFCSR, [204, 7]UTFCSR3)
$AS2: .FCTR ($INS, $RCSR, $RCSR3, $CIF, $STK, $SORT)
$INS: .FCTR [204, 6]CSSINSET
$RCSR: .FCTR [204, 7]UTFCSR
$RCSR3: .FCTR [204, 7]UTFCSR3
$CIF: .FCTR [204, 7]UTFCFIF3
$STK: .FCTR [204, 6]CSSSTACK2
$SUM: .FCTR [204, 6]NFSUM
$ORPT: .FCTR [204, 6]CSSORPT-[204, 7]UTFPCT
$CRPT: .FCTR [204, 6]CSSCRPT

Figure 3-3. CS Program Overlay Descriptor Language File (CS.ODL)
3.2 PROFILE REPORT PROGRAM (PF)

3.2.1 INTRODUCTION

The Profile Report Program (PF) (or Generalized Response Accumulator Program) produces a cross-tabulation (or profile) report of the entries in various fields of a selected SEL data base file. The program supports the Component Information File (CIF), the Change Report Form (CRF) file, the Component Summary Form (CSF) file, and the Run Analysis Form (RAF) file.

3.2.2 PROGRAM STRUCTURE

3.2.2.1 Files Accessed

The PF program accesses two input files and one or more output files, depending on the file type selected. These files are described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,6]PFNL.XXX</td>
<td>A sequential file containing the PF description file (Section 2.2.2), where XXX = file type (CIF, CRF, CSF, or RAF)</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.XXX</td>
<td>SEL data base file for the given project, where XXX = file type (CIF, CRF, CSF, or RAF)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;PRJNAM&gt;.YNN</td>
<td>The profile report file for the given project, where Y = report type (I, H, M, or A) and NN = breakdown variable number (Section 2.2.3)</td>
</tr>
<tr>
<td>&lt;PRJNAM&gt;.NNY</td>
<td>The plot file for the given project, where Y = report type (I, H, M, or A) and NN = breakdown variable number (Section 2.2.3); produced only for certain file types and breakdown variables (Section 2.2.3)</td>
</tr>
</tbody>
</table>

In these file names, <PRJNAM> is the name of the project selected by the user.
3.2.2.2 Baseline Diagram

Figure 3-4 is the baseline diagram for the PF program. The PROFIL routine is the main driver. It obtains the user's choices for project name, report type, and breakdown category; reads the selected file; accumulates responses; and writes the report. The driver loops through this process until a^Z (control Z) is returned by the user in response to a prompt.

3.2.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines forming the PF program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major PF routines are described in Section 3.2.3.6. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the PF program also uses the following system routines: DATE, ERRSET, ERRSNS, and TIME.

3.2.3.1 Process Data and Accumulate Responses

These six major routines read a given data base file and accumulate responses for the specified profile report.

ROUTINE: GETDAT

FUNCTION: Reads the desired file and accumulates all statistics

CALLING SEQUENCE:

CALL GETDAT (BRKVAR, CATSIZ, DBFILE, FILTYP, IDBF, NCAT, RANGES, RINDEX, RNGCHK, VARNUM, K, KTOT, ERROR)
Figure 3-4. Baseline Diagram for the Profile Report Program (PF)
ROUTINE: LCIF

FUNCTION: Reads one record from the CIF and converts all significant data to an integer value from 1 to N

CALLING SEQUENCE:

CALL LCIF (ICIFF, RANGES, RINDEX, L, NULL, EOF, ERROR)

ROUTINE: LCRF

FUNCTION: Reads one record from the CRF file and converts all significant data to an integer value from 1 to N

CALLING SEQUENCE:

CALL LCRF (ICRFF, RANGES, RINDEX, L, NULL, EOF, ERROR)

ROUTINE: LCSF

FUNCTION: Reads one record from the CSF file and converts all significant data to an integer value from 1 to N

CALLING SEQUENCE:

CALL LCSF (ICSFF, RANGES, RINDEX, L, NULL, EOF, ERROR)

ROUTINE: LRAF

FUNCTION: Reads one record from the RAF file and converts all significant data to an integer value from 1 to N

CALLING SEQUENCE:

CALL LRAF (IRAFF, RANGES, RINDEX, L, NULL, EOF, ERROR)
ROUTINE: PROFIL
FUNCTION: Main routine of the PF program, produces the profile report for the project and file type specified
CALLING SEQUENCE: None

3.2.3.2 Write Output Report and Plot Files
These five routines write the output report and plot files.

ROUTINE: CENTTL
FUNCTION: Centers the character titles
CALLING SEQUENCE:

    CALL CENTTL (NAMES, SBTTLS)

ROUTINE: HEADER
FUNCTION: Prints a one-line title for each report page that includes the date and project name
CALLING SEQUENCE:

    CALL HEADER (IRPTF, PRJNAM, RPTITL)

ROUTINE: HSUMRY
FUNCTION: Prints a six-line header summary with data from the Phase Dates (HDR) and Estimated Statistics (EST) files
CALLING SEQUENCE:

    CALL HSUMRY (IRPTF, PRJNAM)

ROUTINE: PRTDAT
FUNCTION: Prints report data
CALLING SEQUENCE:

    CALL PRTDAT (BRKVAR, CATNAM, CATSIZ, IRPTF, K, KTOT, NCAT, PRJNAM, RANGES, RNGCHK, RPTITL, RPTNAM, STEPS, VARNUM)
ROUTINE: WRTPLT

FUNCTION: Writes profile statistics to a temporary file in preparation for plotting

CALLING SEQUENCE:

CALL WRTPLT (BRKVAR, CATNAM, CATSIZ, FILTYP, K, KTOT, MAKPLT, NCAT, PRJNAM, RPTITL, STEPS, VARNUM)

3.2.3.3 Obtain Data From Terminal or External File

These four routines obtain information from a user's response to a terminal prompt or from an external file.

ROUTINE: GETCOL

FUNCTION: Reads the PF description file to obtain descriptions of fields and categories for the selected profile report

CALLING SEQUENCE:

CALL GETCOL (BRKV, COLFIL,
BRKVAR, CATNAM, CATSIZ, MAKPLT, NCAT, RANGES, RINDEX, RNGCHK, RPTITL, STEPS, VARNUM, ERROR)

ROUTINE: GETFLD

FUNCTION: Displays the given text on the terminal and prompts for a character string

CALLING SEQUENCE:

CALL GETFLD (TEXT, EXTFIL, FLDLEN,
TERMNL, EOFTTY, ERROR
FIELD)
. ROUTINE: GETOPT
FUNCTION: Obtains the project name and user options from
the terminal
CALLING SEQUENCE:

    CALL GETOPT (TERMNL,
                   BRKV, COLFIL, DBFILE, FILTYP, PRJNAM, RPTNAM,
                   EOF, ERROR)

ROUTINE: HELP
FUNCTION: Prints help information to the terminal
CALLING SEQUENCE:

    CALL HELP

ROUTINE: NAME3
FUNCTION: Concatenates the given strings to form a complete
file name
CALLING SEQUENCE:

    CALL NAME3 (DISK, UIC, NAME, EXTENS,
                DSN)

3.2.3.4 File Open and Read Routines
These eight routines either open an indexed file or read
records from an indexed file.
ROUTINE: DOPEN2
FUNCTION: Opens an indexed file
CALLING SEQUENCE:

    CALL DOPEN2 (IFILE, FILNAM,
                  FOUND, ERROR)
ROUTINE: RDCIF
FUNCTION: Reads one record from the CIF and converts all data to internal format
CALLING SEQUENCE:

CALL RDCIF (ICIFF, 
PROJNO, CNAME, ICODE, PANV, MODFUN, SYSFUN, 
ORIGIN, NEXEC, NLINES, NCOMNT, IETA1, IETA2, 
NETAL, NETA2, NIOVAR, MCCABE, NFUNCTION, NIO, 
NASGN, NCALL, NFMT, STATUS, EOF, ERROR)

ROUTINE: RDCRF
FUNCTION: Reads one record from the CRF file and converts the data to internal format
CALLING SEQUENCE:

CALL RDCRF (ICRFF, 
FORMNO, PROJNO, PROGNO, FDATE, NCH, NEXAM, 
OVER1, DATDET, DATBEG, EFFORT, CTYPE, CHCOMP, 
ERRTYP, ERRIN, DATERR, LGCERR, ACTVTY, ISOLTM, 
PATCH, RELOED, RELNO, RELDAT, CMTREA, CMTPES, 
CMTGEN, STATUS, EOF, ERROR)

ROUTINE: RDCSF
FUNCTION: Reads one record from the CSF file
CALLING SEQUENCE:

CALL RDCSF (ICSFF, 
FORMNO, PROJNO, PROGNO, PROG1, FDATE, FSTAGE, 
COMPCO, PRECIS, CMPLEX, SWTYPE, PASGN, PCNTL, 
POTHER, STATWO, STMT, BSIZE, INDEP, RELSW, 
ADDTYP, NCALLD, X1, NICALNG, X2, NSHR, X3, 
NDESC, X4, LANG1, PLANG1, LANG2, PLANG2, DES, 
CONSTR, DESRUN, CODRUN, TSTRUN, DESTIM, 
CODTIM, TSTTIM, DESEFF, CODEFF, TSTEFF, 
DESCAT, CODDAT, TSTDAT, DESCR, CALLD, CALNG, 
SHR, AFFECT, OTH, NAMCON, CMT1, CMT2, ISTAT, 
EOF, ERROR)
ROUTINE: RDEST
FUNCTION: Reads one record from the EST file and converts all data to internal format
CALLING SEQUENCE:

CALL RDEST (IESTF, NAME, PROJ, NCOMP, MODDEL, MODNEW, MODMOD, NRUNS, NCHANGE, PAGDOC, LINDEL, LINNEW, LINMOD, TOTEXT, NEWEXT, MODEXT, PROGHR, MGMTHR, OTHHR, HR95, HR75, OTHCMP, STATUS, ACTIVE, PRJCAT, FOUND, ERROR)

ROUTINE: RDHDRX
FUNCTION: Reads the HDR file and returns the phase dates for a given project
CALLING SEQUENCE:

CALL RDHDRX (IHDF, PROJCT, DRANG1, DRANG2, FOUND)

ROUTINE: RDHDR1
FUNCTION: Reads one record from the HDR file and converts all data to internal format
CALLING SEQUENCE:

CALL RDHDR1 (IHDF, PRJNAV, PROJ, DEVCP, TARG, ALIEN, REQ1, REQ2, DES1, DES2, CODE1, CODE2, SYS1, SYS2, ACC1, ACC2, CLEAN1, CLEAN2, MAINT1, MAINT2, STATUS, FOUND, ERROR)

ROUTINE: RDRAF
FUNCTION: Reads one record from the RAF file
3.2.3.5 Routines for String Movement, Comparison, or Conversion

These eight routines deal with string movement, comparison, or conversion.

ROUTINE: BLANK

FUNCTION: Initializes an array to blanks

CALLING SEQUENCE:

CALL BLANK (ARRAY, NUM)

ROUTINE: CHRINT

FUNCTION: Converts the given string to integer in I*2 format

CALLING SEQUENCE:

CALL CHRINT (CHARS, NCHAR, I2NUM, ERROR)

ROUTINE: CNVRNG

FUNCTION: Converts the given range to character format

CALLING SEQUENCE:

CALL CNVRNG (IBRK, IRNG, RANGES, SUBTTL)

ROUTINE: MATCHS (LOGICAL FUNCTION)

FUNCTION: Determines whether two input strings are the same

CALLING SEQUENCE:

MATCHS (ARRAY1, ARRAY2, NBYTES)
ROUTINE: MOVE
FUNCTION: Moves a given number of bytes from one address to another
CALLING SEQUENCE:
    CALL MOVE (A, B, LEN)

ROUTINE: SQEEZ
FUNCTION: Removes blanks from a character string
CALLING SEQUENCE:
    CALL SQEEZ (IN, NSIZE,
                 NONBL, OUT)

ROUTINE: WHERE
FUNCTION: Finds the location of the given character in the given string
CALLING SEQUENCE:
    CALL WHERE (CHAR, STRING, LEN,
                 LOC, FOUND)

3.2.3.6 Variable Description
The variables in the calling sequences of major PF routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRKV</td>
<td>I*2</td>
<td>Item number of variable desired as breakdown variable</td>
</tr>
<tr>
<td>BRKVAR</td>
<td>I*2</td>
<td>Number of categories in PF description file for the breakdown variable</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CATNAM(25,20)</td>
<td>L*1</td>
<td>Array of field names for each field in PF description file</td>
</tr>
<tr>
<td>CATSIZ(20)</td>
<td>I*2</td>
<td>Number of categories for each field</td>
</tr>
<tr>
<td>COLFIL(27)</td>
<td>L*1</td>
<td>Name of PF description file</td>
</tr>
<tr>
<td>DBFILE(27)</td>
<td>L*1</td>
<td>Data base file name to be read</td>
</tr>
<tr>
<td>EOF</td>
<td>L*1</td>
<td>End-of-file flag</td>
</tr>
<tr>
<td>EOFTTY</td>
<td>L*1</td>
<td>Flag for end of file on terminal</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>EXTFILE</td>
<td>I*2</td>
<td>FORTRAN unit number for external file to be read</td>
</tr>
<tr>
<td>FIELD(FLDLEN)</td>
<td>L*1</td>
<td>Field to be obtained</td>
</tr>
<tr>
<td>FILTYP</td>
<td>L*1</td>
<td>Character indicating type of report desired: I = CIF, H = CRF, M = CSF, A = RAF</td>
</tr>
<tr>
<td>FLDLEN</td>
<td>I*2</td>
<td>Length of field</td>
</tr>
<tr>
<td>IBRK</td>
<td>I*2</td>
<td>Index of boundary of category range to convert (from category description record)</td>
</tr>
<tr>
<td>ICIFF</td>
<td>I*2</td>
<td>FORTRAN unit number of the CIF</td>
</tr>
<tr>
<td>ICRFF</td>
<td>I*2</td>
<td>FORTRAN unit number of the CRF file</td>
</tr>
<tr>
<td>ICSFF</td>
<td>I*2</td>
<td>FORTRAN unit number of the CSF file</td>
</tr>
<tr>
<td>IDBF</td>
<td>I*2</td>
<td>Data base file unit number to be read</td>
</tr>
<tr>
<td>IRAFF</td>
<td>I*2</td>
<td>FORTRAN unit number of the RAF file</td>
</tr>
<tr>
<td>IRNG</td>
<td>I*2</td>
<td>Index of category range to convert (row number of category on report)</td>
</tr>
<tr>
<td>IRPTF</td>
<td>I*2</td>
<td>Profile report file unit number</td>
</tr>
<tr>
<td>K(9,8,20)</td>
<td>I*2</td>
<td>Data array containing all data for profile report except totals</td>
</tr>
<tr>
<td>KTOT(9)</td>
<td>I*2</td>
<td>Array of totals for total column on profile report</td>
</tr>
<tr>
<td>L(55)</td>
<td>I*2</td>
<td>Integer representation of each type of data</td>
</tr>
</tbody>
</table>
### Name | Type | Description
---|---|---
MAKPLT(20) | L*1 | Array of switches indicating whether a plot file is to be produced
NAMES(12,8) | L*1 | Titles
NCAT | I*2 | Number of fields in profile report
NULL | L*1 | Flag indicating if this record is usable
PRJNAM(8) | L*1 | Project name
RANGES(9,55) | I*2 | Range boundaries for all fields identified with asterisks in column 5 of the PF description file
RINDEX(55) | I*2 | Array used for sorting capability (not currently implemented)
RNGCHK(20) | L*1 | Array of flags for each field indicating whether the categories for the field are ranges of values
RPTITL(40) | L*1 | Report title
RPTNAM(27) | L*1 | Report file name
SBTTLS(12,8) | L*1 | Centered titles
STEPS(12,8,20) | L*1 | Array of category names for each field
SUBTTL(12) | L*1 | Array containing column titles
TERMNL | L*1 | Flag indicating whether response is to be read from the terminal or an external file
TEXT(FLDLEN) | L*1 | Prompt text string
VARNUM(20) | I*2 | Item numbers for each field

### 3.2.4 TASK BUILD PROCEDURE

#### 3.2.4.1 Command Procedures

The PF program can be generated from the source code by executing the command procedure PFGEN.CMD under UIC [204,6]. This command procedure references three command files—PFPFP.CMD, PFFOR.CMD, and PF.TKB—all under UIC [204,6]. Figure 3-5 is a listing of PFGEN.CMD, the command procedure
@PFGEN.CMD

THIS COMMAND PROCEDURE PRECOMPiles, COMPILES, AND TASK builds
THE PROFILE REPORT PROGRAM (PF).

PRECOMPile ROUTINES WRITTEN IN STRUCTURED FORTRAN

@[204.6]PFFPP.CMD

@PFFPP.CMD

THIS COMMAND PROCEDURE PRECOMPiles ALL SOURCE CODES WRITTEN IN
STRUCTURED FORTRAN FOR THE PROFILE REPORT PROGRAM (PF).

ROUTINES WITH PREFIX PF

:FP Sy:[204.6]PFCENTTL
:FP Sy:[204.6]PFCNVRNG
:FP Sy:[204.6]PFGETCOL
:FP Sy:[204.6]PFGETDAT
:FP Sy:[204.6]PFGETDPT
:FP Sy:[204.6]PFHELP
:FP Sy:[204.6]PFLCIF
:FP Sy:[204.6]PFLRAF
:FP Sy:[204.6]PFPROFIL
:FP Sy:[204.6]PFPRTDAT
:FP Sy:[204.6]PFWRTPLT

ROUTINES WITH PREFIX UT

:FP Sy:[204.7]UTBLANK
:FP Sy:[204.7]UTCHRINT
:FP Sy:[204.7]UTOOPEN2
:FP Sy:[204.7]UTGETFD
:FP Sy:[204.7]UTGETLEN
:FP Sy:[204.7]UTHEADER
:FP Sy:[204.7]UTSUMRY
:FP Sy:[204.7]UTMATCHS
:FP Sy:[204.7]UTMOVE
:FP Sy:[204.7]UTNAME3
:FP Sy:[204.7]UTRCGIF
:FP Sy:[204.7]UTRCRZ
:FP Sy:[204.7]UTRCSF
:FP Sy:[204.7]UTRDEST
:FP Sy:[204.7]UTRDHDR
:FP Sy:[204.7]UTRDHDR1
:FP Sy:[204.7]UTRDRAF
:FP Sy:[204.7]UTSCZ
:FP Sy:[204.7]UTWHER

COMPilee FORTRAN ROUTINES

Figure 3-5. PF Task Generation Command Procedure (PFGEN.CMD) (1 of 2)
Figure 3-5. PF Task Generation Command Procedure (PFGEN.CMD) (2 of 2)

3-39
to precompile, compile, and task build the PF program. The PF program is generated by entering the following command:

@ [204,6]PFGEN

3.2.4.2 Overlay Structure

The PF program is overlaid to reduce the memory space requirement. Figure 3-6 is a listing of the Overlay Descriptor Language file, [204,6]PF.ODL, needed to build the PF program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay. In addition, the Record Management Service (RMS) Indexed Access Programs Library (RMSIAC) is also needed in the overlay. The name of the library is UFRMSIAC.OLB under UIC [204,7]. It contains FORTRAN routines used for accessing RMS indexed files.
THE OVERLAY DEFINITION FOR THE PROFILE REPORT PROGRAM (PF)

$TREE: .FCTR $TREE1.OTSALL.RMSALL

$TREE1: .FCTR $ROOT-RMSROT-?SROT-*(OPT,SCOL,SDAT,SPRT,SPLT)


$ROOT5: .FCTR [204,7]UFMSIAC/LB


$OPT2: .FCTR [204,7]UTFGETL-[204,7]UTCHRINT

$SCOL: .FCTR [204,6]PPGETCOL-[204,6]PFCVNRNG

$SDAT: .FCTR ($SCIF,$CRF,$CSF,$RAF)


$OUT2: .FCTR [204,7]UTRDFRXT-[204,7]UTRDDEXT-$OUT3

$OUT3: .FCTR [204,6]PDFENTTL-[204,7]UFMSIAC/LB

$SPLT: .FCTR [204,6]PFWRTPLT-[204,7]UTNAME3

$LB:[1,1]RMS11M

@LB:[1,1]RMS12X

END

Figure 3-6. PF Program Overlay Descriptor Language File (PF.ODL)
3.3 RESOURCE UTILIZATION REPORT PROGRAM (RU)

3.3.1 INTRODUCTION

The Resource Utilization Report Program (RU) produces a report of manpower and computer resource data subdivided by phase for a given project. The resource data used are obtained from the Component Status Report (CSR) file and the Resource Summary Form (RSF) file for the given project.

3.3.2 PROGRAM STRUCTURE

3.3.2.1 Files Accessed

The RU program accesses five input files and five output files as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,6]RU.NL</td>
<td>A sequential file containing the key input parameters (a user-defined RU input parameters file under the UIC may be provided instead)</td>
</tr>
<tr>
<td>[204,1]EST.HDR</td>
<td>Estimated Statistics (EST) file</td>
</tr>
<tr>
<td>[204,1]HEADER.HDR</td>
<td>Phase Dates (HDR) file</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSR</td>
<td>CSR file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RSF</td>
<td>RSF file for the given project</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;PRJNAM&gt;.RU</td>
<td>File containing the RU report for the given project</td>
</tr>
<tr>
<td>&lt;PRJNAM&gt;.1RU</td>
<td>First plot file for the given project, containing data from the RSF file (subdivided by phase)</td>
</tr>
<tr>
<td>&lt;PRJNAM&gt;.2RU</td>
<td>First plot file for the given project, containing data from the CSR file (subdivided by phase)</td>
</tr>
<tr>
<td>&lt;PRJNAM&gt;.3RU</td>
<td>Second plot file for the given project, containing data from the RSF file (subdivided by manpower category)</td>
</tr>
</tbody>
</table>
Output File Name | Description
--- | ---
<PRJNAM>.4RU | Second plot file for the given project, containing data from the CSR file (subdivided by manpower category)

In these file names, <PRJNAM> is the name of the project selected by the user. The four plot output files are intended for use by the Pie Chart Plotting Program, which is not currently implemented.

3.3.2.2 Baseline Diagram

Figure 3-7 is the baseline diagram for the RU program. The RU routine is the main driver. It reads the RU input parameters file, the EST file, the HDR file, the RSF file, and the CSR file and prints the resource utilization report. RU loops through the above process until a^Z (control Z) is returned by the user in response to a prompt.

3.3.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines forming the RU program are grouped here by functions. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of the variables begins a new line. The calling sequence variables for the major RU routines are described in Section 3.3.3.6. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the RU program also uses the following system routines: DATE, ERRSNS, and TIME.

3.3.3.1 Process Data and Compute Statistics

These seven major routines obtain data from a given CSR or RSF file and compute statistics for the RU report.
Figure 3-7. Baseline Diagram for the Resource Utilization Report Program (RU) (1 of 2)
Figure 3-7. Baseline Diagram for the Resource Utilization Report Program (RU) (2 of 2)
ROUTINE: FIGRUN

FUNCTION: Computes all data used in the second section of the body of the RU report (computer usage, source code size, and change data)

CALLING SEQUENCE:

CALL FIGRUN (XCOST, LNMULT, XMMM, XMWTMM, HR75, HR95, COMDEL, COMNEW, OLDFAST, XPM, XPTMM, RUNS, XSMM, XSWTMM, LINDEL, LINNEW, T95T75, NCHANG, COSPER, EQU75, E75PER, LP, LPM, LPMS, WTLPM, WTLPM, WTLPM, H75PER, H95PER, NCOMP, RUNPER, SLINES, CHGPER)

ROUTINE: FPHASE

FUNCTION: Computes all necessary phase data

CALLING SEQUENCE:

CALL FPHASE (COSTHR, HRMON, MGHR, MGWT, NWEKS, PROGHR, PRWT, SVHR, SVWT, MHR, MMM, MPCT, MWTHHR, MWTM, MWTPCT, MCOST, MPHSPC, PHR, PPM, PFC, PWTHHR, PWTMM, PWTPCT, PCOST, PPSPC, SHR, SMM, SPCT, SWTHHR, SWTM, SWT, PCOST, SCHSPC, THR, TMM, TCPT, TWTHHR, TWTMM, TWTTPCT, TCOST, TPHSPC, WEEKPC)

ROUTINE: GETCSR

FUNCTION: Obtains programmer hour totals by phase from the CSR file

CALLING SEQUENCE:

CALL GETCSR (CSRFILE, CSRNAME, DRANG1, DRANG2, CPRGHR)

ROUTINE: GETRSF

FUNCTION: Reads all of the RSF file and accumulates programmer, management, and services hours for each phase
CALLING SEQUENCE:

CALL GETRSF (DRANG1, DRANG2, IRSFF, RSFFIL, MGHR, RSFPHR, SVHR, ERROR)

ROUTINE: NEXTWK

FUNCTION: Computes data one week after the given date and returns it in YYMMDD format

CALLING SEQUENCE:

CALL NEXTWK (DATE, D)

ROUTINE: REPORT

FUNCTION: Given the key input parameters and RSF or CSR data, computes and prints percentages and totals

CALLING SEQUENCE:

CALL REPORT (IRPTF, PAGENO, PHRASE, COSTHR, HRMON, LNMULT, MGWT, OLDFAp, PRWT, SVWT, T95T75, DRANG1, DRANG2, MGHR, HR75, HR95, N WEEKS, P HRANG, PROGHR, PROJECT, RUNS, SVHR, COMDEL, COMNEW, LINDEL, LINNEW, NCHANG, NDATWK, TURN)

ROUTINE: RU

FUNCTION: Main routine of the RU program, reads the RSF and CSR files and prints the RU report

CALLING SEQUENCE: None

3.3.3.2 Write the RU Report and Plot Files

These seven routines write the RU output report and plot files.

_ROUTINE: HEADER

FUNCTION: Prints a one-line title for each report page that includes the date, project name, and page number

3-47

8818
CALLING SEQUENCE:

CALL HEADER (IRPTF, PROJCT, PAGENO)

ROUTINE: PAGE1
FUNCTION: Prints abbreviations and notes and key input parameters on the first page of the RU report

CALLING SEQUENCE:

CALL PAGE1 (IRPTF, PAGENO, COSTHR, COSTMM, CSTFIL, HRMON, HRYR, LNMULT, MGWT, NLFIL, OLDFAC, PRWT, PROJCT, RPTFIL, RSFFIL, SVWT, T95T75)

ROUTINE: PRDATA
FUNCTION: Prints Sections 1 and 2 of the body (pages 2 and 3) of the RU report

CALLING SEQUENCE:

CALL PRDATA (IRPTF, PAGENO, PHRASE, PROJCT, DRANG1, DRANG2, MGCST, MGRSF, N WEEKS, PHRANG, PRGCST, PRGRSF, MHR, MMM, MPCT, MWTHR, MWTMM, MWTPCT, MCOST, MPHSPC, PRH, PMM, PPCT, PWTHR, PWTMM, PWTPCT, PCOST, PPHSPC, SHR, SMM, SPCT, SWTHR, SWTMM, SWTPCT, SCOST, SPHSPC, THR, TMM, TPCT, TWTHR, TW TMM, TWTPCT, TCOST, TPHSPC, WEEKPC, COST, COSPER, EQU75, E75PER, LNMULT, LP, LPM, LPMS, WTLT, WTLPM, WTLPMS, HR75, H75PER, HR95, H95PER, NCOMP, RUNS, RUNPER, SLINES, NONCOM, COSP, COSPMS, NCHANG, CHGPER)

ROUTINE: PREXTR
FUNCTION: Prints the third section of the body of the RU report

CALLING SEQUENCE:

CALL PREXTR (IRPTF, NONCOM, COSP, COSPMS)
ROUTINE: PRPHAS

FUNCTION: Prints all data that have been processed according to phase

CALLING SEQUENCE:

CALL PRPHAS (IRPTF, DRANG1, DRANG2, MGCST, MGRSF, NWEKES, PHRANG, PRGCSF, PRGRSF, MHR, MMM, MPCT, MWTHR, MWTMM, WTPCT, COST, MPHSPC, PHR, PMM, PPCT, PWTHR, PWTMM, PWTPCT, PCOST, PPHSPC, SHR, SMM, SPCT, SWTMM, SW1PC, PCOST, PHSPC, TH, TMM, TPCT, TWHHR, TWTHM, TWPCT, TCOST, TPHSPC, WEEKPC)

ROUTINE: PRRUN

FUNCTION: Prints various data related to source lines, computer usage, and changes

CALLING SEQUENCE:

CALL PRRUN (IRPTF, COST, COSPER, EQU75, E75PER, LNMULT, LP, LPM, LPMS, WTLP, WTLPM, WTPMS, HR75, H75PER, HR95, H95PER, NCOMP, RUNS, RUNPER, SLINES, NCHANG, CHGPER)

ROUTINE: WRTPLT

FUNCTION: Writes data to two intermediate files in preparation for pie chart plotting

CALLING SEQUENCE:

CALL WRTPLT (MGMTHR, PRJNAM, PROGHR, SERVHR, TURN)

3.3.3.3 Obtain Data From Terminal or External File

These four routines obtain information from a user's response to a terminal prompt or from an external file.

ROUTINE: GETFLD

FUNCTION: Displays the given text on the terminal and prompts for a character string
CALLING SEQUENCE:

CALL GETFLD (TEXT, EXTFIL, FLDLEN, TERMNL, EOFTTY, ERROR FIELD)

ROUTINE: MAKNAM

FUNCTION: Concatenates the given strings to form a complete file name

CALLING SEQUENCE:

CALL MAKNAM (DISK, UIC, NAME, EXTENS, DSN)

ROUTINE: NAME3

FUNCTION: Concatenates the given strings to form a complete file name

CALLING SEQUENCE:

CALL NAME3 (DISK, UIC, NAME, EXTENS, DSN)

ROUTINE: READNL

FUNCTION: Reads the RU input parameters file

CALLING SEQUENCE:

CALL READNL (INLF, TERMNL, COSTHR, COSTMM, CSTFIL, HRMON, HRYR, LNMULT, MGWI, NLFIL, OLDFAC, PROJ, PRWT, RPTFIL, RSFFIL, SVWT, T95T75, EOF, ERROR)

3.3.3.4 File Open and Read Routines

These seven routines either open an indexed file or read records from an indexed file.
ROUTINE: DDOPNR
FUNCTION: Opens an indexed file
CALLING SEQUENCE:

    CALL DDOPNR (IFILE, FILNAM, FOUND, ERROR)

ROUTINE: DOPEN2
FUNCTION: Opens an indexed file
CALLING SEQUENCE:

    CALL DOPEN2 (IFILE, FILNAM, FOUND, ERROR)

ROUTINE: RDCSR
FUNCTION: Reads one record from the CSR file
CALLING SEQUENCE:

    CALL RDCSR (CSRFIL,
                FORMNO, SEQNO, PROJNO, PROGNO, FDATE, COMPCO,
                TIMES, OTHNAM, OTHOUR, ISTAT, PHASE, EOF, ERROR)

ROUTINE: RDEST
FUNCTION: Reads one record from the EST file and converts all data to internal format
CALLING SEQUENCE:

    CALL RDEST (IESTF, NAME,
                 PROJ, NCOMP, MODDEL, MODNEW, MODMOD, NRUNS,
                 NCHANG, PAGDOC, LINDEL, LINNEW, LINMOD,
                 TOTEXT, NEWEXT, MODEXT, PROGHR, MGMTHR,
                 OTHRHR, HR95, HR75, OTHCMP, STATUS, ACTIVE,
                 PRJCAT, FOUND, ERROR)
ROUTINE: RDHDRX

FUNCTION: Reads the HDR file and returns the phase dates for a given project

CALLING SEQUENCE:

CALL RDHDRX (IHDRF, PROJCT, DRANG1, DRANG2, FOUND)

ROUTINE: RDHDR1

FUNCTION: Reads one record from the HDR file and converts all data to internal format

CALLING SEQUENCE:

CALL RDHDR1 (IHDRF, PRJNAM,
             PROJ, DEVCMP, TARG, ALIEN, REQ1, REQ2, DES1,
             DES2, CODE1, CODE2, SYS1, SYS2, ACC1, ACC2,
             CLEAN1, CLEAN2, MAINT1, MAINT2, STATUS,
             FOUND, ERROR)

ROUTINE: RDRSF

FUNCTION: Reads one record on the RSF file and returns all data on that record plus an array of week dates for each resource entry on the record

CALLING SEQUENCE:

CALL RDRSF (RSFFIL,
             FORMNO, SEQNO, PROJNO, RESCOD, RESID, FDATE,
             PCMGMT, WKDATE, NRUNS, TIMES, STATUS, PHASE,
             LASTWK, EOF, ERROR)

3.3.3.5 Routines for String Movement or Comparison

These eight routines deal with string movement or comparison.

ROUTINE: MATCHS (LOGICAL FUNCTION)

FUNCTION: Determines whether the two input strings are the same
CALLING SEQUENCE:

MATCHS (ARRAY1, ARRAY2, NBYTES)

ROUTINE: MATCH4 (LOGICAL FUNCTION)

FUNCTION: Determines whether a given number is in a given array

CALLING SEQUENCE:

MATCH4 (N, IARRAY, NARRAY)

ROUTINE: MOVE

FUNCTION: Moves a given number of bytes from one address to another

CALLING SEQUENCE:

CALL MOVE (A, B, LEN)

ROUTINE: PHSCHK

FUNCTION: Determines if the given date is within the given date range

CALLING SEQUENCE:

CALL PHSCHK (FDATE, DRANG1, DRANG2, PHNUM, INPHAS)

ROUTINE: SQEEZ

FUNCTION: Removes blanks from a character string

CALLING SEQUENCE:

CALL SQEEZ (IN, NSIZE, NONBL, OUT)
• ROUTINE: WHERE

FUNCTION: Finds the location of the given character in the given string

CALLING SEQUENCE:

CALL WHERE (CHAR, STRING, LEN, LOC, FOUND)

3.3.3.6 Variable Description

The variables in the calling sequences of major RU routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHGPER(3)</td>
<td>R*4</td>
<td>Number of changes per 1000 lines per type (new, delivered, adjusted)</td>
</tr>
<tr>
<td>COMDEL</td>
<td>I*2</td>
<td>Number of components delivered</td>
</tr>
<tr>
<td>COMNEW</td>
<td>I*2</td>
<td>Number of new components</td>
</tr>
<tr>
<td>COSPER(3)</td>
<td>R*4</td>
<td>Cost per type (new, delivered, adjusted)</td>
</tr>
<tr>
<td>COSPM</td>
<td>R*4</td>
<td>Cost per person-month using programmer and management time only</td>
</tr>
<tr>
<td>COSPMS</td>
<td>R*4</td>
<td>Cost per person-month using programmer, management, and services time</td>
</tr>
<tr>
<td>COST</td>
<td>R*4</td>
<td>Total cost based on weighted hours</td>
</tr>
<tr>
<td>COSTHR</td>
<td>R*4</td>
<td>Cost per hour</td>
</tr>
<tr>
<td>COSTMM</td>
<td>R*4</td>
<td>Cost per person-month</td>
</tr>
<tr>
<td>CPRGHR(6)</td>
<td>R*4</td>
<td>Total hours spent in each phase from CSR file record</td>
</tr>
<tr>
<td>CSRFIL</td>
<td>I*2</td>
<td>FORTRAN unit number for CSR file</td>
</tr>
<tr>
<td>CSRNAM(25)</td>
<td>L*1</td>
<td>CSR file name</td>
</tr>
<tr>
<td>CSTFIL(25)</td>
<td>L*1</td>
<td>CSR file name</td>
</tr>
<tr>
<td>DRANG1(3,6)</td>
<td>I*2</td>
<td>Phase start dates</td>
</tr>
<tr>
<td>DRANG2(3,6)</td>
<td>I*2</td>
<td>Phase end dates</td>
</tr>
<tr>
<td>EOF</td>
<td>L*1</td>
<td>Terminal EOF flag</td>
</tr>
<tr>
<td>EQU75</td>
<td>R*4</td>
<td>IBM S/360-95 plus S/360-75 computer time in equivalent S/360-75 time</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>E75PER(3)</td>
<td>R*4</td>
<td>Equivalent S/360-75 computer time per type (new, delivered, adjusted)</td>
</tr>
<tr>
<td>HRMON</td>
<td>R*4</td>
<td>Hours per month</td>
</tr>
<tr>
<td>HRYR</td>
<td>R*4</td>
<td>Hours per year</td>
</tr>
<tr>
<td>HR75</td>
<td>R*4</td>
<td>S/360-75 computer time in hours</td>
</tr>
<tr>
<td>HR95</td>
<td>R*4</td>
<td>S/360-95 computer time in hours</td>
</tr>
<tr>
<td>H75PER(3)</td>
<td>R*4</td>
<td>S/360-75 computer time per type (new, delivered, adjusted)</td>
</tr>
<tr>
<td>H95PER(3)</td>
<td>R*4</td>
<td>S/360-95 computer time per type (new, delivered, adjusted)</td>
</tr>
<tr>
<td>INLF</td>
<td>I*2</td>
<td>FORTRAN unit number for RU input parameters file</td>
</tr>
<tr>
<td>IRPTF</td>
<td>I*2</td>
<td>FORTRAN unit number for RU output report file</td>
</tr>
<tr>
<td>IRSFF</td>
<td>I*2</td>
<td>FORTRAN unit number for RSF file</td>
</tr>
<tr>
<td>LINDEL</td>
<td>I*2</td>
<td>Number of delivered source lines (in thousands)</td>
</tr>
<tr>
<td>LINNEW</td>
<td>I*2</td>
<td>Number of new source lines (in thousands)</td>
</tr>
<tr>
<td>LNMULT</td>
<td>I*2</td>
<td>Source lines multiple used in computing statistics</td>
</tr>
<tr>
<td>LP(3)</td>
<td>I*2</td>
<td>Source lines produced per person-month using programmer time only</td>
</tr>
<tr>
<td>LPM(3)</td>
<td>I*2</td>
<td>Source lines produced per person-month using programmer and management time</td>
</tr>
<tr>
<td>LPMS(3)</td>
<td>I*2</td>
<td>Source lines produced per person-month using programmer, management, and services time</td>
</tr>
<tr>
<td>MCOST(6)</td>
<td>R*4</td>
<td>Weighted management cost</td>
</tr>
<tr>
<td>MGCST(6)</td>
<td>I*2</td>
<td>Number of CSR forms with management data</td>
</tr>
<tr>
<td>MGHR(6)</td>
<td>R*4</td>
<td>Management hours from the RSFs</td>
</tr>
<tr>
<td>MGMTHR(6)</td>
<td>I*2</td>
<td>Management hours by phase</td>
</tr>
<tr>
<td>MGRSF(6)</td>
<td>I*2</td>
<td>Number of RSFs with management data</td>
</tr>
<tr>
<td>MGWT</td>
<td>R*4</td>
<td>Management weight</td>
</tr>
</tbody>
</table>

3-55

8818
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MHR(6)</td>
<td>*2</td>
<td>Management hours by phase</td>
</tr>
<tr>
<td>MMM(6)</td>
<td>*4</td>
<td>Management hours in person-months by phase</td>
</tr>
<tr>
<td>MPCT(6)</td>
<td>*2</td>
<td>Percent of management hours in each phase</td>
</tr>
<tr>
<td>MPHSPC(6)</td>
<td>*2</td>
<td>Percent of weighted management cost for each phase</td>
</tr>
<tr>
<td>MWTHR(6)</td>
<td>*2</td>
<td>Weighted management hours by phase</td>
</tr>
<tr>
<td>MWTPM(6)</td>
<td>*4</td>
<td>Weighted management hours in person-months by phase</td>
</tr>
<tr>
<td>MWTPCT(6)</td>
<td>*2</td>
<td>Percent of weighted management hours of a phase</td>
</tr>
<tr>
<td>NCHANG</td>
<td>*2</td>
<td>Number of changes</td>
</tr>
<tr>
<td>NCOMP(4)</td>
<td>*2</td>
<td>Number of components by type (new, delivered, adjusted, old)</td>
</tr>
<tr>
<td>NDATWK(6)</td>
<td>*2</td>
<td>Number of weeks with data in the phase</td>
</tr>
<tr>
<td>NLFIL(25)</td>
<td>*1</td>
<td>RU input parameters file name</td>
</tr>
<tr>
<td>NONCOM</td>
<td>*2</td>
<td>Source lines excluding comments</td>
</tr>
<tr>
<td>NWEKS(6)</td>
<td>*2</td>
<td>Number of weeks in phase</td>
</tr>
<tr>
<td>OLDIPC</td>
<td>*4</td>
<td>Factor used to compute adjusted lines of code from old and new figures</td>
</tr>
<tr>
<td>PAGENO</td>
<td>*2</td>
<td>Page number on report</td>
</tr>
<tr>
<td>PCOST(6)</td>
<td>*4</td>
<td>Weighted programmer cost by phase</td>
</tr>
<tr>
<td>PHR(6)</td>
<td>*2</td>
<td>Programmer hours by phase</td>
</tr>
<tr>
<td>PHRANG(6,2)</td>
<td>*2</td>
<td>Number range of phases</td>
</tr>
<tr>
<td>PHRASE(50)</td>
<td>*1</td>
<td>Title of RU report</td>
</tr>
<tr>
<td>PMM(6)</td>
<td>*4</td>
<td>Programmer hours in person-months by phase</td>
</tr>
<tr>
<td>PPCT(6)</td>
<td>*2</td>
<td>Percent of programmer hours in each phase</td>
</tr>
<tr>
<td>PPHSPC(6)</td>
<td>*2</td>
<td>Percent of weighted programmer cost for each phase</td>
</tr>
<tr>
<td>PRGCST(6)</td>
<td>*2</td>
<td>Number of CSR forms with programmer data</td>
</tr>
<tr>
<td>PRGRSF(6)</td>
<td>*2</td>
<td>Number of RSFs with programmer data</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>PRJNAM(8)</td>
<td>L*1</td>
<td>Project name</td>
</tr>
<tr>
<td>PROGRHR(6)</td>
<td>I*2</td>
<td>Programmer hours by phase</td>
</tr>
<tr>
<td>PROJ(8)</td>
<td>L*1</td>
<td>Project name</td>
</tr>
<tr>
<td>PROJECT(8)</td>
<td>L*1</td>
<td>Project name from RSF file</td>
</tr>
<tr>
<td>PRWT</td>
<td>R*4</td>
<td>Programmer weight</td>
</tr>
<tr>
<td>PWTHR(6)</td>
<td>I*2</td>
<td>Weighted programmer hours by phase</td>
</tr>
<tr>
<td>WTMMM(6)</td>
<td>R*4</td>
<td>Weighted programmer hours in person-months by phase</td>
</tr>
<tr>
<td>PWTPC(6)</td>
<td>I*2</td>
<td>Percent of weighted programmer hours of a phase</td>
</tr>
<tr>
<td>RPTFIL(25)</td>
<td>L*1</td>
<td>RU report file name</td>
</tr>
<tr>
<td>RSFIL(25)</td>
<td>L*1</td>
<td>RSF file name</td>
</tr>
<tr>
<td>RSFPHR(6)</td>
<td>R*4</td>
<td>Programmer hours for each phase</td>
</tr>
<tr>
<td>RUNPER(3)</td>
<td>R*4</td>
<td>Number of runs per type (new, delivered, adjusted)</td>
</tr>
<tr>
<td>RUNS</td>
<td>I*2</td>
<td>Total number of runs</td>
</tr>
<tr>
<td>SCOST(6)</td>
<td>R*4</td>
<td>Weighted services cost for each phase</td>
</tr>
<tr>
<td>SERVHR(6)</td>
<td>I*2</td>
<td>Services hours by phase</td>
</tr>
<tr>
<td>SHR(6)</td>
<td>I*2</td>
<td>Services hours by phase</td>
</tr>
<tr>
<td>SLINES(4)</td>
<td>I*2</td>
<td>Number of source lines (in thousands) (new, delivered, adjusted, old)</td>
</tr>
<tr>
<td>SMM(6)</td>
<td>R*4</td>
<td>Services hours in person-months by phase</td>
</tr>
<tr>
<td>SPCT(6)</td>
<td>I*2</td>
<td>Percent of services hours in each phase</td>
</tr>
<tr>
<td>SPHSPC(6)</td>
<td>I*2</td>
<td>Percent of weighted services cost for each phase</td>
</tr>
<tr>
<td>SVHR(6)</td>
<td>R*4</td>
<td>Services hours for each phase</td>
</tr>
<tr>
<td>SVWT</td>
<td>R*4</td>
<td>Services weight</td>
</tr>
<tr>
<td>SWTHR(6)</td>
<td>I*2</td>
<td>Weighted services hours by phase</td>
</tr>
<tr>
<td>SWTMM(6)</td>
<td>R*4</td>
<td>Weighted services hours in person-months by phase</td>
</tr>
<tr>
<td>SWTTPC(6)</td>
<td>I*2</td>
<td>Percent of weighted services hours for each phase</td>
</tr>
<tr>
<td>TCOST(6)</td>
<td>R*4</td>
<td>Weighted total cost for each phase</td>
</tr>
</tbody>
</table>

3-57

8818
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TERMNL</td>
<td>L*1</td>
<td>Flag indicating whether terminal or external file is to be read</td>
</tr>
<tr>
<td>THR(6)</td>
<td>I*2</td>
<td>Total hours by phase</td>
</tr>
<tr>
<td>TMM(6)</td>
<td>R*4</td>
<td>Total hours in person-months by phase</td>
</tr>
<tr>
<td>TPCT(6)</td>
<td>I*2</td>
<td>Percent of total hours for each phase</td>
</tr>
<tr>
<td>TPHSPC(6)</td>
<td>I*2</td>
<td>Percent of weighted total cost for each phase</td>
</tr>
<tr>
<td>TURN</td>
<td>I*2</td>
<td>Flag indicating whether the programmer data are from the CSR or the RSF file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 1, from RSF file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 2, from CSR file</td>
</tr>
<tr>
<td>TWTHR(6)</td>
<td>I*2</td>
<td>Weighted total hours by phase</td>
</tr>
<tr>
<td>TWTMM(6)</td>
<td>R*4</td>
<td>Weighted total hours in person-months by phase</td>
</tr>
<tr>
<td>TWTPCT(6)</td>
<td>I*2</td>
<td>Percent of weighted total hours for each phase</td>
</tr>
<tr>
<td>T95T75</td>
<td>R*4</td>
<td>Factor used to convert S/360-95 computer time to S/360-75 time</td>
</tr>
<tr>
<td>WEEKPC(6)</td>
<td>I*2</td>
<td>Percent of weeks for each phase</td>
</tr>
<tr>
<td>WTLPM(3)</td>
<td>I*2</td>
<td>Weighted source lines produced per person-month using programmer time only</td>
</tr>
<tr>
<td>WTLPMMS(3)</td>
<td>I*2</td>
<td>Weighted source lines produced per person-month using programmer and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>management time</td>
</tr>
<tr>
<td>XCOST</td>
<td>R*4</td>
<td>Total cost based on weighted hours</td>
</tr>
<tr>
<td>XMOMM</td>
<td>R*4</td>
<td>Total management hours</td>
</tr>
<tr>
<td>XMWTMM</td>
<td>R*4</td>
<td>Total weighted management hours</td>
</tr>
<tr>
<td>XPM</td>
<td>R*4</td>
<td>Total programmer hours</td>
</tr>
<tr>
<td>XPWTMM</td>
<td>R*4</td>
<td>Total weighted programmer hours</td>
</tr>
<tr>
<td>X5MM</td>
<td>R*4</td>
<td>Total services hours</td>
</tr>
<tr>
<td>XSWTMM</td>
<td>R*4</td>
<td>Total weighted services hours</td>
</tr>
</tbody>
</table>
3.3.4 TASK BUILD PROCEDURE

3.3.4.1 Command Procedures

The RU program can be generated from the source code by executing the command procedure RUGEN.CMD under UIC [204,6]. This command procedure references three command files--RUFPP.CMD, RUFOR.CMD, and RU.TKB--all under UIC [204,6]. Figure 3-8 is a listing of RUGEN.CMD, the command procedure to precompile, compile, and task build the RU program. The RU program is generated by entering the following command:

```
@ [204,6] RUGEN
```

3.3.4.2 Overlay Structure

The RU program is overlaid to reduce the memory space requirement. Figure 3-9 is a listing of the Overlay Descriptor Language file, [204,6]RU.ODL, needed to build the RU program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay. In addition, the RMS Indexed Access Program Library (RMSIAC) is also needed in the overlay. The name of the library is [204,7]UFRMSIAC.OLB. It contains FORTRAN routines necessary for accessing RMS indexed files.
Figure 3-8. RU Task Generation Command Procedure (RUGEN.CMD) (1 of 2)
Figure 3-8. RU Task Generation Command Procedure (RUGEN.CMD) (2 of 2)
Figure 3-9. RU Program Overlay Descriptor Language File (RU.ODL)
3.4 WEEKLY HOUR AND FORM COUNT PROGRAM (WK)

3.4.1 INTRODUCTION

The Weekly Hour and Form Count Report Program (WK) produces reports of hour or form counts from a desired SEL data base file for a given project. There are currently 14 different WK reports. Each report contains counts of records, forms, or other data given by programmer by week (Section 2.4).

3.4.2 PROGRAM STRUCTURE

3.4.2.1 Files Accessed

Each of the 14 reports currently produced by the WK program accesses four input files and three output files. All possible files are listed below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]ENCODE.HDR</td>
<td>Encoding Dictionary (ENC) file (accessed by all report types)</td>
</tr>
<tr>
<td>[204,1]EST.HDR</td>
<td>Estimated Statistics (EST) file (accessed by all report types)</td>
</tr>
<tr>
<td>[204,1]HEADER.HDR</td>
<td>Phase Dates (HDR) file (accessed by all report types)</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.ACC</td>
<td>Accounting Information (ACC) file for the given project (accessed by report types XW1, XW2, and XW3)</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CRF</td>
<td>Change Report Form (CRF) file for the given project (accessed by report type HW)</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSF</td>
<td>Component Summary Form (CSF) file for the given project (accessed by report type MW)</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSR</td>
<td>Component Status Report (CSR) file for the given project (accessed by report types TH and TW)</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RAF</td>
<td>Run Analysis Form (RAF) file for the given project (accessed by report types AW1 and AW2)</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RSF</td>
<td>Resource Summary Form (RSF) file for the given project (accessed by report types RH1, RH2, RH3, RP, and RR)</td>
</tr>
</tbody>
</table>

8818
### Output File Name

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;PRJNAM&gt;.xxx</code></td>
<td>Report file for the given project, where <code>xxx</code> = report type (AW1, AW2, HW, MW, RH1, RH2, RH3, RP, RR, TH, TW, XW1, XW2, or XW3)</td>
</tr>
<tr>
<td><code>&lt;PRJNAM&gt;1xxx</code></td>
<td>Plot file for the given project for pie chart plotting (not implemented), where <code>xxx</code> = report type</td>
</tr>
<tr>
<td><code>&lt;PRJNAM&gt;2xxx</code></td>
<td>Plot file for the given project for graphing (Section 3.7), where <code>xxx</code> = report type</td>
</tr>
</tbody>
</table>

In these files names, `<PRJNAM>` is the name of the project selected by the user.

#### 3.4.2.2 Baseline Diagram

Figure 3-10 is the baseline diagram for the WK program. The WKDMP routine is the main driver. It displays the help information, obtains the project name and report type, reads the desired file for a given project, reads the HDR and EST files, and produces the report by resource or programmer by week with subtotals given by phase. WKDMP loops through the above process until a ^Z (control Z) is returned by the user in response to a prompt.

#### 3.4.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines forming the WK program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major WK routines are described in Section 3.4.3.7. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the WK program also uses the following system routines: DATE, ERRSET, ERRSNS, and TIME.

8818
Figure 3-10. Baseline Diagram for the Weekly Hour and Form Count Report Program (WK) (1 of 3)
Figure 3-10. Baseline Diagram for the Weekly Hour and Form Count Report Program (WK) (2 of 3)
Figure 3-10. Baseline Diagram for the Weekly Hour and Form Count Report Program (WK) (3 of 3)
3.4.3.1  Process Data and Compute Statistics

These twelve major routines obtain data from the given SEL data base file and compute statistics for the WK report.

ROUTINE:  ACCHR7

FUNCTION:  Reads one ACC file record and returns the date of record, computer code, and IBM S/360-75 time

CALLING SEQUENCE:

CALL ACCHR7 (IDBF,
               DATE, KOUNT, RESID, NULL, EOF, ERROR)

ROUTINE:  ACCHR9

FUNCTION:  Reads one ACC file record and returns the date of record, computer code, and IBM S/360-95 time

CALLING SEQUENCE:

CALL ACCHR9 (IDBF,
               DATE, KOUNT, RESID, NULL, EOF, ERROR)

ROUTINE:  ACCRUN

FUNCTION:  Reads one record from the ACC file and returns the date of record, computer code, and run count

CALLING SEQUENCE:

CALL ACCRUN (IDBF,
               DATE, KOUNT, RESID, NULL, EOF, ERROR)

ROUTINE:  CRFCNT

FUNCTION:  Reads one record from the CRF file and returns the date of form, programmer number, and count

CALLING SEQUENCE:

CALL CRFCNT (IDBF,
              DATE, KOUNT, RESID, NULL, EOF, ERROR)

3-68

8818
ROUTINE: CSFCNT

FUNCTION: Reads one CSF file record and returns the date of form, programmer code, and count

CALLING SEQUENCE:

CALL CSFCNT (IDBF,
             DATE, KOUNT, RESID, NULL, EOF, ERROR)

ROUTINE: CSRCNT

FUNCTION: Reads one CSR file record and returns the date of form, programmer code, and count

CALLING SEQUENCE:

CALL CSRCNT (IDBF,
             DATE, KOUNT, RESID, NULL, EOF, ERROR)

ROUTINE: CSRHR

FUNCTION: Reads one CSR file record and returns the date of form, programmer number, and hour count

CALLING SEQUENCE:

CALL CSRHR (IDBF,
             DATE, KOUNT, RESID, NULL, EOF, ERROR)

ROUTINE: MAKWKS

FUNCTION: Sets up an array of weeks covering the given timespan

CALLING SEQUENCE:

CALL MAKWKS (DRANG1, DRANG2,
             NWEEKS, WEEKS)
ROUTINE: NEXTWK

FUNCTION: Computes date 1 week from the given date in YYMMDD format

CALLING SEQUENCE:

CALL NEXTWK (DATE, D)

ROUTINE: RAFCNT

FUNCTION: Reads one RAF file record and returns the date of form, programmer number, and form count

CALLING SEQUENCE:

CALL RAFCNT (IDBF, DATE, KOUNT, RESID, NULL, EOF, ERROR)

ROUTINE: RARUNS

FUNCTION: Reads one RAF file record and returns the date of form, programmer number, and run count

CALLING SEQUENCE:

CALL RARUNS (IDBF, DATE, KOUNT, RESID, NULL, EOF, ERROR)

ROUTINE: RSFHR

FUNCTION: Accumulates staff hours from the RSF file for each week from the beginning of the design phase to the end of the cleanup phase

CALLING SEQUENCE:

CALL RSFHR (DRANG1, DRANG2, IRSFF, KEY, RSFNAM, RSFRUN, TYPE, AFTTOT, ALLTOT, BEFTOF, HRDATA, NPROG, NWEEKS, PHDATA, PHTOT, PRGAFT, PRGBEF, PRGTOT, PROGNO, WEEKS, WKTOT, ERROR)
ROUTINE: WKDATA

FUNCTION: Accumulates staff hours or counts for each week in the given timespan from the given data base file

CALLING SEQUENCE:

CALL WKDATA (DRANG1, DRANG2, IDBF, RSFNAM, TYPE, AFTTOT, ALLTOT, BEFTOT, HRDATA, NPROG, NWEKS, PHDATA, PHTOT, PRGAFT, PRGBEF, PRGTOT, PROGNO, WEEKS, WKTOT, ERROR)

ROUTINE: WKDMP

FUNCTION: Main routine of the WK program, reads the desired file for a given project and produces a report by person by week with subtotals by phase

CALLING SEQUENCE: None

3.4.3.2 Write Output Reports and Plot Files

These seven routines write the output report and plot files.

ROUTINE: DMPRPT

FUNCTION: Prints the complete WK report

CALLING SEQUENCE:

CALL DMPRPT (AFTTOT, ALLTOT, BEFTOT, DESCR, DRANG1, DRANG2, HRDATA, IRPTF, NPROG, NWEKS, PHDATA, PHTOT, PRGAFT, PRGBEF, PRGTOT, PRJNAM, RPTITL, RPTNAM, SRTIDX, WEEKS, WKTOT)

ROUTINE: FSUMRY

FUNCTION: Prints a six-line header summary with data from the HDR and EST files

CALLING SEQUENCE:

CALL FSUMRY (IRPTF, PRJNAM)
OUTLINE: HEADER

FUNCTION: Prints a one-line title for each report page, including the date and the project name

CALLING SEQUENCE:

CALL HEADER (IRPTF, PRJNAM, RPTITL)

ROUTINE: PRT1

FUNCTION: Prints the WK report header page

CALLING SEQUENCE:

CALL PRT1 (DESCR, DRANG1, DRANG2, IRPTF, NPROG, PRJNAM, RPTITL, SRTIDX)

ROUTINE: PRT2

FUNCTION: Prints the WK report data page

CALLING SEQUENCE:

CALL PRT2 (AFTTOT, ALLTOT, BEFTOT, DESCR, DRANG1, DRANG2, HRDATA, IRPTF, NPROG, NWEEKS, PHDATA, PHTOT, PRGAFT, PRGBEF, PRGTOT, PRJNAM, RPTITL, SRTIDX, WEEKS, WKTOT)

ROUTINE: WRTPLT

FUNCTION: Writes the given data to an intermediate file in preparation for pie chart plotting

CALLING SEQUENCE:

CALL WRTPLT (DATA, DESCR, EXT, NDATA, PIETTL, PRJNAM, RPTITL)

ROUTINE: WRTPL3

FUNCTION: Writes the given data to an intermediate file in preparation for graphing
CALLING SEQUENCE:

CALL WRTPL3 (DATA, EXT, KINES, MARKER, NDATA, PIETTL, PRJNAM, RPTITL, XMAX, XTITLE, YMAX, YTITLE)

ROUTINE: WRTPL3
FUNCTION: Writes the given data to an intermediate file in preparation for graphing

CALLING SEQUENCE:

CALL WRTPL3 (DATA, EXT, KINES, MARKER, NDATA, PIETTL, PRJNAM, RPTITL, XMAX, XTITLE, YMAX, YTITLE)

3.4.3.3 Obtain Data From Terminal or External File

These five routines obtain information from a user's response to a terminal prompt or from an external file.

ROUTINE: FENCA
FUNCTION: Finds the description field on the Encoding Dictionary corresponding to the given type and code

CALLING SEQUENCE:

CALL FENCA (IENCF, TYPE, CODE, NAME, REST, FOUND)

ROUTINE: GETFLD
FUNCTION: Displays the given text on the terminal and prompts for a character string

CALLING SEQUENCE:

CALL GETFLD (TEXT, EXTFIL, FLDLEN, TERMNL, EOFTTY, ERROR, FIELD)

ROUTINE: GETOPT
FUNCTION: Obtains the project name from the terminal
CALLING SEQUENCE:

CALL GETOPT (TERMNL,  
   PRJNAM, RPTITL, RPTNAM, RSFNAM, TYPE, EOF,  
   ERROR)

ROUTINE: HELP
FUNCTION: Prints help information to the terminal

CALLING SEQUENCE:

   CALL HELP

ROUTINE: NAME3
FUNCTION: Concatenates the given strings to form a complete file name

CALLING SEQUENCE:

   CALL NAME3 (DISK, UIC, NAME, EXTENS,  
   DSN)

3.4.3.4 Sort and Search Routines
These seven routines provide some sort and search functions.

ROUTINE: HIPT (INTEGER*2 FUNCTION)
FUNCTION: Finds the first integer having a single significant digit that is greater than the given integer

CALLING SEQUENCE:

   HIPT(L)

ROUTINE: INWK
FUNCTION: Determines whether the given date is within the date range


CALLING SEQUENCE:

CALL INWK (DATIN, DATE1, DATE2, INWEEK)

ROUTINE: MAXIM (INTEGER*2 FUNCTION)
FUNCTION: Finds the maximum number in an array of integers
CALLING SEQUENCE:

MAXIM (ARRAY, NARRAY)

ROUTINE: PHSCHK
FUNCTION: Determines whether the given date is within the start and end dates of the given range
CALLING SEQUENCE:

CALL PHSCHK (FDATE, DRANG1, DRANG2, PHNUM, INPHAS)

ROUTINE: PROCNM
FUNCTION: Converts given programmer numbers into programmer names
CALLING SEQUENCE:

CALL PROCNM (IENCF, NPROG, PROGNO, KTYPE, DESCR, SRTIDX, ERROR)

ROUTINE: STORE
FUNCTION: Determines whether the given number is in the given array, adds it if it is not, and returns the location of the given number in the given array
CALLING SEQUENCE:

CALL STORE (RESID, MAXPRG, 
PROGNO, NPROG, 
IDNUM, BADID)

ROUTINE: WKCHEK

FUNCTION: Determines which week in a given array of weeks contains the given date

CALLING SEQUENCE:

CALL WKCHEK (DATIN, NWEEKS, WEEKS, 
WKNUM, INWEEK)

3.4.3.5 File Open and Read Routines

These ten routines either open an indexed file or read records from an indexed file.

ROUTINE: FACC

FUNCTION: Reads one record from the ACC file

CALLING SEQUENCE:

CALL FACC (IACCF, 
PRJCOD, DATE, TIME, TSOFOR, TSOBCK, RJE, 
CRDRDR, CPL, CPU95, IO95, RUNS95, FAIL95, CP2, 
CPU75, IO75, RUNS75, FAIL75, ISTAT, EOF, ERROR)

ROUTINE: FCRF

FUNCTION: Reads one record from the CRF file and converts the data to internal format

CALLING SEQUENCE:

CALL FCRF (ICRFF, 
FORMNO, PROJNO, PROGNO, FDATE, NCH, NEXAM, 
OVER1, DATDET, DATBEG, EFFORT, CTYPE, CHCOMP, 
ERRTYP, ERRIN, DATERR, LGCERR, ACTVTY, ISOLTM, 
PATCH, RELOLD, RELNO, RELDAT, CMTREA, CMTDES, 
CMTGEN, STATUS, EOF, ERROR)

3-76

8818
ROUTINE: FCSF

FUNCTION: Reads one record from the CSF file

CALLING SEQUENCE:

CALL FCSF (ICSFF,
           FORMNO, PROJNO, PROGNO, PROGI, FDATE, FSTAGE,
           COMPCO, PRECIS, CMPLEX, SWTYP, PASGN, PCNTL,
           POTHER, STATWO, STMT, BTSIZE, INDEP, RELSW,
           ADDTYP, NCALLD, X1, NCALNG, X2, NSHR, X3,
           NDESC, X4, LANG1, PLANG1, LANG2, PLANG2, DES,
           CONSTR, DESRUN, CODRUN, TSTRUN, DESTIM,
           CODTIM, TSTTIM, DESEFF, CODEFF, TSTEFF,
           DESDAT, CODDAT, TSTDAT, DESCR, CALLD, CALNG,
           SHR, AFFECT, OTH, NAMCON, CMT1, CMT2, ISTAT,
           EOF, ERROR)

ROUTINE: FCSR

FUNCTION: Reads one record from the CSR file using a FORTRAN read

CALLING SEQUENCE:

CALL FCSR (ICSRF,
           FORMNO, SEQNO, PROJNO, PROGNO, FDATE, COMPCO,
           TIMES, OTHNAM, OTHOUR, ISTAT, PHASE, EOF,
           ERROR)

ROUTINE: FEST

FUNCTION: Reads one record from the EST file and converts all data to internal format

CALLING SEQUENCE:

CALL FEST (IESTF, NAME,
           PROJ, NCOMP, MODDEL, MODNEW, MODMOD, NRUNS,
           NCHANG, PAGDOC, LINDEL, LINNEW, LINMOD,
           TOTEXT, NEWEXT, MODEXT, PROGRH, MGMTHR,
           OTHHR, HR95, HR75, OTHCMP, STATUS, ACTIVE,
           PRJCAT, FOUND, ERROR)

3-77

8818
ROUTINE: FHDR

FUNCTION: Reads one record from the HDR file and converts all data to internal format

CALLING SEQUENCE:

CALL FHDR (IHDRF, PRJNAM,
            PROJ, DEVCMP, TARG, ALIEN, RANGES, STATUS,
            ERROR)

ROUTINE: FOPEN

FUNCTION: Opens an indexed file

CALLING SEQUENCE:

CALL FOPEN (IUNIT, FILNAM,
            ERROR)

ROUTINE: FRAF

FUNCTION: Reads one record from the RAF file using a FORTRAN read

CALLING SEQUENCE:

CALL FRAF (IRAFF,
            FORMNO, SEQNO, PROJNO, PROGNO, RDATE, MACHIN,
            INTERA, PURPOS, NCOMP, COMPCO, FIRST, METOBJ,
            RESULT, COMENT, ISTAT, EOF, ERROR)

ROUTINE: FREAD

FUNCTION: Reads one indexed record

CALLING SEQUENCE:

CALL FREAD (IUNIT, KEYVAL, KEYLEN, LRECL,
            BUFFER, ERROR)
ROUTINE: FRSF

FUNCTION: Reads one record from the RSF file and returns all data on that record plus an array of dates for each week for which there is a resource entry on the record

CALLING SEQUENCE:

CALL FRSF (IRSFF,
           FORMNO, SEQNO, PROJNO, RESCOD, RESID, FDATE,
           PCGMT, WKDATE, NRUNS, TIMES, STATUS, PHASE,
           LASTWK, EOF, ERROR)

3.4.3.6 Routines for String Movement or Comparison

These four routines concern string movement or comparison.

ROUTINE: BLANK

FUNCTION: Initializes an array to blanks

CALLING SEQUENCE:

CALL BLANK (ARRAY, NUM)

ROUTINE: MATCHS (LOGICAL FUNCTION)

FUNCTION: Determines whether the two input strings are the same

CALLING SEQUENCE:

MATCHS (ARRAY1, ARRAY2, NBYTES)

ROUTINE: MOVE

FUNCTION: Moves given number of bytes from one address to another

CALLING SEQUENCE:

CALL MOVE (A, B, LEN)
ROUTINE: WHERE

FUNCTION: Finds the location of the given character in the given string

CALLING SEQUENCE:

CALL WHERE (CHAR, STRING, LEN, LOC, FOUND)

3.4.3.7 Variable Description

The variables in the calling sequences of major WK routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFTTOT</td>
<td>I*2</td>
<td>Total number of programmer hours after cleanup</td>
</tr>
<tr>
<td>ALLTOT</td>
<td>I*4</td>
<td>Total programmer hours</td>
</tr>
<tr>
<td>ARRAY (NARRAY)</td>
<td>I*2</td>
<td>Array to be searched</td>
</tr>
<tr>
<td>BADID</td>
<td>L*1</td>
<td>Error flag to indicate that there is no room for the new number in the given array</td>
</tr>
<tr>
<td>BEFTOT</td>
<td>I*2</td>
<td>Total programmer hours before design</td>
</tr>
<tr>
<td>DATE (3)</td>
<td>I*2</td>
<td>Form date (YY,MM,DD)</td>
</tr>
<tr>
<td>DATE1 (3)</td>
<td>I*2</td>
<td>Range start date (YY,MM,DD)</td>
</tr>
<tr>
<td>DATE2 (3)</td>
<td>I*2</td>
<td>Range end date (YY,MM,DD)</td>
</tr>
<tr>
<td>DATIN (3)</td>
<td>I*2</td>
<td>Given date (YY,MM,DD)</td>
</tr>
<tr>
<td>DESC (20,20)</td>
<td>L*1</td>
<td>Programmer names</td>
</tr>
<tr>
<td>DRANG1 (3, 6)</td>
<td>I*2</td>
<td>Phase start dates</td>
</tr>
<tr>
<td>DRANG2 (3, 6)</td>
<td>I*2</td>
<td>Phase end dates</td>
</tr>
<tr>
<td>EOF</td>
<td>L*1</td>
<td>End-of-file flag</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>HRDATA (20,400)</td>
<td>I*2</td>
<td>Number of programmer (or other) hours for each week</td>
</tr>
<tr>
<td>IDBF</td>
<td>I*2</td>
<td>Unit number for data base file</td>
</tr>
<tr>
<td>IDNUM</td>
<td>I*2</td>
<td>Location of given number in array</td>
</tr>
<tr>
<td>IENC</td>
<td>I*2</td>
<td>Unit number for ENC file</td>
</tr>
</tbody>
</table>

3-80

8818
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INWEEK</td>
<td>L*1</td>
<td>Flag indicating whether given date falls within range</td>
</tr>
<tr>
<td>IRPTF</td>
<td>I*2</td>
<td>Unit number for output report file</td>
</tr>
<tr>
<td>IRSFF</td>
<td>I*2</td>
<td>Unit number for RSF file</td>
</tr>
<tr>
<td>KEY</td>
<td>L*1</td>
<td>Code used to determine which resource is desired</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M = manpower</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C = computer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>O = other (services)</td>
</tr>
<tr>
<td>KOUNT</td>
<td>I*2</td>
<td>Hour, person, or run count for given record</td>
</tr>
<tr>
<td>KTYPE</td>
<td>I*2</td>
<td>Resource type</td>
</tr>
<tr>
<td>L</td>
<td>I*2</td>
<td>Given number</td>
</tr>
<tr>
<td>MAXPRG</td>
<td>I*2</td>
<td>Maximum number of array elements allowed</td>
</tr>
<tr>
<td>NARRAY</td>
<td>I*2</td>
<td>Size of array</td>
</tr>
<tr>
<td>NPROG</td>
<td>I*2</td>
<td>Number of programmers</td>
</tr>
<tr>
<td>NULL</td>
<td>L*1</td>
<td>Flag indicating whether record read is usable</td>
</tr>
<tr>
<td>NWEEKS</td>
<td>I*2</td>
<td>Number of weeks in project</td>
</tr>
<tr>
<td>PHDATA(20,5)</td>
<td>I*2</td>
<td>Phase subtotals</td>
</tr>
<tr>
<td>PHTOT(5)</td>
<td>I*2</td>
<td>Phase totals</td>
</tr>
<tr>
<td>PRGAFT(20)</td>
<td>I*2</td>
<td>Programmer totals after cleanup</td>
</tr>
<tr>
<td>PRGBEF(20)</td>
<td>I*2</td>
<td>Programmer totals before design</td>
</tr>
<tr>
<td>PRGTOT(20)</td>
<td>I*2</td>
<td>Totals for each programmer</td>
</tr>
<tr>
<td>PRJNAM(8)</td>
<td>L*1</td>
<td>Project name</td>
</tr>
<tr>
<td>PROGNO(20)</td>
<td>I*4</td>
<td>Programmer numbers</td>
</tr>
<tr>
<td>RESID</td>
<td>I*4</td>
<td>Programmer or computer code</td>
</tr>
<tr>
<td>RPTITL(40)</td>
<td>L*1</td>
<td>Report title</td>
</tr>
<tr>
<td>RPTNAM(27)</td>
<td>L*1</td>
<td>Report file name</td>
</tr>
<tr>
<td>RSFNAM(27)</td>
<td>L*1</td>
<td>Data base file name</td>
</tr>
<tr>
<td>RSFRUN</td>
<td>L*1</td>
<td>Flag indicating that RSF file run count is desired</td>
</tr>
<tr>
<td>SRTIDX(20)</td>
<td>I*2</td>
<td>Sorted index array to alphabetize programmers</td>
</tr>
<tr>
<td>TERMNL</td>
<td>L*1</td>
<td>Flag of whether to read from terminal or external file</td>
</tr>
</tbody>
</table>

3-81
3.4.4 TASK BUILD PROCEDURE

3.4.4.1 Command Procedures

The WK program can be generated from the source code by executing the command procedure WKGEN.CMD under UIC [204,6]. This command procedure references three command files—WKFPP.CMD, WKFOR.CMD, and WK.TKB—all under UIC [204,6]. Figure 3-11 is a listing of the command procedure WKGEN.CMD, which precompiles, compiles, and builds the WK program task image. The WK program is generated by entering the following command:

@{204,6}WKGEN

3.4.4.2 Overlay Structure

The WK program is overlaid to reduce the memory space requirement. Figure 3-12 is a listing of the Overlay Descriptor Language file, [204,6]WK.ODL, needed to build the WK program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay.
Figure 3-11. WK Task Generation Command Procedure (WKGEN.CMD) (1 of 3)
Figure 3-11. WK Task Generation Command Procedure (WKGEN.CMD) (2 of 3)
Figure 3-11. WK Task Generation Command Procedure
(WKGEN.CMD) (3 of 3)
Figure 3-12. WK Program Overlay Descriptor Language File (WK.ODL)
3.5 COMPONENT INFORMATION REPORT BY FUNCTION TYPE PROGRAM (REP4) AND ITS PREPROCESSOR, THE CHANGE AND ERROR ACCUMULATION PROGRAM (CG)

3.5.1 INTRODUCTION

The Component Information Report by Function Type Program (REP4) produces a list of components and associated data for a given project, organized by the function type of the component and sorted by the number of executable statements. The change and error data on this report are read from an intermediate file produced by the Change and Error Accumulation Program (CG).

3.5.2 PROGRAM STRUCTURE

3.5.2.1 Files Accessed

The CG program accesses two input files and two output files, as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CIF</td>
<td>Component Information File (CIF) for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CRF</td>
<td>Change Report Form (CRF) file for the given project</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;PRJNAM&gt;.CHN</td>
<td>CG intermediate output file containing change and error data for the given project</td>
</tr>
<tr>
<td>FOR006.DAT</td>
<td>File containing all component names not found on the CIF for the given project</td>
</tr>
</tbody>
</table>

The REP4 program accesses two input files and one output file, as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CIF</td>
<td>CIF for the given project</td>
</tr>
<tr>
<td>&lt;PRJNAM&gt;.CHN</td>
<td>CG intermediate file containing change and error data for the given project</td>
</tr>
</tbody>
</table>
3.5.2.2 Baseline Diagrams

Figure 3-13 is the baseline diagram for the CG program. The XCH routine is the main driver. It obtains the project name, reads the CIF and the CRF file for the given project, accumulates the change and error data from the CRF file, and writes the output files. XCH loops through this process until a \( ^2 \) (control Z) is returned in response to the prompt for the project name.

Figure 3-14 is the baseline diagram for the REP4 program. The driver routine, REP4, obtains the project name and selected subsystem, reads the CG intermediate file and the CIF for the given project, determines the component type, sorts all components by number of executable statements, and writes the output report. REP4 loops through this process until a \( ^2 \) (control Z) is returned in response to a subsystem prefix prompt.

3.5.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines referenced by the CG and REP4 programs are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major CG and REP4 routines are described in Section 3.5.3.5. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the CG and REP4 programs also reference the following system routines: DATE, ERRSNS, SECNDS, and TIME.

3-88

8818
Figure 3-13. Baseline Diagram for the Change and Error Accumulation Program (CG)
Figure 3-14. Baseline Diagram for the Component Information Report by Function Type Program (REP4)
3.5.3.1 Process Data and Compute Statistics

These six major routines obtain data from a given CIF or CRF file and compute statistics for the CG or the REP4 program.

ROUTINE: CHDATA

FUNCTION: Accumulates change and error data by component from the CRF file

CALLING SEQUENCE:

CALL CHDATA (LUNIT, MUNIT, NUNIT, OUTDSN)

ROUTINE: CTYPE

FUNCTION: Determines the function type of a component

CALLING SEQUENCE:

CALL CTYPE (ICTEXC, ICTFNR, ICTIO, KASGN, KCALL, KFMT, ITYPE)

ROUTINE: PERCNT

FUNCTION: Computes percentages of several statistics

CALLING SEQUENCE:

CALL PERCNT (ICTDOS, ICTEXC, ICTFNR, ICTIFF, ICTIO, IDECIS, KASGN, KCALL, KFMT, PASGN, PCALL, PDEC, PDOS, PFUNC, PIFS, PIO, PTOTS)

ROUTINE: REP4

FUNCTION: Main routine of the REP4 program, extracts data from the CIF and the CG intermediate file, determines the function type of the components, and writes the output report

CALLING SEQUENCE: None
ROUTINE: TYPREP

FUNCTION: Reads records from the CIF, computes statistics, and writes the report subdivided by function type of component

CALLING SEQUENCE:

    CALL TYPREP (ICHNGF, IREPF, LUNDB, ISORT, ITYPE, NSORT, ZPROJ, PREFIX, INAME)

ROUTINE: XCH

FUNCTION: Main routine of the CG program, accumulates change and error data from the CRF file and writes it to an intermediate output file

CALLING SEQUENCE: None

3.5.3.2 Input and Output Routines

These four routines perform input or output functions.

ROUTINE: CHNGES

FUNCTION: Reads the CG intermediate data file and returns the number of changes and errors for a given component name; if the component name is not found, the routine returns 999 for the output variables

CALLING SEQUENCE:

    CALL CHNGES (ANAME, ICHNGF, NCHS, NERRS, TOTCH)

ROUTINE: DCTION

FUNCTION: Prints dictionary of abbreviations for page 1 of the REP4 report

CALLING SEQUENCE:

    CALL DCTION (PROJ)
ROUTINE: DRDCIF

FUNCTION: Reads one record from the CIF and converts all data to internal format

CALLING SEQUENCE:

CALL DRDCIF (LUNIT, IKEY, KEYVAL, PROJNO, CPREFX, CNAME, ICODE, PANV, MODFUN, SYSFUN, ORIGIN, NEXEC, NLINES, NCOMNT, IETA1, IETA2, NETA1, NETA2, NIOVAR, NDECIS, NFUNCT, NIO, NASGN, NCALL, NFMNT, EOF, ERROR, LEN)

ROUTINE: DRDCRF

FUNCTION: Reads one record from the CRF file and converts all data to internal format

CALLING SEQUENCE:

CALL DRDCRF (MUNIT, FORMNO, PROJNO, PROGNO, FDATE, NCH, NEXAM, OVER1, DATRET, DATBEG, EFFORT, CHTYPE, CHCOMP, ERRNTYP, ERRIN, DATERR, LGCERR, ACTVTY, ISOLTM, PATCH, RELOLD, RELNO, RELDAT, CMTREA, CMTDES, CMTGEN, ISTAT, EOF, ERROR)

3.5.3.3 Sort and Search Routines

These two routines perform sort or search functions.

ROUTINE: FILEIT

FUNCTION: Determines if the given name is in the current list and adds it if it is not

CALLING SEQUENCE:

CALL FILEIT (ERRIN, MAXNAM, NAME, NEWCH, NEWERR, NEWNAM, NNEW, ERROR)

3-93
ROUTINE: SORT

FUNCTION: Produces an array of indexes sorted in order based on the given I*2 array

CALLING SEQUENCE:

CALL SORT (I2, NSORT, ISORT)

3.5.3.4 Routines Performing String Operations

These two routines perform string operations.

ROUTINE: MAKNAM

FUNCTION: Concatenates the given strings to form a complete file name

CALLING SEQUENCE:

CALL MAKNAM (DISK, UIC, NAME, EXTENS, DSN)

ROUTINE: SQEEZ

FUNCTION: Removes blanks from a character string

CALLING SEQUENCE:

CALL SQEEZ (IN, NSIZE, NONBL, OUT)

3.5.3.5 Variable Description

The variables in the calling sequences of major CG and REP4 routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANAME</td>
<td>R*8</td>
<td>Component name</td>
</tr>
<tr>
<td>ERRIN</td>
<td>I*2</td>
<td>Flag indicating when error entered system</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>ICHNGF</td>
<td>I*2</td>
<td>Change and error data file (CG intermediate file) unit number</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>ICTDOS</td>
<td>I*2</td>
<td>Number of DO and DOWHILE statements</td>
</tr>
<tr>
<td>ICTEXC</td>
<td>I*2</td>
<td>Number of executable statements</td>
</tr>
<tr>
<td>ICFNR</td>
<td>I*2</td>
<td>Number of FUNCTION references</td>
</tr>
<tr>
<td>ICFIFF</td>
<td>I*2</td>
<td>Number of IF and .IF statements</td>
</tr>
<tr>
<td>ICTIO</td>
<td>I*2</td>
<td>Number of I/O statements</td>
</tr>
<tr>
<td>IDECIS</td>
<td>I*2</td>
<td>Number of decisions (McCabe's measure)</td>
</tr>
<tr>
<td>IDNAME(NSORT)</td>
<td>R*8</td>
<td>Array of names of each component</td>
</tr>
<tr>
<td>IREPF</td>
<td>I*2</td>
<td>REP4 output report file unit number</td>
</tr>
<tr>
<td>ISORT(NSORT)</td>
<td>I*2</td>
<td>Sorted index array</td>
</tr>
<tr>
<td>ITYPE(NSORT)</td>
<td>I*2</td>
<td>Function type of each component</td>
</tr>
<tr>
<td>I2(NSORT)</td>
<td>I*2</td>
<td>Array on which sort is based</td>
</tr>
<tr>
<td>KASGN</td>
<td>I*2</td>
<td>Number of assignment statements</td>
</tr>
<tr>
<td>KCALL</td>
<td>I*2</td>
<td>Number of CALLs</td>
</tr>
<tr>
<td>KFMT</td>
<td>I*2</td>
<td>Number of FORMAT statements</td>
</tr>
<tr>
<td>LUNDB</td>
<td>I*2</td>
<td>CIF unit number</td>
</tr>
<tr>
<td>LUNIT</td>
<td>I*2</td>
<td>Unit number associated with the CIF</td>
</tr>
<tr>
<td>MAXNAM</td>
<td>I*2</td>
<td>Maximum number of component names allowed in name array</td>
</tr>
<tr>
<td>MUNIT</td>
<td>I*2</td>
<td>Unit number associated with the CRF file</td>
</tr>
<tr>
<td>NAME</td>
<td>R*8</td>
<td>Component name</td>
</tr>
<tr>
<td>NCHS</td>
<td>I*2</td>
<td>Number of changes for the given component</td>
</tr>
<tr>
<td>NERRS</td>
<td>I*2</td>
<td>Number of errors for the given component</td>
</tr>
<tr>
<td>NEWCH(NNEW)</td>
<td>I*2</td>
<td>Array of number of changes for each component</td>
</tr>
<tr>
<td>NEWERR(NNEW)</td>
<td>I*2</td>
<td>Array of number of errors for each component</td>
</tr>
<tr>
<td>NEWNAM(NNEW)</td>
<td>R*8</td>
<td>Array of component names identified in the CRF file</td>
</tr>
<tr>
<td>NNEW</td>
<td>I*2</td>
<td>Number of components identified in the CRF file</td>
</tr>
<tr>
<td>NSORT</td>
<td>I*2</td>
<td>Number of records to be sorted</td>
</tr>
</tbody>
</table>

3-95

8818
### Name Type Description

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUNIT</td>
<td>I*2</td>
<td>Unit number of the CG intermediate file</td>
</tr>
<tr>
<td>OUTDSN(25)</td>
<td>L*1</td>
<td>File name of the CG intermediate file</td>
</tr>
<tr>
<td>PASGN</td>
<td>R*4</td>
<td>Percent of assignment statements</td>
</tr>
<tr>
<td>PCALL</td>
<td>R*4</td>
<td>Percent of CALLs</td>
</tr>
<tr>
<td>PDEC</td>
<td>R*4</td>
<td>Percent of decisions</td>
</tr>
<tr>
<td>PDOS</td>
<td>R*4</td>
<td>Percent of DO and DOWHILE statements</td>
</tr>
<tr>
<td>PPUNC</td>
<td>R*4</td>
<td>Percent of FUNCTION references</td>
</tr>
<tr>
<td>PIFS</td>
<td>R*4</td>
<td>Percent of IF and .IF statements</td>
</tr>
<tr>
<td>PIO</td>
<td>R*4</td>
<td>Percent of I/O plus FORMAT statements</td>
</tr>
<tr>
<td>PREFIX</td>
<td>I*2</td>
<td>Subsystem prefix given by user</td>
</tr>
<tr>
<td>PROJ(8)</td>
<td>L*1</td>
<td>Project name</td>
</tr>
<tr>
<td>PTOTS</td>
<td>R*4</td>
<td>Percent of CALL statements plus FUNCTION references</td>
</tr>
<tr>
<td>TOTCH</td>
<td>I*2</td>
<td>Total number of changes and errors</td>
</tr>
<tr>
<td>ZPROJ(NSORT)</td>
<td>I*2</td>
<td>Subsystem prefix for each component</td>
</tr>
</tbody>
</table>

### 3.5.4 TASK BUILD PROCEDURE

#### 3.5.4.1 Command Procedures

The CG program can be generated from the source code by executing the command procedure CGGEN.CMD under UIC [204,6] (Figure 3-15). CGGEN.CMD references another command procedure, CG.TKB, under UIC [204,6], which builds the task image for the CG program.

The REP4 program can be generated from the source code by executing the command procedure R4GEN.CMD under UIC [204,6] (Figure 3-16). Three other command procedures, R4FPP.CMD, R4FOR.CMD, and R4.TKB, under UIC [204,6], are referenced by this command procedure.
Figure 3-15. CG Task Generation Command Procedure
(CCGEN.CMD)
Figure 3-16. REP4 Task Generation Command Procedure (R4GEN.CMD) (1 of 2)
FOR/F4P/OBJECT:[204.6]R5CTYPE [204.6]R5CTYPE
ROUTINES WITH PREFIX UT
FOR/F4P/OBJECT:[204.7]UTDRDCIF [204.7]UTDRDCIF
FOR/F4P/OBJECT:[204.7]UTMAKNAM [204.7]UTMAKNAM
FOR/F4P/OBJECT:[204.7]UTSQEZ [204.7]UTSQEZ
BUILD THE REP4 TASK IMAGE

Figure 3-16. REP4 Task Generation Command Procedure (R4GEN.CMD) (2 of 2)
The CG program is generated by entering the following command:

@CGGEN

The REP4 program is generated by entering this command:

@R4GEN

3.5.4.2 Overlay Structure

The CG and REP4 programs are both overlaid to reduce the memory space requirement. The files containing the Overlay Descriptor Language needed to generate the task images for these two programs are CG.ODL and R4.ODL, respectively. Figure 3-17 is a listing of CG.ODL; Figure 3-18 is R4.ODL. The system libraries RMS11M.ODL and RMS12X.ODL are needed for both overlays. In addition, the RMS Indexed Access Program Library (RMSIAC) is needed in both overlays. The name of the library is UFRMSIAC.OLB. It contains the FORTRAN routines necessary to access RMS indexed files.
Figure 3-17. CG Program Overlay Descriptor Language File (CG.ODL)

Figure 3-18. REP4 Program Overlay Descriptor Language File (R4.ODL)
3.6 COMPONENT INFORMATION REPORT PROGRAM (REP5)

3.6.1 INTRODUCTION

The Component Information Report Program (REP5) produces a list of components and associated data for a given project. For each component, basic data from the Component Information File (CIF), Halstead parameters computed from the basic data, and the change and error data retrieved from the CG intermediate file produced by the CG program (Section 3.5) are reported. Correlation coefficients between the various statistics presented are also given.

3.6.2 PROGRAM STRUCTURE

3.6.2.1 Files Accessed

The REP5 program accesses two input files, one output file, and one scratch file, as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CIF</td>
<td>CIF for the given project</td>
</tr>
<tr>
<td>&lt;PRJNAM&gt;.CHN</td>
<td>CG intermediate file containing change and error data produced by the CG program for the given project</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;PRJNAM&gt;.RP5</td>
<td>REP5 output report for the given project</td>
</tr>
</tbody>
</table>

A scratch file is used by the REP5 program to temporarily store data that will later be used to compute the correlation coefficient matrix.

For these file names, <PRJNAM> is the name of the project selected by the user.

3.6.2.2 Baseline Diagram

Figure 3-19 is the baseline diagram for the REP5 program. The REP5 routine is the driver that obtains the project name and selected subsystem, reads the CG intermediate file and
Figure 3-19. Baseline Diagram for the Component Information Report Program (REP5)
CIF for the given project, computes the Halstead parameters, and writes the output report. REP5 loops through the above process until a^{Z} (control Z) is returned in response to a subsystem prefix prompt.

3.6.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines referenced by the REP5 program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major REP5 routines are described in Section 3.6.3.4. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines in this section, the REP5 program also uses the following system routines: DATE, SECNDS, and TIME.

3.6.3.1 Process Data and Compute Statistics

These five major routines obtain data from a given CIF and compute statistics for the output report.

ROUTINE: COEF

FUNCTION: Computes the correlation coefficient matrix for a given set of variables

CALLING SEQUENCE:

CALL COEF (ISCRAH, IREPF, NUM, IREC, TITLE)

ROUTINE: CTYPE

FUNCTION: Determines the function type of a component

CALLING SEQUENCE:

CALL CTYPE (ICTEXC, ICTFNR, ICTIO, KASGN, KCALL, KFMT, ITYPE)
ROUTINE: ESTIM
FUNCTION: Computes the values of several Halstead parameters
CALLING SEQUENCE:

CALL ESTIM (ICTHIO, IETA1, IETA2, NETA1, NETA2,
   IETA, NETA, LENGTH, IVOL, PRGLVL, ALNGLV,
   IEFORT, TOTIM, NBUGS, IVSTAR, STROUD, ERROR)

ROUTINE: GETRPT
FUNCTION: Extracts pertinent data from the CIF and writes it to the output report
CALLING SEQUENCE:

CALL GETRPT (LUNDB, ITERMF, IREPF, ISCRAH, ICHNGF,
   PROJNM)

ROUTINE: REP5
FUNCTION: Main routine of the REP5 program, extracts data from the CIF and from the CG intermediate file, computes statistics, and writes the output report
CALLING SEQUENCE: None

3.6.3.2 File Open and Read Routines
These two routines either open an indexed file or read records from a file.

ROUTINE: CHNGES
FUNCTION: Reads the CG intermediate file and returns the number of changes and errors for a given component name; if the component name is not found, the routine returns 999 for the output variables
CALLING SEQUENCE:

CALL CHNGES (ANAME, ICHNGF,
   NCHS, NERRS, TOTCH)
ROUTINE: REDCIF

FUNCTION: Reads one record from the CIF and converts all data to internal format

CALLING SEQUENCE:

CALL REDCIF (LUNDB,
PROJNO, CNAME, ICODE, PANV, MODFUN, SYSFUN,
ORIGIN, NEXEC, NLINES, NCOMNT, IETA1, IETA2,
NETA1, NETA2, NIOVAR, MCCABE, NFUNCT, NIO,
NASGN, NCALL, NFMT, EOF, ERROR)

3.6.3.3 Routine Performing String Operations

This routine performs a string operation.

ROUTINE: MAKNAM

FUNCTION: Concatenates the given strings to form a complete file name

CALLING SEQUENCE:

CALL MAKNAM (DISK, UIC, NAME, EXTENS, DSN)

3.6.3.4 Variable Description

The variables in the calling sequences of major REP5 routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALNGLV</td>
<td>R*4</td>
<td>Language level</td>
</tr>
<tr>
<td>ANAME</td>
<td>R*8</td>
<td>Component name</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>ICHNGF</td>
<td>I*2</td>
<td>Change and error data file (CG intermediate file) unit number</td>
</tr>
<tr>
<td>ICTEXC</td>
<td>I*2</td>
<td>Number of executable statements</td>
</tr>
<tr>
<td>ICTFNR</td>
<td>I*2</td>
<td>Number of function references</td>
</tr>
<tr>
<td>ICHTIO</td>
<td>I*2</td>
<td>Number of input and output variables for component</td>
</tr>
</tbody>
</table>

3-106

8818
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICTIO</td>
<td>I*2</td>
<td>Number of I/O statements</td>
</tr>
<tr>
<td>IEFORT</td>
<td>I*4</td>
<td>Effort required</td>
</tr>
<tr>
<td>IETA</td>
<td>I*2</td>
<td>Number of unique elements</td>
</tr>
<tr>
<td>IETAL</td>
<td>I*2</td>
<td>Number of unique operators</td>
</tr>
<tr>
<td>IETA2</td>
<td>I*2</td>
<td>Number of unique operands</td>
</tr>
<tr>
<td>IREC</td>
<td>I*2</td>
<td>Total number of records in file</td>
</tr>
<tr>
<td>IREPF</td>
<td>I*2</td>
<td>Unit number associated with the REPS output report file</td>
</tr>
<tr>
<td>ISCRAH</td>
<td>I*2</td>
<td>Unit number associated with the scratch file</td>
</tr>
<tr>
<td>ITERMF</td>
<td>I*2</td>
<td>Unit number associated with the terminal</td>
</tr>
<tr>
<td>ITYPE</td>
<td>I*2</td>
<td>Component function type</td>
</tr>
<tr>
<td>IVOL</td>
<td>I*2</td>
<td>Program volume</td>
</tr>
<tr>
<td>IVSTAR</td>
<td>I*2</td>
<td>Potential volume</td>
</tr>
<tr>
<td>KASGN</td>
<td>I*2</td>
<td>Number of assignment statements</td>
</tr>
<tr>
<td>KCALL</td>
<td>I*2</td>
<td>Number of CALLs</td>
</tr>
<tr>
<td>KFMT</td>
<td>I*2</td>
<td>Number of FORMAT statements</td>
</tr>
<tr>
<td>LENGTH</td>
<td>I*2</td>
<td>Predicted length</td>
</tr>
<tr>
<td>LUNDB</td>
<td>I*2</td>
<td>Unit number associated with the CIF</td>
</tr>
<tr>
<td>NBUGS</td>
<td>I*2</td>
<td>Predicted number of bugs</td>
</tr>
<tr>
<td>NCHS</td>
<td>I*2</td>
<td>Number of changes for the given component</td>
</tr>
<tr>
<td>NERRS</td>
<td>I*2</td>
<td>Number of errors</td>
</tr>
<tr>
<td>NETA</td>
<td>I*2</td>
<td>Total number of elements</td>
</tr>
<tr>
<td>NETAL</td>
<td>I*2</td>
<td>Total number of operators</td>
</tr>
<tr>
<td>NETA2</td>
<td>I*2</td>
<td>Total number of operands</td>
</tr>
<tr>
<td>NUM</td>
<td>I*2</td>
<td>Number of lines of data</td>
</tr>
<tr>
<td>PRGLVL</td>
<td>R*4</td>
<td>Program level</td>
</tr>
<tr>
<td>PROJNM(8)</td>
<td>L*1</td>
<td>Project name</td>
</tr>
<tr>
<td>STROUD</td>
<td>I*4</td>
<td>Stroud number (discriminations per hour)</td>
</tr>
<tr>
<td>TITLE(10)</td>
<td>R*8</td>
<td>Arrays of column titles</td>
</tr>
</tbody>
</table>
### Name | Type | Description
--- | --- | ---
TOTCH | I*2 | Total number of changes and errors
TOTIM | R*4 | Total programming time required

#### 3.6.4 TASK BUILD PROCEDURE

#### 3.6.4.1 Command Procedures

The REP5 program can be generated from the source code by executing the command procedure R5GEN.CMD under UIC [204,6]. This command procedure references three command procedures—R5FPP.CMD, R5FOR.CMD, and R5.TKB—all under UIC [204,6]. Figure 3-20 is a listing of R5GEN.CMD, the command procedure to precompile, compile, and task build the REP5 program. The REP5 program is generated by entering the following command:

```
@ [204,6] R5GEN
```

#### 3.6.4.2 Overlay Structure

The REP5 program is overlaid to reduce the memory space requirement. Figure 3-21 is a listing of the Overlay Descriptor Language file, [204,6]R5.ODL, needed to build the REP5 program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay.
Figure 3-20. Task Generation Command Procedure for the REP5 Program (R5GEN.CMD)
Figure 3-21. REP5 Program Overlay Descriptor Language File (R5.ODL)
3.7 GRAPHING PROGRAM (GQ)

3.7.1 INTRODUCTION

The Graphing Program (GQ) reads an external data file containing a set of points and produces a graph of the data. It also optionally fits a polynomial of degree less than or equal to 10 to the given set of points and computes various associated statistics.

3.7.2 PROGRAM STRUCTURE

3.7.2.1 Files Accessed

The GQ program accesses two input files and one output file as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,6]GQ.NL</td>
<td>GQ input parameters file</td>
</tr>
<tr>
<td>&lt;PRJNAM&gt;.XXX</td>
<td>External file containing project name, X-axis title, Y-axis title, and a set of X, Y values for the points to be plotted. The file name for the external data file is of the form &lt;PRJNAM&gt;.XXX if produced by the PF or the WK program, where &lt;PRJNAM&gt; is the name of the project for which the program was executed and XXX denotes the type of data (Sections 2.2.3 and 2.4.3). If generated by the user, the file name is arbitrary.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOR0XX.DAT</td>
<td>Output graph and statistics report (XX is the output unit number specified in the GQ input parameters file).</td>
</tr>
</tbody>
</table>

3.7.2.2 Baseline Diagram

Figure 3-22 is the baseline diagram for the GQ program. The GRFDRV routine is the main driver. It reads the GQ input parameters file, initializes the user's terminal, reads the external data file, and produces a graph of the given data.
Figure 3-22. Baseline Diagram for the Graphing Program (GQ) (2 of 2)
3.7.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The subroutines forming the GQ program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major GQ routines are described in Section 3.7.3.6. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the GQ program also uses the following system routines: CLEAR, DATE, DISCKP, GETLUN, TIME, WAIT, and WTQIO.

3.7.3.1 Process Data and Compute Statistics

These 13 major routines obtain data from the external data file, compute statistics, and produce the graph.

ROUTINE: FLAG (LOGICAL FUNCTION)

FUNCTION: Sets a given character to the flag character if the given character is blank

CALLING SEQUENCE:

    FLAG (CHAR, QFLAG)

ROUTINE: GRFDRV

FUNCTION: Main routine of the GQ program, reads a file containing a set of points and produces a graph of the data

CALLING SEQUENCE: None

ROUTINE: GTNOIS

FUNCTION: Computes a noise value from the data points
CALLING SEQUENCE:

CALL GTNOIS (NPTS, QFLAG, Y, QCHARS, AVNOIS, ERROR)

ROUTINE: INVERT

FUNCTION: Inverts a matrix in place and solves a set of simultaneous linear equations

CALLING SEQUENCE:

CALL INVERT (A, B, N, L, C, IER)

ROUTINE: LOOP

FUNCTION: Computes the minimum chi square and rejects data points outside a specified factor times the standard deviation; also prints a graph and statistics as desired

CALLING SEQUENCE:

CALL LOOP (ADEBUG, DDATE, DIFFAC, IOFFSE, IOP, IPAGE, IPART1, IPART4, IPR, IWID, KCYCLE, KXSHFT, MCOIN, M_LINES, MXFRAC, MXITER, MXORDR, NAV1, NAV2, NPTS, NSTREK, PROJ, QBAND, QBEST, QCHARS, QCHR, QCUM, QCYCLE, QFLAG, QGRAPH, QINTG, QMARKR, QNL, QOMITO, QPRINT, QRESCN, QSCALX, QSCREEN, QSTATS, QTRUNC, RES, RPTITL, SIGFAC, TOL, X, XFACTR, XH, XTITLE, Y, YDFAC, YFACTR, YH, YLOW, YTITLE, CHIONE, COEF, MCO, STDV)

ROUTINE: POLYFT

FUNCTION: Performs a least-squares polynomial fit to a set of data points
CALLING SEQUENCE:

CALL POLYFT (X, Y, NPTS, MCOEF, TOL, QFLAG, QCHARS,
CHI, COEF, RES, STDV, SUMABS, SUMMR2, SUMR2,
XMEAN, IER)

ROUTINE: RESCAN

FUNCTION: Checks to determine if points should be flagged or unflagged

CALLING SEQUENCE:

CALL RESCAN (COEF, MCO, NPTS, QCHR1, QFLAG, RES, STDV,
X, Y,
QCHARS)

ROUTINE: SCAN1

FUNCTION: Performs a preliminary scan on the data and flags those points obviously out of a reasonable range

CALLING SEQUENCE:

CALL SCAN1 (DIFFAC, NPTS, QCHR1, QFLAG, Y, NSTREK,
QCHARS,
AVNOIS, ERROR)

ROUTINE: SCAN2

FUNCTION: Cycles through all points (ignoring previously flagged points) and computes the average Y-values for the previous NPTS points and the succeeding NPTS points; flags the current point if the difference between its Y-value and these averages exceeds a specified tolerance

CALLING SEQUENCE:

CALL SCAN2 (AVNOIS, MXFRAC, MXITER, NAV1, NAV2, NPTS,
NSTREK, QCHR1, QFLAG, Y, YDFAC,
QCHARS,
YDFAC2, ERROR)

3-116

8818
ROUTINE: SCRDIF

FUNCTION: Computes the average difference in Y-values for all data points and flags data points whose difference from the previous point and subsequent point varies more than a given factor times the average difference.

CALLING SEQUENCE:

CALL SCRDIF (DIFFAC, MXFRAC, MXITER, NAV1, NAV2, NPTS, NSTREK, QCHR1, QFLAG, Y, YDFAC, QCHARS, AVNOIS, YDFAC2, ERROR)

ROUTINE: TF (LOGICAL FUNCTION)

FUNCTION: Returns a value of .TRUE. if the input number is not zero.

CALLING SEQUENCE:

TF (N)

ROUTINE: WRKDAT

FUNCTION: Takes the given X and Y arrays and manipulates and scales the data as desired by the given input parameters; also computes several statistics related to the standard deviation.

CALLING SEQUENCE:

CALL WRKDAT (ADEBUG, CHIONE, DIFFAC, IOFFSE, IPART4, MCO, MXFRAC, MXITER, NAV1, NAV2, NSTREK, QBAND, QCHR, QFLAG, QOMIT0, QSCALX, QSCREN, QTURNC, SIGFAC, TOL, YDFAC, YFACTR, NPTS, QCHARS, QMARKR, X, Y, AFRAC, AREA1, AREA2, AVNOIS, CHI, COEF, KZEROS, NPTPLT, NPREJ, RES, STDV, SUMABS, SUMMR2, SUMR2, XFACTR, XMEAN, YDFAC2, ERROR)
ROUTINE: YVALUE (REAL FUNCTION)

FUNCTION: Computes the Y-value associated with a given X-value for the polynomial with the given coefficients and degree

CALLING SEQUENCE:

YVALUE (COEF, MCO, XVAL)

3.7.3.2 Print a Graph and Statistics Report

These five routines produce a graph and statistics chart of the given data.

ROUTINE: GRAPH

FUNCTION: Generates a one-page Cartesian printer plot for any set of data with automatic scaling

CALLING SEQUENCE:

CALL GRAPH (IOPT, IPR, IWID, KXSHFT, MLINES, N, N2, QCHARS, QMARKR, QXTITL, QYITL, X, XH, XL, Y, YH, YL,)
LINES)

ROUTINE: HEADR2

FUNCTION: Prints a one-line title for each report page that includes the date and project name

CALLING SEQUENCE:

CALL HEADR2 (IRPTF, PRJNAM, RPTITL, IPAGE)

ROUTINE: PHSRPT

FUNCTION: Prints phase date information on the first page of the graphing report

CALLING SEQUENCE:

CALL PHSRPT (IPR, NPTS, QMARKR, T10)

3-118

8818
ROUTINE: REPORT

FUNCTION: Produces a graph and statistical chart of the given data

CALLING SEQUENCE:

CALL REPORT (QBAND, QBEST, QCUM, QCYCLE, QGRAPH, QINTG, QNL, QOMITO, QPRINT, QSCREEN, QSTATS, QTRUNC, QRESCN)

ROUTINE: REPOR2

FUNCTION: An ENTRY point of routine REPORT

CALLING SEQUENCE:

CALL REPOR2 (AFRAC, AREA1, AREAZ, AVIlOIS, CHI, COEF, DDATE, DIFFAC, IOFFSE, IOPT, IPAGE, IPART1, IPART4, IPR, IRES, IWID, KCYCLE, KXSHFT, KZEROS, MCO, MLINES, MXFRAC, MXITER, MXORDR, NAV1, NAV2, NPTPLT, NPTREJ, NPTS, NSTREK, PROJ, QCHARS, QFLAG, QMARKR, RES, RPTITL, SIGFAC, STDV, SUMABS, SUMR2, SUMR2, TOL, X, XFACTR, XH, XMEAN, XTITLE, Y, YDFAC, YDFAC2, YFACTR, YH, YLOW, YTITLE)

3.7.3.3 Obtain Data From Terminal or External Data Set

These four routines obtain information from a user's response to a terminal prompt or from an external data set.

ROUTINE: GETFLD

FUNCTION: Displays the given text on the terminal and prompts for a character string

CALLING SEQUENCE:

CALL GETFLD (TEXT, EXTFIL, FLDLEN, TERMNL, EOFTTY, ERROR, FIELD)
ROUTINE: GETNL2

FUNCTION: Reads a sequential file and fills a parameter array

CALLING SEQUENCE:

CALL GETNL2 (NLDSN, NLFIL, MAXNL, NL, ERROR)

ROUTINE: GETOPT

FUNCTION: Retrieves user options for the current run

CALLING SEQUENCE:

CALL GETOPT (IPR, IWID, MCO, M_LINES, QCHR, QDSN, QEOF)

ROUTINE: RDPLT3

FUNCTION: Reads an external data file for X and Y values and X and Y axis titles

CALLING SEQUENCE:

CALL RDPLT3 (IPLTF, PLTNAM, EXTFIL, MAXREC, QCUM, QMAKEX, TERMNL, CHAR, PROJ, RPTITL, PIETTL, X, Y, NCOUNT, XHI, XTITLE, YHI, YTITLE, MARKER, DDATE, FACTRY, EOFTTY, ERROR)

3.7.3.4 Routines for String Movement or Comparison

These five routines deal with string movement or comparison.

ROUTINE: BLANK

FUNCTION: Initializes an array to blanks

CALLING SEQUENCE:

CALL BLANK (ARRAY, NUM)
ROUTINE: CHRINT

FUNCTION: Converts the given string to integers in I*2 format

CALLING SEQUENCE:

CALL CHRINT (CHARS, NCHAR, I2NUM, ERROR)

ROUTINE: MOVE

FUNCTION: Moves a given number of bytes from one address to another

CALLING SEQUENCE:

CALL MOVE (A, B, LEN)

ROUTINE: NAME3

FUNCTION: Concatenates the given strings to form a complete file name

CALLING SEQUENCE:

CALL NAME3 (DISK, UIC, NAME, EXTENS, DSN)

ROUTINE: WHERE

FUNCTION: Locates the given characters in the given string

CALLING SEQUENCE:

CALL WHERE (CHAR, STRING, LEN, LOC, FOUND)

3.7.3.5 Plot Routines

These 12 routines deal with plotting the graph on the terminal or graphing equipment.
ROUTINE:  KCHRAT (LOGICAL FUNCTION)
FUNCTION:  Obtains the character at the given point
CALLING SEQUENCE:

    KCHRAT (X, Y)

ROUTINE:  KEND
FUNCTION:  Finishes production of a graph and prints the developed grid
CALLING SEQUENCE:

    CALL KEND (LINES)

ROUTINE:  KLINE
FUNCTION:  Writes the given character string to the current file, terminal, or IIS graphics device
CALLING SEQUENCE:

    CALL KLINE (X, Y, DIR, LEN, CHARS)

ROUTINE:  KSTART
FUNCTION:  Initializes the screen or IIS graphics device and a grid for a plot
CALLING SEQUENCE:

    CALL KSTART

ROUTINE:  SCALE
FUNCTION:  Chooses the best scale for plotting any set of data
CALLING SEQUENCE:

    CALL SCALE (XMIN, XMAX, NMAX,
           XI, DX, NX, NDECX, NDIGX)

3-122

8818
ROUTINE: TBLINK
FUNCTION: Turns on the blink function of the VT100 terminal
CALLING SEQUENCE:
    CALL TBLINK

ROUTINE: TCLEAN
FUNCTION: Finishes the production of a graph and prints the developed grid
CALLING SEQUENCE:
    CALL TCLEAN (QGRID, XMAX, YMAX, LINES)

ROUTINE: TCLEAR
FUNCTION: Clears the terminal or IIS graphics device
CALLING SEQUENCE:
    CALL TCLEAR

ROUTINE: TINIT
FUNCTION: Initializes the terminal in preparation for graphics
CALLING SEQUENCE:
    CALL TINIT

ROUTINE: TNOBLI
FUNCTION: Turns off the blink option of the VT100 terminal
CALLING SEQUENCE:
    CALL TNOBLI
ROUTINE: TPOINT

FUNCTION: Writes the given characters starting at the given point

CALLING SEQUENCE:

CALL TPOINT (X, Y, DIR, LEN, CHARS)

ROUTINE: TTERM

FUNCTION: Changes the default terminal number

CALLING SEQUENCE:

CALL TTERM (JTERM)

3.7.3.6 Variable Description

The variables in the calling sequences of major GQ routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADEBUG(80)</td>
<td>I*2</td>
<td>Debug array</td>
</tr>
<tr>
<td>AFRAC</td>
<td>R*4</td>
<td>Area under computed curve divided by area under actual data</td>
</tr>
<tr>
<td>AREAL</td>
<td>R*4</td>
<td>Area under computed curve</td>
</tr>
<tr>
<td>AREA2</td>
<td>R*4</td>
<td>Area under actual data (including flagged points)</td>
</tr>
<tr>
<td>AVNOIS</td>
<td>R*4</td>
<td>Average noise value</td>
</tr>
<tr>
<td>CHAR</td>
<td>L*1</td>
<td>A given character</td>
</tr>
<tr>
<td>CHI</td>
<td>R*8</td>
<td>Chi square</td>
</tr>
<tr>
<td>CHIONE</td>
<td>L*1</td>
<td>Flag indicating if first attempt to fit polynomial</td>
</tr>
<tr>
<td>COEF(10)</td>
<td>R*8</td>
<td>Coefficients of fit</td>
</tr>
<tr>
<td>DDATE(9)</td>
<td>L*1</td>
<td>Date of data</td>
</tr>
<tr>
<td>DIFPAC</td>
<td>R*4</td>
<td>Difference factor</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>IOFFSE</td>
<td>I*2</td>
<td>Parameter that forces start and end of curve fit to data to 0, if 1; if 0, does not force curve to 0</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IOPT</td>
<td>I*2</td>
<td>Parameter to plot count of overlapping points, if 1; if 0, does not plot count</td>
</tr>
<tr>
<td>IPAGE</td>
<td>I*2</td>
<td>Current page number</td>
</tr>
<tr>
<td>IPART1</td>
<td>I*2</td>
<td>Maximum number of points allowed</td>
</tr>
<tr>
<td>IPART4</td>
<td>I*2</td>
<td>Size of X, Y, and character arrays (4 * IPART1)</td>
</tr>
<tr>
<td>IPR</td>
<td>I*2</td>
<td>Output unit number</td>
</tr>
<tr>
<td>IRES</td>
<td>I*2</td>
<td>Number of reject cycle</td>
</tr>
<tr>
<td>IWID</td>
<td>I*2</td>
<td>Width of graph in columns, including titles</td>
</tr>
<tr>
<td>KCYCLE</td>
<td>I*2</td>
<td>Number of times to cycle through data rejecting flagged points</td>
</tr>
<tr>
<td>KXSHFT</td>
<td>I*2</td>
<td>Column to start graph</td>
</tr>
<tr>
<td>KZEROS</td>
<td>I*2</td>
<td>Number of trailing zero data points flagged</td>
</tr>
<tr>
<td>MCO</td>
<td>I*2</td>
<td>Order of fit desired</td>
</tr>
<tr>
<td>MCOIN</td>
<td>I*2</td>
<td>Minimum order of polynomial to be fit to data</td>
</tr>
<tr>
<td>M LINES</td>
<td>I*2</td>
<td>Number of rows allowed in graph</td>
</tr>
<tr>
<td>MXFRAC</td>
<td>R*4</td>
<td>Maximum fraction of flagged points</td>
</tr>
<tr>
<td>MXITER</td>
<td>I*2</td>
<td>Maximum number of iterations</td>
</tr>
<tr>
<td>MXORDR</td>
<td>I*2</td>
<td>Maximum order of polynomial to be fit to data</td>
</tr>
<tr>
<td>NAV1</td>
<td>I*2</td>
<td>Number of preceding points to consider</td>
</tr>
<tr>
<td>NAV2</td>
<td>I*2</td>
<td>Number of succeeding points to consider</td>
</tr>
<tr>
<td>NPTBLT</td>
<td>I*2</td>
<td>Number of points plotted</td>
</tr>
<tr>
<td>NPTREJ</td>
<td>I*2</td>
<td>Number of points flagged (rejected)</td>
</tr>
<tr>
<td>NPTS</td>
<td>I*2</td>
<td>Number of data points</td>
</tr>
<tr>
<td>NSTREK</td>
<td>I*2</td>
<td>Maximum number of consecutive flagged points allowed</td>
</tr>
<tr>
<td>PROJ(8)</td>
<td>L*1</td>
<td>Project name</td>
</tr>
<tr>
<td>QBAND</td>
<td>L*1</td>
<td>Flag indicating whether to plot band around fitted curve</td>
</tr>
<tr>
<td>QBEST</td>
<td>L*1</td>
<td>Flag indicating whether program is to find polynomial of best fit</td>
</tr>
<tr>
<td>QCHARS</td>
<td>L*1</td>
<td>Array of characters to be plotted</td>
</tr>
<tr>
<td>(NPTS)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3-125

8818
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QCHR(4)</td>
<td>L*1</td>
<td>Characters to be used:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 1, Data points</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 2, Upper edge of band around curve</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 3, Lower edge of band around curve</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 4, Curve fit to data points</td>
</tr>
<tr>
<td>QCHRL1</td>
<td>L*1</td>
<td>Data point character for unflagged points</td>
</tr>
<tr>
<td>QCUM</td>
<td>L*1</td>
<td>Flag indicating whether to accumulate data as it is read in</td>
</tr>
<tr>
<td>QCYCLE</td>
<td>L*1</td>
<td>Flag indicating whether to print graph report each time through reject cycle</td>
</tr>
<tr>
<td>QDSN(27)</td>
<td>L*1</td>
<td>Name of file to be read</td>
</tr>
<tr>
<td>QEOF</td>
<td>L*1</td>
<td>End of file flag</td>
</tr>
<tr>
<td>QFLAG</td>
<td>L*1</td>
<td>Flag character</td>
</tr>
<tr>
<td>QGRAPH</td>
<td>L*1</td>
<td>Flag indicating whether to print graph page</td>
</tr>
<tr>
<td>QINTG</td>
<td>L*1</td>
<td>Flag indicating whether to print data as integers on last page of report</td>
</tr>
<tr>
<td>QMARKR (IPART1)</td>
<td>L*1</td>
<td>Array of characters to be printed at bottom of graph (phase characters)</td>
</tr>
<tr>
<td>QNL</td>
<td>L*1</td>
<td>Flag indicating whether to print input parameter (first) page of report</td>
</tr>
<tr>
<td>QOMITO</td>
<td>L*1</td>
<td>Flag indicating whether to ignore zero data points</td>
</tr>
<tr>
<td>QPRINT</td>
<td>L*1</td>
<td>Flag indicating whether to print graph report each cycle through curve fitting process</td>
</tr>
<tr>
<td>QRESCN</td>
<td>L*1</td>
<td>Flag indicating whether to recheck editing of data and fitting of polynomial</td>
</tr>
<tr>
<td>QSCALX</td>
<td>L*1</td>
<td>Flag indicating whether to scale X data points</td>
</tr>
<tr>
<td>QSCREEN</td>
<td>L*1</td>
<td>Flag indicating whether to screen data points relative to preceding and succeeding points</td>
</tr>
<tr>
<td>QSTATS</td>
<td>L*1</td>
<td>Flag indicating whether to print statistics page of report</td>
</tr>
<tr>
<td>QTRUNC</td>
<td>L*1</td>
<td>Flag indicating whether to truncate zero data points at end of array</td>
</tr>
<tr>
<td>RES (NPTS)</td>
<td>R*4</td>
<td>Residuals from curve fit to data</td>
</tr>
</tbody>
</table>
### Name | Type | Description
--- | --- | ---
RPTITL(40) | L*1 | Report title
SIGFAC | R*4 | Sigma factor used to plot band around curve fit to data
STDV | R*8 | Standard deviation
SUMABS | R*8 | Sum of absolute residuals
SUMMR2 | R*8 | Sum of minimum residuals squared
SUMR2 | R*8 | Sum of residuals squared
TOL | R*4 | Tolerance of data
T10 | I*2 | Tab location of printed information
X(NPTS) | R*4 | X data values
XFACTR | R*4 | X scaling factor
XH | R*4 | X axis maximum
XMEAN | R*8 | Mean Y value
XTITLE(40) | L*1 | X axis title
XVAL | R*4 | X value
Y(NPTS) | R*4 | Y data values
YDFAC | R*4 | Y delta factor
YDFAC2 | R*4 | Final prescan boundary factor
YFACTR | R*4 | Y scaling factor
YH | R*4 | Y axis maximum
YLOW | R*4 | Y axis minimum
YTITLE(40) | L*1 | Y axis title

### 3.7.4 TASK BUILD PROCEDURE

#### 3.7.4.1 Command Procedures

The GQ task can be generated from the source code by executing the command procedure GQGEN.CMD under UIC [204,6]. This command procedure references three command files--GQFP.PP.CMD, GQFOR.CMD, and GQ.TKB--all under UIC [204,6]. Figure 3-23 is a listing of GQGEN.CMD, the command procedure to pre-compile, compile, and task build the GQ program. The GQ program is generated by entering the following command:

```
@[204,6]GQGEN
```
Figure 3-23. GQ Task Generation Command Procedure (GQGEN.CMD) (1 of 3)
Figure 3-23. GQ Task Generation Command Procedure
(GQGEN.CMD) (2 of 3)
Figure 3-23. GQ Task Generation Command Procedure (GQGEN.CMD) (3 of 3)
3.7.4.2 Overlay Structure

The GQ program is overlaid to reduce the memory space requirement. Figure 3-24 is a listing of the Overlay Descriptor Language file, [204,6]GQ.ODL, needed to build the GQ program task image.
Figure 3-24. GQ Program Overlay Descriptor Language File (GQ.ODL)
3.8 FORM COUNTER PROGRAM (NF)

3.8.1 INTRODUCTION

The Form Counter Program (NF) produces a report containing counts of forms in the SEL data base files for a given project. Counts are given by form type and programmer for the following types of forms: Change Report Form (CRF), Component Summary Form (CSF), Component Status Report (CSR), Run Analysis Form (RAF), and Resource Summary Form (RSF).

3.8.2 PROGRAM STRUCTURE

3.8.2.1 Files Accessed

The NF program accesses eight input files and one output file as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]ENCODE.HDR</td>
<td>Encoding Dictionary (ENC) File</td>
</tr>
<tr>
<td>[204,1]EST.HDR</td>
<td>EST file</td>
</tr>
<tr>
<td>[204,1]HEADER.HDR</td>
<td>HDR file</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CRF</td>
<td>CRF file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSF</td>
<td>CSF file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSR</td>
<td>CSR file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RAF</td>
<td>RAF file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RSF</td>
<td>RSF file for the given project</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;PRJNAM&gt;.NF</td>
<td>Output report for the given project</td>
</tr>
</tbody>
</table>

In these file names, <PRJNAM> is the name of the project selected by the user.

3.8.2.2 Baseline Diagram

Figure 3-25 is the baseline diagram for the NF program. The NFORMS routine is the main driver. It obtains the project name; counts all forms on the CRF, CSF, CSR, RAF, and RSF files; and then produces a report of form counts for the given project.
Figure 3-25. Baseline Diagram for the Form Counter Program (NF)
3.8.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines forming the NF program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major NF routines are described in Section 3.8.3.8. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the NF program also uses the following system routines: DATE, ERRSET, SECNDS, and TIME.

3.8.3.1 Process Data and Count Forms

These seven major routines count all forms on the data base in each file for a given project.

ROUTINE: CRF
FUNCTION: Totals all CRFs

CALLING SEQUENCE:

    CALL CRF (PROJCT,
              ERROR, NPROG, PROG,
              NCRF)

ROUTINE: CSF
FUNCTION: Totals all CSFs

CALLING SEQUENCE:

    CALL CSF (PROJCT,
              ERROR, NPROG, PROG,
              NCSF)

ROUTINE: CSR
FUNCTION: Totals all CSRs

3-135

8818
CALLING SEQUENCE:

CALL CSR (PROJECT,
            ERROR, NPROG, PROG,
            NCSR)

ROUTINE:  NFORMS
FUNCTION: Main routine of the NF program, counts all forms on the database in each file for the given project

CALLING SEQUENCE:  None

ROUTINE:  RAF
FUNCTION:  Totals all RAFs

CALLING SEQUENCE:

CALL RAF (PROJECT,
            ERROR, NPROG, PROG,
            NRAF)

ROUTINE:  RSF
FUNCTION:  Totals all RSFs

CALLING SEQUENCE:

CALL RSF (PROJECT,
            ERROR, NPROG, PROG,
            NRSF)

ROUTINE:  SUMS
FUNCTION:  Totals all form counts

CALLING SEQUENCE:

CALL SUMS (NPROG, NATM, NCRF, NCSF, NCSR, NGPS, NRAF,
           NRSF,
           NALL, TATM, TCRF, TCSF, TCSR, TGPS, TRAF,
           TRSF, TALL)
3.8.3.2 Write the Form Count Report

These three routines write the report of form counts for the given project.

ROUTINE: FRPT

FUNCTION: Prints a report of form counts of each form type by programmer

CALLING SEQUENCE:

CALL FRPT (NATM, NCRF, NCSF, NCSR, NGPS, NRAF, NRSF, NALL, TATM, TCRF, TCSF, TCSR, TGPS, TRAF, TRSF, TALL, IORDER, NPROG, PROG, PROJCT)

ROUTINE: FSUMRY

FUNCTION: Prints a six-line header summary with data from the HDR and EST files

CALLING SEQUENCE:

CALL FSUMRY (IRPTF, PRJNAM)

ROUTINE: HEADER

FUNCTION: Prints a one-line title for each report page including the date and project name

CALLING SEQUENCE:

CALL HEADER (IRPTF, PRJNAM, RPTITL)

3.8.3.3 Obtain Data From Terminal or External Data Set

These two routines obtain information from a user's response to a terminal prompt or from an external data set.

ROUTINE: FENCA

FUNCTION: Locates the description field on the Encoding Dictionary corresponding to the given type and code
CALLING SEQUENCE:

CALL FENCA (IENCF, TYPE, CODE, 
NAME, REST, FOUND)

ROUTINE: GETFLD

FUNCTION: Displays the given text on the terminal and 
prompts for a given character string

CALLING SEQUENCE:

CALL GETFLD (TEXT, EXTFIL, FLDLEN, 
TERMNL, EOFTTY, ERROR, 
FIELD)

3.8.3.4 Sort and Search Routines

These two routines provide sort and search functions.

ROUTINE: SORT4

FUNCTION: Produces an array of indices sorted according to 
the given I*4 array

CALLING SEQUENCE:

CALL SORT4 (I4, NSORT, 
ISORT)

ROUTINE: STACK

FUNCTION: Determines if the given name is in the current 
list and adds it if it is not

CALLING SEQUENCE:

CALL STACK (MX, PROGNO, 
NPROG, PROG, 
NFRM, ERROR)

3-138

8818
3.8.3.5 File Open and Read Routines

These nine routines either open an indexed file or read records from an indexed file.

ROUTINE: FCRF

FUNCTION: Reads one record from the CRF file and converts the data to internal format

CALLING SEQUENCE:

CALL FCRF (ICRFF, FORMNO, PROJNO, PROGNO, FDATE, NCH, NEXAM, OVER1, DATDET, DATBEG, EFFORT, CHTYPE, CHCOMP, ERRYP, ERRIN, DATERR, LGCERR, ACTVTY, ISOLTM, PATCH, RELOLD, RELNO, RELDAT, CMTREA, CMTDES, CMTGEN, STATUS, EOF, ERROR)

ROUTINE: FCSF

FUNCTION: Reads one record from the CSF file

CALLING SEQUENCE:

CALL FCSF (ICSFF, FORMNO, PROJNO, PROGNO, PROGI, FDATE, FSTAGE, COMPCO, PRECIS, CMPLEX, SWTYP, PASGN, PCNTL, POTHER, STATWO, STMT, BSIZE, INDEP, RELSW, ADDTYP, NCALLD, X1, NCALLG, X2, NSHR, X3, NDESC, X4, LANG1, PLANG1, LANG2, PLANG2, DES, CONSTR, DESRUN, CODRUN, TSTRUN, DESTIM, CODTIM, TSTTIM, DESEFF, CODEFF, TSTEFF, DESDAT, CODDAT, TSTDAT, DESCR, CALLD, CALNG, SHR, AFFECT, OTH, NAMCON, CMT1, CMT2, ISTAT, EOF, ERROR)

ROUTINE: FCSR

FUNCTION: Reads one record from the CSR file using FORTRAN read
CALLING SEQUENCE:

CALL FCSR (ICSRF,
    FORMNO, SEQNO, PROJNO, PROGNO, FDATE, COMPCO,
    TIMES, OTHNAM, OTHOUR, ISTAT, PHASE, EOF,
    ERROR)

ROUTINE: FEST

FUNCTION: Reads one record from the EST file and converts all data to internal format

CALLING SEQUENCE:

CALL FEST (IESTF, NAME,
    PROJ, NCOMP, MODDEL, MODNEW, MODMOD, NRUNS,
    NCHANG, PAGDOC, LINDEL, LINNEW, LINMOD,
    TOTEXT, NEWEXT, MODEXT, PROGHR, MGMT'HR,
    OTHHR, HR95, HR75, OTHCMP, STATUS, ACTIVE,
    PRJCAT, FOUND, ERROR)

ROUTINE: FHDR

FUNCTION: Reads one record from the HDR file using the secondary key (project name)

CALLING SEQUENCE:

CALL FHDR (IHDRF, PRJNAM,
    PROJ, DEVCMP, TARG, ALIEN, RANGES, STATUS,
    ERROR)

ROUTINE: FOPEN

FUNCTION: Opens an indexed file

CALLING SEQUENCE:

CALL FOPEN (IUNIT, FILNAM,
    ERROR)
ROUTINE: FRAF
FUNCTION: Reads one record from the RAF file using FORTRAN read

CALLING SEQUENCE:

CALL FRAF (IRAFF,
    FORMNO, SEQNO, PROJNO, PROGNO, RDATE, MACHIN,
    INTERA, PURPOS, NCOMP, COMPCO, FIRST, METOBJ,
    RESULT, COMENT, ISTAT, EOF, ERROR)

ROUTINE: FREAD
FUNCTION: Reads one indexed record

CALLING SEQUENCE:

CALL FREAD (IUNIT, KEYVAL, KEYLEN, LRECL,
    BUFFER, ERROR)

ROUTINE: FRSF
FUNCTION: Reads one record from the RSF file using FORTRAN read; returns all data on that record converted to internal format plus an array containing each week for which there are data in the record

CALLING SEQUENCE:

CALL FRSF (IRSSF,
    FORMNO, SEQNO, PROJNO, RESCOD, RESID, FDATE,
    PCMGMT, WKDATE, NRUNS, TIMES, STATUS, PHASE,
    LASTWK, EOF, ERROR)

3.8.3.6 Routines for String Movement or Comparison

These three routines deal with string movement or comparison.

ROUTINE: MOVE
FUNCTION: Moves a given number of bytes from one address to another
CALLING SEQUENCE:

CALL MOVE (A, B, LEN)

ROUTINE: NAME3
FUNCTION: Concatenates the given strings to form a complete file name

CALLING SEQUENCE:

CALL NAME3 (DISK, UIC, NAME, EXTENS, DSN)

ROUTINE: WHERE
FUNCTION: Locates the given characters in the given string

CALLING SEQUENCE:

CALL WHERE (CHAR, STRING, LEN, LOC, FOUND)

3.8.3.7 Mathematical Functions

These two routines perform mathematical functions.

ROUTINE: NEXTWK
FUNCTION: Computes the date 1 week after the given date and returns it in MM, DD, YY format

CALLING SEQUENCE:

CALL NEXTWK (DATE, D)

ROUTINE: SUM (INTEGER*2 FUNCTION)
FUNCTION: Computes the sum of all integers in a given array

CALLING SEQUENCE:

SUM (ARRAY, N)
### 3.8.3.8 Variable Description

The variables in the calling sequences of major NF routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARRAY (MX)</td>
<td>I*2</td>
<td>Array of numbers</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>IORDER (MX)</td>
<td>I*2</td>
<td>Sorted index array for programmer numbers</td>
</tr>
<tr>
<td>ISORT (NSORT)</td>
<td>I*2</td>
<td>Sorted index array</td>
</tr>
<tr>
<td>I4 (NSORT)</td>
<td>I*4</td>
<td>Array on which sort is based</td>
</tr>
<tr>
<td>MX</td>
<td>I*2</td>
<td>Number of programmers allowed</td>
</tr>
<tr>
<td>N</td>
<td>I*2</td>
<td>Number of array elements to be summed</td>
</tr>
<tr>
<td>NALL (MX)</td>
<td>I*2</td>
<td>Number of all forms for each programmer</td>
</tr>
<tr>
<td>NATM (MX)</td>
<td>I*2</td>
<td>Number of Attitude Maintenance (ATM) forms for each programmer</td>
</tr>
<tr>
<td>NCRF (MX)</td>
<td>I*2</td>
<td>Number of CRFs for each programmer</td>
</tr>
<tr>
<td>NCSF (MX)</td>
<td>I*2</td>
<td>Number of CSFs for each programmer</td>
</tr>
<tr>
<td>NCSR (MX)</td>
<td>I*2</td>
<td>Number of CSRs for each programmer</td>
</tr>
<tr>
<td>NFRM (MX)</td>
<td>I*2</td>
<td>Number of forms for each programmer for given form type</td>
</tr>
<tr>
<td>NGPS (MX)</td>
<td>I*2</td>
<td>Number of General Project Summary (GPS) forms for each programmer</td>
</tr>
<tr>
<td>NPROG</td>
<td>I*2</td>
<td>Number of programmers found</td>
</tr>
<tr>
<td>NRAF (MX)</td>
<td>I*2</td>
<td>Number of RAFs for each programmer</td>
</tr>
<tr>
<td>NRSF (MX)</td>
<td>I*2</td>
<td>Number of RSFs for each programmer</td>
</tr>
<tr>
<td>NSORT</td>
<td>I*2</td>
<td>Number of entries in array I4</td>
</tr>
<tr>
<td>PROG (MX)</td>
<td>I*4</td>
<td>Array of programmer numbers</td>
</tr>
<tr>
<td>PROGNO</td>
<td>I*4</td>
<td>Given programmer number</td>
</tr>
<tr>
<td>PROJECT (9)</td>
<td>L*1</td>
<td>Project name</td>
</tr>
<tr>
<td>TALL</td>
<td>I*2</td>
<td>Total number of all forms</td>
</tr>
<tr>
<td>TATM</td>
<td>I*2</td>
<td>Total number of ATM forms</td>
</tr>
<tr>
<td>TCRF</td>
<td>I*2</td>
<td>Total number of CRFs</td>
</tr>
<tr>
<td>TCSF</td>
<td>I*2</td>
<td>Total number of CSFs</td>
</tr>
<tr>
<td>TCSR</td>
<td>I*2</td>
<td>Total number of CSRs</td>
</tr>
</tbody>
</table>

3-143

8818
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TGPS</td>
<td>I*2</td>
<td>Total number of GPS forms</td>
</tr>
<tr>
<td>TRAF</td>
<td>I*2</td>
<td>Total number of RAFs</td>
</tr>
<tr>
<td>TRSF</td>
<td>I*2</td>
<td>Total number of RSFs</td>
</tr>
</tbody>
</table>

3.8.4 TASK BUILD PROCEDURE

3.8.4.1 Command Procedures

The NF program can be generated from the source code by executing the command procedure NFGEN.CMD under UIC [204,6]. This command procedure references three command files—NFFPP.CMD, NFFFOR.CMD, and NF.TKB—all under UIC [204,6]. Figure 3-26 is a listing of NFGEN.CMD, the command procedure to precompile, compile, and task build the NF program. The NF program is generated by executing the following command:

@([204,6]NFGEN

3.8.4.2 Overlay Structure

The NF program is overlaid to reduce the memory space requirement. Figure 3-27 is a listing of the Overlay Descriptor Language file, [204,6]NF.ODL, needed to build the NF program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay.
Figure 3-26. NF Task Generation Command Procedure (NFGEN.CMD) (1 of 2)
Figure 3-26. NF Task Generation Command Procedure (NFGEN.CMD) (2 of 2)
OVERLAY DESCRIPTOR LANGUAGE FOR THE FORM COUNTER PROGRAM (NF)

$ROOT: .FCTR $ROOT,OTSALL,RMSALL
$ROOT: .FCTR $R1-$R2-$R5-$R6-RMSROT-OTSROT-$SUBS
$R1: .FCTR [204.6]NFMFORMS-[204.6]NFSTACK-[204.6]NFSUM
$R2: .FCTR [204.6]NFSORT-[204.7]UTGETFLD-[204.7]UTWHERE
$R5: .FCTR [204.7]UTNAME3-[204.7]UTMOVE
$R6: .FCTR [204.7]UTFREAD-[204.7]UTFOPEN
$SUBS: .FCTR *($CRF,$CSF,$CSR,$RAF,$RSF,$SFRPT,$SUMS)
$CRF: .FCTR [204.6]NFCRF-[204.7]UTFCRF
$CSR: .FCTR [204.6]NFCSR-[204.7]UTFCSR
$RAF: .FCTR [204.6]NFRAF-[204.7]UTFRAF
$RSF: .FCTR [204.6]NFRSF-[204.7]UTFRSF-[204.7]UTNEXTWK
$SFRPT: .FCTR [204.6]NFSRPT-($HDR,$SUM,$ENC)
$HDR: .FCTR [204.7]UTFHEADER
$SUM: .FCTR [204.7]UTFSUMRY-($HDR,$EST)
$EST: .FCTR [204.7]UTFEST
$SUMS: .FCTR [204.6]NFSUMS

%LB: [1,1]RMS11M.ODL
%LB: [1,1]RMS12X.ODL
.END

Figure 3-27. NF Program Overlay Descriptor Language File (NF.ODL)
3.9 SEL DATA BASE LISTING PROGRAM (LISTDB)

3.9.1 INTRODUCTION

The SEL Data Base Listing Program (LISTDB) produces formatted and interpreted listings of SEL data base files. File types include Attitude Maintenance (ATM), Component Information File (CIF), Change Report Form (CRF), Component Summary Form (CSF), Component Status Report (CSR), Growth History (HIS), Run Analysis Form (RAF), and Resource Summary Form (RSF).

3.9.2 PROGRAM STRUCTURE

3.9.2.1 Files Accessed

The LISTDB program accesses nine input files and eleven output files as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CIF</td>
<td>CIF for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CRF</td>
<td>CRF file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSF</td>
<td>CSF file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSR</td>
<td>CSR file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.HIS</td>
<td>HIS file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RAF</td>
<td>RAF file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RSF</td>
<td>RSF file for the given project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.ATM</td>
<td>ATM file for the given project</td>
</tr>
</tbody>
</table>

In these file names, <PRJNAM> denotes the name of the project selected by the user.

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISTDB.CIF</td>
<td>Output listing of the CIF</td>
</tr>
<tr>
<td>LISTDB.CRF</td>
<td>Output listing of the CRF file (change report)</td>
</tr>
<tr>
<td>LISTDB.ERR</td>
<td>Output listing of the CRF file (error report)</td>
</tr>
<tr>
<td>LISTDB.CSF1</td>
<td>Output listing of the CSF file (part one)</td>
</tr>
</tbody>
</table>

3-148

8818
<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISTDB.CF2</td>
<td>Output listing of the CSF file (part two)</td>
</tr>
<tr>
<td>LISTDB.CF3</td>
<td>Output listing of the CSF file (part three)</td>
</tr>
<tr>
<td>LISTDB.CSR</td>
<td>Output listing of the CSR file</td>
</tr>
<tr>
<td>LISTDB.HIS</td>
<td>Output listing of the HIS file</td>
</tr>
<tr>
<td>LISTDB.RAF</td>
<td>Output listing of the RAF file</td>
</tr>
<tr>
<td>LISTDB.RSF</td>
<td>Output listing of the RSF file</td>
</tr>
<tr>
<td>LISTDB.ATM</td>
<td>Output listing of the ATM file</td>
</tr>
</tbody>
</table>

3.9.2.2 Baseline Diagram

Figure 3-28 is the baseline diagram for the LISTDB program. The LISTDB routine is the main driver. It obtains the project names and file types and then processes the selected database files and displays them.

3.9.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines forming the LISTDB program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major LISTDB routines are described in Section 3.9.3.5. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the LISTDB program also uses the following system routines: ERRSET and SECONDS.

3.9.3.1 Process Data and Produce Formatted Lists of Files

These 21 major routines process data and produce a formatted list of an SEL data base file.
Figure 3-28. Baseline Diagram for the SEL Data Base Listing Program (LISTDB) (3 of 5)
Figure 3-28. Baseline Diagram for the SEL Data Base Listing Program (LISTDB) (4 of 5)
Figure 3-28. Baseline Diagram for the SEL Data Base Listing Program (LSTDB) (5 of 5)
ROUTINE: ATMDEC

FUNCTION: Decodes and verifies the fields of an ATM file record

CALLING SEQUENCE:

CALL ATMDEC (ATMREC, ENCREC, ENCKEY, PRTLIN, COMPS,
LABELS, LUCIF, LUENC)

ROUTINE: CF1DEC

FUNCTION: Decodes part one of a CSF file record

CALLING SEQUENCE:

CALL CF1DEC (CSFREC, PRTLIN, ENCREC, ENCKEY, LABELS,
COMPS, LUCIF, LUENC)

ROUTINE: CF2DEC

FUNCTION: Decodes part two of a CSF file record

CALLING SEQUENCE:

CALL CF2DEC (CSFREC, PRTLIN, ENCREC, ENCKEY, LABELS,
LUENC)

ROUTINE: CF3DEC

FUNCTION: Decodes part three of a CSF file record

CALLING SEQUENCE:

CALL CF3DEC (CSFREC, PRTLIN, ENCREC, ENCKEY, LABELS,
LUENC)

ROUTINE: CIFDEC

FUNCTION: Decodes and validates the fields of a CIF record

CALLING SEQUENCE:

CALL CIFDEC (CIFREC, ENCREC, ENCKEY, LABELS, LUENC)

3-155

8818
ROUTINE: CRFDEC
FUNCTION: Decodes the primary fields of a CRF file record
CALLING SEQUENCE:

    CALL CRFDEC (CRFREC, PRTLIN, ENCREC, ENCKEY, LABELS, PRNAME, LUCIF, LUENC)

ROUTINE: CSRDEC
FUNCTION: Decodes the fields of a CSR file record
CALLING SEQUENCE:

    CALL CSRDEC (CSRREC, PRTLIN, ENCREC, ENCKEY, LFOR, LABELS, LUCIF, LUENC)

ROUTINE: ERRDEC
FUNCTION: Decodes the fields of the CRF error report
CALLING SEQUENCE:

    CALL ERRDEC (CRFREC, PRTLIN, ENCREC, ENCKEY, LABELS, LUENC)

ROUTINE: LISTDB
FUNCTION: Main driver of the LISTDB program, produces formatted lists of SEL data base files
CALLING SEQUENCE: None

ROUTINE: LSTATM
FUNCTION: Reads, decodes, and displays records from the ATM file
CALLING SEQUENCE:

    CALL LSTATM (PRNAME, ATMREC, ENCREC, ENCKEY, LUCIF, LUATM, LUENC, LUDSP)
ROUTINE: LSTCIF
FUNCTION: Reads, decodes, and displays CIF records
CALLING SEQUENCE:

CALL LSTCIF (PRNAME, CIFREC, ENCREC, ENCKEY, LUCIF, LUENC, LUDSP)

ROUTINE: LSTCRF
FUNCTION: Reads, decodes, and displays CRF file records and 
also displays an error report if indicated
CALLING SEQUENCE:

CALL LSTCRF (PRNAME, CRFREC, ENCREC, ENCKEY, LUCIF, LUCRF, LUENC, LUDSP, LUERR)

ROUTINE: LSTCSF
FUNCTION: Reads, decodes, and displays CSF file records in 
three parts
CALLING SEQUENCE:

CALL LSTCSF (PRNAME, CSFREC, ENCREC, ENCKEY, LUCIF, LUCSF, LUENC, LUDS1, LUDS2, LUDS3)

ROUTINE: LSTCSR
FUNCTION: Reads, decodes, and displays CSR file records
CALLING SEQUENCE:

CALL LSTCSR (PRNAME, CSRREC, ENCREC, ENCKEY, LUCIF, LUCSR, LUENC, LUDSP)

ROUTINE: LSTFIL
FUNCTION: Constructs file names and reads and prints file 
contents
CALLING SEQUENCE:

CALL LSTFIL (NAMTAB, FILIND, PROTAB, NPRO)

ROUTINE: LSTHIS
FUNCTION: Reads, decodes, and displays HIS file records
CALLING SEQUENCE:

CALL LSTHIS (PRNAME, HISREC, LUHIS, LUDSP)

ROUTINE: LSTRAF
FUNCTION: Reads, decodes, and displays RAF file records
CALLING SEQUENCE:

CALL LSTRAF (PRNAME, RAFREC, ENCREC, ENCKEY, LUCIF, LURAF, LUENC, LUDSP)

ROUTINE: LSTRSF
FUNCTION: Reads, decodes, and validates RSF file data
CALLING SEQUENCE:

CALL LSTRSF (PRNAME, RSFREC, ENCREC, ENCKEY, LURSF, LUENC, LUDSP)

ROUTINE: RAFDEC
FUNCTION: Decodes and verifies an RAF file record
CALLING SEQUENCE:

CALL RAFDEC (RAFREC, PRTLIN, ENCREC, ENCKEY, LFOR, LABELS, COMPS, LUCIF, LUENC)

ROUTINE: RSFDEC
FUNCTION: Decodes and displays an RSF file record
CALLING SEQUENCE:

CALL RSFDEC (RSFREC, ENCREC, ENCKEY, LFOR, PRNAME,
             LUDSP, LUENC)

ROUTINE: SELFIL

FUNCTION: Opens a data base file and calls the correspond-
ing read/display routine

CALLING SEQUENCE:

CALL SELFIL (PRNAME, FLNAME, IT, LUDBS, LUENC,
              LUDSP, LUALT, LUOPT, LUCIF)

3.9.3.2 Decode or Verify Data

These 16 routines mainly decode or verify a data field.

ROUTINE: DATTAB

FUNCTION: Computes 10 dates at 7-day intervals subsequent
to the start date

CALLING SEQUENCE:

CALL DATTAB (START, DATES)

ROUTINE: DAYSQR

FUNCTION: Decodes time-to-implement field for the ATM file
record

CALLING SEQUENCE:

CALL DAYSQR (INBYT, OUTFLD)

ROUTINE: FP4TAB

FUNCTION: Decodes numeric fields for the CSR file record

CALLING SEQUENCE:

CALL FP4TAB (INFLD, OUTFLD, NFL)
ROUTINE: FULENC

FUNCTION: Converts numeric codes to alphabetic equivalents using the Encoding Dictionary

CALLING SEQUENCE:

    CALL FULENC (LDATA, LTYPE, ENCREC, ENCKEY, LABELS, NVAL, LUENC)

ROUTINE: INTEMH

FUNCTION: Decodes the complexity field for the CSF file record

CALLING SEQUENCE:

    CALL INTEMH (INBYT, OUTFLD)

ROUTINE: INTMCO

FUNCTION: Interprets resource type

CALLING SEQUENCE:

    CALL INTMCO (INBYT, LSTBYT, OUTFLD, NUM)

ROUTINE: INTNUC

FUNCTION: Decodes the form stage field for the CSF file record

CALLING SEQUENCE:

    CALL INTNUC (INBYT, OUTFLD)

ROUTINE: INTRDM

FUNCTION: Interprets phase flag

CALLING SEQUENCE:

    CALL INTRDM (INFLD, OUTFLD)
ROUTINE: INTYNO
FUNCTION: Interprets yes-no responses
CALLING SEQUENCE:
    CALL INTYNO (INBYT, OUTFLD)

ROUTINE: REDCMP
FUNCTION: Converts numeric codes to alphabetic equivalents using the CIF
CALLING SEQUENCE:
    CALL REDCMP (LDATA, CIFREC, CIFKEY, LABELS, NVAL, LUCIF)

ROUTINE: REDENC
FUNCTION: Converts numeric codes to alphabetic equivalents using the Encoding Dictionary
CALLING SEQUENCE:
    CALL REDENC (LDATA, LTYPE, ENCREC, ENCKEY, LABELS, NVAL, LUENC)

ROUTINE: RSFTAB
FUNCTION: Verifies resource fields
CALLING SEQUENCE:
    CALL RSFTAB (RSFREC, HRSLIN, RUNLIN, RTYPE)

ROUTINE: VALTAB
FUNCTION: Decodes change types and error activities for the ATM file record
CALLING SEQUENCE:
    CALL VALTAB (INFLD, OUTFLD, NAMTAB, NFL, TBYT)

3-161
8818
ROUTINE: VERDAT
FUNCTION: Verifies date
CALLING SEQUENCE:
    CALL VERDAT (INFLD, OUTFLD)

ROUTINE: VERFP4
FUNCTION: Verifies numeric field
CALLING SEQUENCE:
    CALL VERFP4 (INFLD, OUTFLD)

ROUTINE: VERNUM
FUNCTION: Decodes a numeric field
CALLING SEQUENCE:
    CALL VERNUM (INFLD, OUTFLD, FLEN)

3.9.3.3 Obtain Data From Terminal
These two routines obtain information from a user's response to a terminal prompt.

ROUTINE: GETFIL
FUNCTION: Prompts for, validates, and marks file names
CALLING SEQUENCE:
    CALL GETFIL (NAMTAB, NFIL, FILIND)

ROUTINE: GETNAM
FUNCTION: Prompts for project names, checks them against the Encoding Dictionary, and saves them in a table
CALLING SEQUENCE:
    CALL GETNAM (PROTAB, NPRO)
3.9.3.4 Routine With String Movement

These 2 routines deal with string movement.

ROUTINE: MOVEBL

FUNCTION: Moves blanks to an array of specified length

CALLING SEQUENCE:

CALL MOVEBL (VALUE, LENGTH)

ROUTINE: MOVECR

FUNCTION: Moves a given number of bytes from one address to another

CALLING SEQUENCE:

CALL MOVECR (INBUFF, OUTBUF, LENGTH)

3.9.3.5 Variable Description

The variables in the calling sequences of main LISTDB routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATMREC(77)</td>
<td>L*1</td>
<td>Buffer array to hold an ATM file record</td>
</tr>
<tr>
<td>CIFKEY(3)</td>
<td>L*1</td>
<td>Tertiary key for the CIF (component code)</td>
</tr>
<tr>
<td>CIFREC(80)</td>
<td>L*1</td>
<td>Buffer array to hold a CIF record</td>
</tr>
<tr>
<td>COMPS(11)</td>
<td>R*8</td>
<td>Array containing component names</td>
</tr>
<tr>
<td>CRFREC(101)</td>
<td>L*1</td>
<td>Buffer array to hold a CRF file record</td>
</tr>
<tr>
<td>CSFREC(250)</td>
<td>L*1</td>
<td>Buffer array to hold a CSF file record</td>
</tr>
<tr>
<td>CSRREC(79)</td>
<td>L*1</td>
<td>Buffer array to hold a CSR file record</td>
</tr>
<tr>
<td>DATES(22)</td>
<td>I*2</td>
<td>Dates (M1, D1, M2, D2, ..., M11, D11)</td>
</tr>
<tr>
<td>ENCKEY(8)</td>
<td>L*1</td>
<td>Primary key for the Encoding Dictionary (code type and code)</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ENCREC(60)</td>
<td>L*1</td>
<td>Buffer array to hold an Encoding Dictionary record</td>
</tr>
<tr>
<td>FILIND(8)</td>
<td>L*1</td>
<td>Flag indicating whether a given file is to be listed or not</td>
</tr>
<tr>
<td>FLEN</td>
<td>I*2</td>
<td>Length of a given numeric field</td>
</tr>
<tr>
<td>FLNAME(23)</td>
<td>L*1</td>
<td>File name</td>
</tr>
<tr>
<td>HISREC(29)</td>
<td>L*1</td>
<td>Buffer array to hold an HIS file record</td>
</tr>
<tr>
<td>HRSLIN(58)</td>
<td>L*1</td>
<td>Array containing number of hours used for runs</td>
</tr>
<tr>
<td>INBYT</td>
<td>L*1</td>
<td>Input character</td>
</tr>
<tr>
<td>INFLD(X)</td>
<td>L*1</td>
<td>Input characters (length X is variable, dependent on the length of a particular field)</td>
</tr>
<tr>
<td>IT</td>
<td>I*2</td>
<td>File identification number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 1, CIF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 2, CRF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 3, CSF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 4, CSR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 5, RAF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 6, RSF</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 7, HIS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>= 8, ATM</td>
</tr>
<tr>
<td>LABELS(X)</td>
<td>R*8</td>
<td>Decoded value for a field (length X is variable)</td>
</tr>
<tr>
<td>LDATA(X)</td>
<td>L*1</td>
<td>Input numeric codes that are to be converted to alphabetic names using the Encoding Dictionary or CIF (length X is variable)</td>
</tr>
<tr>
<td>LFOR(6)</td>
<td>L*1</td>
<td>Decoded form number</td>
</tr>
<tr>
<td>LSTBYT</td>
<td>L*1</td>
<td>Previous resource type</td>
</tr>
<tr>
<td>LTYPE(X)</td>
<td>L*1</td>
<td>Code type on Encoding Dictionary (length X is variable, X must be multiple of 3)</td>
</tr>
<tr>
<td>LUALT</td>
<td>I*2</td>
<td>Unit number for the second output listing file</td>
</tr>
<tr>
<td>LUATM</td>
<td>I*2</td>
<td>ATM file unit number</td>
</tr>
<tr>
<td>LUCIF</td>
<td>I*2</td>
<td>CIF unit number</td>
</tr>
<tr>
<td>LUCRF</td>
<td>I*2</td>
<td>CRF file unit number</td>
</tr>
<tr>
<td>LUCSF</td>
<td>I*2</td>
<td>CSF file unit number</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>LUCSR</td>
<td>I*2</td>
<td>CSR file unit number</td>
</tr>
<tr>
<td>LUDBS</td>
<td>I*2</td>
<td>Unit number for a given data base file</td>
</tr>
<tr>
<td>LUDSP</td>
<td>I*2</td>
<td>Output report file unit number</td>
</tr>
<tr>
<td>LUDS1</td>
<td>I*2</td>
<td>CSF file output report part one unit number</td>
</tr>
<tr>
<td>LUDS2</td>
<td>I*2</td>
<td>CSF file output report part two unit number</td>
</tr>
<tr>
<td>LUDS3</td>
<td>I*2</td>
<td>CSF file output report part three unit number</td>
</tr>
<tr>
<td>LUENC</td>
<td>I*2</td>
<td>Unit number for Encoding Dictionary</td>
</tr>
<tr>
<td>LUERR</td>
<td>I*2</td>
<td>Unit number for the error report of the CRF file</td>
</tr>
<tr>
<td>LUHIS</td>
<td>I*2</td>
<td>Unit number for the HIS file</td>
</tr>
<tr>
<td>LUOPT</td>
<td>I*2</td>
<td>Unit number for the third output listing file</td>
</tr>
<tr>
<td>LURAF</td>
<td>I*2</td>
<td>RAF file unit number</td>
</tr>
<tr>
<td>LURSF</td>
<td>I*2</td>
<td>RSF file unit number</td>
</tr>
<tr>
<td>NAMTAB(8)</td>
<td>R*4</td>
<td>File name table</td>
</tr>
<tr>
<td>NFIL</td>
<td>I*2</td>
<td>Number of files to be listed</td>
</tr>
<tr>
<td>NFL</td>
<td>I*2</td>
<td>Number of fields</td>
</tr>
<tr>
<td>NPRO</td>
<td>I*2</td>
<td>Number of projects</td>
</tr>
<tr>
<td>NUM</td>
<td>I*2</td>
<td>Code type indicator for RSF record</td>
</tr>
<tr>
<td>NVAL</td>
<td>I*2</td>
<td>Number of bytes of a given field to be decoded</td>
</tr>
<tr>
<td>OUTFLD(X)</td>
<td>L*1</td>
<td>Decoded output characters (length X is variable)</td>
</tr>
<tr>
<td>PRNAME</td>
<td>R*8</td>
<td>Project name</td>
</tr>
<tr>
<td>PROTAB(20)</td>
<td>R*8</td>
<td>Project name array</td>
</tr>
<tr>
<td>PRTLIN(X)</td>
<td>L*1</td>
<td>Decoded output characters (length X is variable)</td>
</tr>
<tr>
<td>RAFREC(53)</td>
<td>L*1</td>
<td>Buffer array to hold an RAF file record</td>
</tr>
<tr>
<td>RSFREC(115)</td>
<td>L*1</td>
<td>Buffer array to hold an RSF file record</td>
</tr>
<tr>
<td>RTYPE</td>
<td>L*1</td>
<td>Resource type</td>
</tr>
</tbody>
</table>

3-165

8818
### Name Type Description

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUNLIN(33)</td>
<td>L*1</td>
<td>Decoded number of runs for computer resource</td>
</tr>
<tr>
<td>START(6)</td>
<td>L*1</td>
<td>First date</td>
</tr>
<tr>
<td>TBYT(2)</td>
<td>L*1</td>
<td>Error detection activities identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>D = detection, I = isolation, B = both</td>
</tr>
</tbody>
</table>

#### 3.9.4 TASK BUILD PROCEDURE

##### 3.9.4.1 Command Procedures

The LISTDB program can be generated from the source code by executing the command procedure [204,6]DLGEN.CMD. This command procedure references three command files--DLFPP.CMD, DLFOR.CMD, and LISTDB.TKB--all under UIC [204,6]. Figure 3-29 is a listing of DLGEN.CMD, the command procedure to precompile, compile, and task build the LISTDB program. The LISTDB program is generated by executing the following command:

```
@[204,6]DLGEN
```

##### 3.9.4.2 Overlay Structure

The LISTDB program is overlaid to reduce the memory space requirement. Figure 3-30 is a listing of the Overlay Descriptor Language file, [204,6]LISTDB.ODL, needed to build the LISTDB program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay.
Figure 3-29. LISTDB Task Generation Command Procedure (DLGEN.CMD) (1 of 3)
Figure 3-29. LISTDB Task Generation Command Procedure (DLGEN.CMD) (2 of 3)
:FOR/F4P/OBJECT:[204,6]DLVERNUM [204,6]DLVERNUM
: ROUTINE WITH PREFIX DM
: ;FOR/F4P/OBJECT:[204,15]DMMOVEBL [204,15]DMMOVEBL
: ; COMPARE ASSEMBLER ROUTINE
: MAC/OBJECT:[204,7]UTCHAREQ [204,7]UTCHAREQ
: ; TASK BUILD THE LISTDB PROGRAM
: TKB *[204,6]LISTDB.TKB
: ; LISTDB.TKB
: ; COMMAND PROCEDURE TO TASK BUILD THE SEL DATA BASE LISTING PROGRAM
: (LISTDB)
: ; [204,5]LISTDB=[204,6]LISTDB/MP
: ; UNITS=8
: ; ACTFIL=6
: ; MAXBUF=250
: //

Figure 3-29. LISTDB Task Generation Command Procedure (DLGEN.CMD) (3 of 3)
Figure 3-30. LISTDB Program Overlay Descriptor Language File (LISTDB.ODL)
3.10 SEL DATA BASE RECENT ACTIVITY REPORT PROGRAM (RC)

3.10.1 INTRODUCTION

The SEL Data Base Recent Activity Report Program (RC) generates a one-page report of the additions, deletions and changes to records in the SEL data base since the last back-up date. This information is retrieved from the transaction files.

3.10.2 PROGRAM STRUCTURE

3.10.2.1 Files Accessed

The RC program accesses eight input files and one output file as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]ENCODE.HDR</td>
<td>Encoding Dictionary (ENC) file</td>
</tr>
<tr>
<td>DB0:[204,1]TRANS.CIF</td>
<td>Component Information Transaction file</td>
</tr>
<tr>
<td>DB0:[204,1]TRANS.CrF</td>
<td>Change Report Form Transaction file</td>
</tr>
<tr>
<td>DB0:[204,1]TRANS.CSF</td>
<td>Component Summary Form Transaction file</td>
</tr>
<tr>
<td>DB0:[204,1]TRANS.CSR</td>
<td>Component Status Report Transaction file</td>
</tr>
<tr>
<td>DB0:[204,1]TRANS.HIS</td>
<td>Growth History Transaction file</td>
</tr>
<tr>
<td>DB0:[204,1]TRANS.RAF</td>
<td>Run Analysis Form Transaction file</td>
</tr>
<tr>
<td>DB0:[204,1]TRANS.RSF</td>
<td>Resource Summary Form Transaction file</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECENT.RPT</td>
<td>Recent activity output report file</td>
</tr>
</tbody>
</table>

3-171
3.10.2.2 Baseline Diagram

Figure 3-31 is the baseline diagram for the RC program. The RECENT routine is the main driver. It obtains the project name from the Encoding Dictionary; counts all adds, deletes, and changes from the transaction files for a given project; and then generates a report of all adds, deletes, and changes for all projects.

3.10.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines forming the RC program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any), and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major RC routines are described in Section 3.10.3.6. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the RC program also uses the following system routines: DATE, ERRSET, SECNDS, and TIME.

3.10.3.1 Process Data and Compute Statistics

These two major routines count all adds, deletes, and changes in the transaction files for all projects.

ROUTINE: RECENT

FUNCTION: Main routine of the RC program, generates a one-page report of the additions, deletions, and changes to records in the SEL data base

CALLING SEQUENCE: None
Figure 3-31. Baseline Diagram for the SEL Data Base Recent Activity Report Program (RC)
ROUTINE: SUMTYP

FUNCTION: Obtains a count of all additions, deletions, and changes to the given file type in the data base from the transaction files

CALLING SEQUENCE:

CALL SUMTYP (ITYP, LOC, MAXACT, MAXPRJ, MAXTYP, RECL, TYP, COUNTS, DATE)

3.10.3.2 Write Output Report

These two routines write a one-page report of the additions, deletions, and changes for all projects.

ROUTINE: HEADER

FUNCTION: Prints a one-line title for each report page that includes the date and project

CALLING SEQUENCE:

CALL HEADER (IRPTF, PRJNAM, RPTITL)

ROUTINE: RPT

FUNCTION: Prints a one-page report of the transaction file counts

CALLING SEQUENCE:

CALL RPT (COUNTS, DATE, MAXACT, MAXPRJ, MAXTYP, PRJNAM, SRTKEY)

3.10.3.3 File Open and Read Routines

These five routines either open an indexed file or read records from an indexed file.
ROUTINE: FENCA

FUNCTION: Finds the description field on the Encoding Dictionary corresponding to the given type and code

CALLING SEQUENCE:

CALL FENCA (IENCF, TYPE, CODE, NAME, REST, FOUND)

ROUTINE: FOPEN

FUNCTION: Opens an indexed file

CALLING SEQUENCE:

CALL FOPEN (IUNIT, FILNAM, ERROR)

ROUTINE: FREAD

FUNCTION: Reads one indexed record

CALLING SEQUENCE:

CALL FREAD (IUNIT, KEYVAL, KEYLEN, LRECL, BUFFER, ERROR)

ROUTINE: GETPRJ

FUNCTION: Obtains all project names from the Encoding Dictionary

CALLING SEQUENCE:

CALL GETPRJ (MAXPRJ, PRJNAM)

ROUTINE: RDSEQ

FUNCTION: Reads one record from a sequential file
CALLING SEQUENCE:

CALL RDSEQ (IUNIT, NCHARS, CHARS, EOF)

3.10.3.4 Sort Routine
This one routine provides a sort function.

ROUTINE: SORT8
FUNCTION: Generates an array of indices to alphabetize the given name array

CALLING SEQUENCE:

CALL SORT8 (MAX, NSORT, NAMES, SRTKEY)

3.10.3.5 Routines for String Movement or Comparison
These five routines deal with string movement or comparison.

ROUTINE: BLANK
FUNCTION: Initializes an array to blanks

CALLING SEQUENCE:

CALL BLANK (ARRAY, NUM)

ROUTINE: CHARGT (LOGICAL FUNCTION)
FUNCTION: Determines whether the first string alphabetically follows the second

CALLING SEQUENCE:

CHARGT (STRNG1, STRNG2, LEN)

ROUTINE: INTG (INTEGER*2 FUNCTION)
FUNCTION: Converts the given characters to integer

CALLING SEQUENCE:

INTG (BUFFER, LEN)
ROUTINE: MOVE
FUNCTION: Moves a given number of bytes from one address to another
CALLING SEQUENCE:
   CALL MOVE (A, B, LEN)

ROUTINE: NAME3
FUNCTION: Concatenates the given strings to form a complete file name
CALLING SEQUENCE:
   CALL NAME3 (DISK, UIC, NAME, EXTENS, DSN)

3.10.3.6 Variable Description
The variables in the calling sequences of main RC routines are described below.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COUNTS</td>
<td>I*2</td>
<td>Count of all additions, deletions, and changes of all database files as recorded on the transaction files</td>
</tr>
<tr>
<td>(MAXACT, MAXTYP, MAXPRJ)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATE(6)</td>
<td>L*1</td>
<td>Last backup date</td>
</tr>
<tr>
<td>ITYP</td>
<td>I*2</td>
<td>Number of current file type</td>
</tr>
<tr>
<td>LOC</td>
<td>I*2</td>
<td>Location of the field of the project code within a record</td>
</tr>
<tr>
<td>MAX</td>
<td>I*2</td>
<td>Maximum number of project names</td>
</tr>
<tr>
<td>MAXACT</td>
<td>I*2</td>
<td>Total number of activities (add, delete, change, total)</td>
</tr>
<tr>
<td>MAXPRJ</td>
<td>I*2</td>
<td>Maximum number of projects</td>
</tr>
<tr>
<td>MAXTYP</td>
<td>I*2</td>
<td>Total number of files + 1</td>
</tr>
<tr>
<td>NAMES(8,NSORT)</td>
<td>L*1</td>
<td>Names to be sorted</td>
</tr>
<tr>
<td>NSORT</td>
<td>I*2</td>
<td>Number of names to be sorted</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PRJNAM(8,MAXPRJ)</td>
<td>L*1</td>
<td>Project names</td>
</tr>
<tr>
<td>RECL</td>
<td>I*2</td>
<td>Logical record length for a given transaction file</td>
</tr>
<tr>
<td>SRTKEY(MAXPRJ)</td>
<td>I*2</td>
<td>Sort index array to alphabetize project names</td>
</tr>
<tr>
<td>TYP(3)</td>
<td>L*1</td>
<td>Current file type (e.g., 'CIF')</td>
</tr>
</tbody>
</table>

3.10.4 TASK BUILD PROCEDURE

3.10.4.1 Command Procedure

The RC program can be generated from the source code by executing the command procedure RCGEN.CMD under UIC [204,6]. This command procedure references three command procedures—RCFPP.CMD, RCFOR.CMD, and RC.TKB—all under UIC [204,6]. Figure 3-32 is a listing of RCGEN.CMD, the command procedure to precompile, compile, and task build the RC program. The RC program is generated by executing the following command:

```bash
@[204,6]RCGEN
```

3.10.4.2 Overlay Structure

The RC program is overlaid to reduce the memory space requirement. Figure 3-33 is a listing of the Overlay Descriptor Language file, [204,6]RC.ODL, needed to build the RC program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay.
# RCGEN.CMD

COMMAND PROCEDURE TO GENERATE THE RECENT ACTIVITY REPORT PROGRAM (RC) FROM THE SOURCE CODES (P. LO 7/30/82)

PRECOMPILE ALL STRUCTURED FORTRAN SOURCE CODES

# [204.6]RCFPP

# RCFPP.CMD

COMMAND PROCEDURE TO PRECOMPILE ALL ROUTINES WRITTEN IN STRUCTURED FORTRAN FOR THE SEL DATA BASE RECENT ACTIVITY REPORT PROGRAM (RC) (P. LO 7/30/82)

ROUTINES WITH PREFIX RC

FPP SY:[204.6]RCGETPRJ
FPP SY:[204.6]RCRPT
FPP SY:[204.6]RCSORT8
FPP SY:[204.6]RCSUMTYP

ROUTINES WITH PREFIX UT

FPP SY:[204.7]UTBLANK
FPP SY:[204.7]UTCHARGT
FPP SY:[204.7]UTFENCAC
FPP SY:[204.7]UTFOPEN
FPP SY:[204.7]UTFREAD
FPP SY:[204.7]UTHEADER
FPP SY:[204.7]UTINTG
FPP SY:[204.7]UTMOVE
FPP SY:[204.7]UTNAME3
FPP SY:[204.7]UTRDSEQ

COMPILE ALL FORTRAN ROUTINES

# [204.6]RCFOR

# RCFOR.CMD

COMMAND PROCEDURE TO COMPILE ALL FORTRAN ROUTINES FOR THE SEL DATA BASE RECENT ACTIVITY REPORT PROGRAM (RC) (P. LO 7/30/82)

ROUTINES WITH PREFIX RC

FOR/F4P/OBJECT:[204.6]RCGETPRJ [204.6]RCGETPRJ
FOR/F4P/OBJECT:[204.6]RCRPT [204.6]RCRPT
FOR/F4P/OBJECT:[204.6]RCSORT8 [204.6]RCSORT8
FOR/F4P/OBJECT:[204.6]RCSUMTYP [204.6]RCSUMTYP

Figure 3-32. RC Task Generation Command Procedure (RCGEN.CMD) (1 of 2)
Figure 3-32. RC Task Generation Command Procedure (RCGEN.CMD) (2 of 2)
Figure 3-33. RC Program Overlay Descriptor Language File (RC.ODL)
3.11 SEL DATA BASE RECORD COUNTING REPORT PROGRAM (RPSTSCTR)

3.11.1 INTRODUCTION

The SEL Data Base Record Counting Program (RPSTSCTR) counts the number of records in each file in the SEL data base and produces a one-page report of all counts.

3.11.2 PROGRAM STRUCTURE

3.11.2.1 Files Accessed

The RPSTSCTR program accesses all SEL data base files as input files and produces one output report file. In addition, the user's copy of the File Name and Status (STS) file is accessed as both an input and an output file.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]ENCODE.HDR</td>
<td>Encoding Dictionary (ENC) file</td>
</tr>
<tr>
<td>[204,1]HEADER.HDR</td>
<td>Phase Dates (HDR) file</td>
</tr>
<tr>
<td>[204,1]STAT.HDR</td>
<td>STS file</td>
</tr>
<tr>
<td>[204,1]EST.HDR</td>
<td>Estimated Statistics (EST) file</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CIF</td>
<td>Component Information File (CIF) for each project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CMT</td>
<td>Comment file for each project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CRF</td>
<td>Change Report Form (CRF) file for each project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSF</td>
<td>Component Summary Form (CSF) file for each project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CSR</td>
<td>Component Status Report (CSR) file for each project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.HIS</td>
<td>Growth History (HIS) file for each project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RAF</td>
<td>Run Analysis Form (RAF) file for each project</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.RSF</td>
<td>Resource Summary Form (RSF) file for each project</td>
</tr>
<tr>
<td>[User's UIC]STAT.HDR</td>
<td>User's copy of the STS file</td>
</tr>
</tbody>
</table>

In these file names, <PRJNAM> is the project name.
Output File Name | Description
--- | ---
STSCTR.RPT | Output report file
[User's UIC]STAT.HDR | User's copy of the STS file

3.11.2.2 Baseline Diagram

Figure 3-34 is the baseline diagram for the RPSTSCTR program. The STSCTR routine is the driver that opens all input files, counts the number of records in each file, and then writes the output report. It also updates the user's copy of the STS file.

3.11.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The RPSTSCTR program references only two routines, STSCTR (main routine) and MOVECR, in addition to two system routines (DATE, SECNDS) and the RMSIAC routines. These two routines are described below. However, descriptions of the calling sequence variables for MOVECR are not provided.

ROUTINE: STSCTR

FUNCTION: Main routine of the RPSTSCTR program, counts the number of records in each file in the SEL database and produces a one-page report of all counts

CALLING SEQUENCES: NONE

ROUTINE: MOVECR

FUNCTION: Moves given number of characters from one address to another.

CALLING SEQUENCES:

CALL MOVECR (INBUFF, OTBUFF, LENGTH)
Figure 3-34. Baseline Diagram for the SEL Data Base Record Counting Report Program (RPSTSCTR)
3.11.4 TASK BUILD PROCEDURE

3.11.4.1 Command Procedures

The RPSTSCTR program can be generated from the source code by executing the command procedure RPSTSGEN.CMD under UIC [204,6]. Figure 3-35 is a listing of this command procedure, which precompiles and compiles the FORTRAN routine, compiles the ASSEMBLER routine, and task builds the RPSTSCTR program. RPSTSGEN.CMD references another command procedure, RPSTSCTR.TKB, also under UIC [204,6], which builds the RPSTSCTR program task image. The RPSTSCTR program is generated by entering the following command:

@([204,6]RPSTSGEN

3.11.4.2 Overlay Structure

The RPSTSCTR task is overlaid to reduce the memory space requirement. Figure 3-36 is a listing of the Overlay Descriptor Language file, [204,6]RPSTSCTR.ODL, needed to build the RPSTSCTR program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay. In addition, the RMS Indexed Access Programs Library (RMSIAC), [204,7]UFRMSIAC.OLB, is also needed in the overlay. This library contains FORTRAN routines used to access RMS indexed files.
Figure 3-35. RPSTSCTR Task Generation Command Procedure (RPSTSGEN.CMD)

@RPSTSCTR.CMD

COMMAND PROCEDURE TO GENERATE THE SEL DATA BASE RECORD COUNTING PROGRAM (RPSTSCTR) FROM SOURCE CODES
(P. LQ 8/11/82)

PRECOMPILE STRUCTURED FORTRAN SOURCE CODES

FPP SY:[204.6]RPSTSCTR

COMPILE FORTRAN SOURCE CODES

FOR/F4P/OBJECT:[204.6]RPSTSCTR [204.6]RPSTSCTR

COMPILE ASSEMBLER ROUTINE

MAC/OBJECT:[204.7]UTCHAREQ [204.7]UTCHAREQ

TASK BUILD THE RPSTSCTR PROGRAM

TKB @[204.6]RPSTSCTR.TKB

@RPSTSCTR.TKB

COMMAND PROCEDURE TO BUILD THE SEL DATA BASE RECORD COUNTING PROGRAM (RPSTSCTR)

:[204.5]RPSTSCTR/FU,RPSTSCTR/NOSP/SH=[204.6]RPSTSCTR.ODL/MP

:ACTFIL=2

:UNITS=20

:MAXBUF=250

://

Figure 3-36. RPSTSCTR Program Overlay Descriptor Language File (RPSTSCTR.ODL)

@RPSTSCTR.ODL

THE OVERLAY STRUCTURE FOR THE SEL DATA BASE RECORD COUNTING PROGRAM (RPSTSCTR)
(P. LQ 8/11/82)

.ROOT RMSROT-OTSROT-$ROOT,OTSALL,RMSALL

$ROOT: .FCTR [204.6]RPSTSCTR-[204.7]UTCHAREQ-[204.7]UFRMSIAC/LB

$LB:[1,1]RMS11M.ODL

$LB:[1,1]RMS12X.ODL

.END

3-186
3.12 COMPONENT NAME REPORT GENERATOR PROGRAM (RPCOMPNM)

3.12.1 INTRODUCTION

The Component Name Report Generator Program (RPCOMPNM) reads all Component Information Files (CIFs) on the SEL data base and produces a formatted and alphabetized report of component names and codes for all such files.

3.12.2 PROGRAM STRUCTURE

3.12.2.1 Files Accessed

The RPCOMPNM program accesses all CIFs and the Encoding Dictionary as the input files and one output file.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]ENCODE.HDR</td>
<td>Encoding Dictionary (ENC) file</td>
</tr>
<tr>
<td>[204,1]&lt;PRJNAM&gt;.CIF</td>
<td>CIF for each project, where &lt;PRJNAM&gt; is the project name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPNAMES.RPT</td>
<td>Output report file</td>
</tr>
</tbody>
</table>

3.12.2.2 Baseline Diagram

Figure 3-37 is the baseline diagram for the RPCOMPNM program. The COMRPT routine is the driver that opens all input files, reads the desired data from the files, and writes the output report.

3.12.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

In addition to two system routines (DATE, SECNDS) and the RMSIAC routines, the RPCOMPNM program references only one routine, the driver (COMRPT), as described below.

ROUTINE: COMRPT

FUNCTION: Reads component names and codes from CIFs and writes a formatted report of all components for all projects.

CALLING SEQUENCES: None
3.12.4 TASK BUILD PROCEDURE

3.12.4.1 Command Procedures

The RPCOMPNM program can be generated from the source code by executing the command procedure RPCOMGEN.CMD under UIC [204,6]. This command procedure precompiles and compiles the FORTRAN routines and task builds the RPCOMPNM program. It references another command procedure, RPCOMPNM.TKB, also under UIC [204,6], which builds the RPCOMPNM program task image. Figure 3-38 is a listing of RPCOMGEN.CMD. The RPCOMPNM program is generated by executing the following command:

`@([204,6]RPCOMGEN`

3.12.4.2 Overlay Structure

The RPCOMPNM program is overlaid to reduce the memory space requirement. Figure 3-39 is a listing of the Overlay Descriptor Language file, [204,6]RPCOMPNM.ODL, needed to build the RPCOMPNM program task image. The system libraries RMS11M.ODL and RMS12X.ODL and the RMS Indexed Access Programs Library (RMSIAC) are needed in the overlay. The name of this last library is UFRMSIAC.OLB, under UIC [204,7]; it contains FORTRAN routines used to access RMS indexed files.
Figure 3-38. RPCOMPNM Task Generation Command Procedure (RPCOMGEN.CMD)

Figure 3-39. RPCOMPNM Program Overlay Descriptor Language File (RPCOMPNM.ODL)
3.13 SUBJECTIVE EVALUATIONS FILE LISTING PROGRAM (DBRPTSEF)

3.13.1 INTRODUCTION

The Subjective Evaluations File Listing Program (DBRPTSEF) reads the Subjective Evaluations File (SEF) on the SEL database and generates a formatted report of the contents of the file organized by the category of measure.

3.13.2 PROGRAM STRUCTURE

3.13.2.1 Files Accessed

The DBRPTSEF program accesses two input files and one output file as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1] ENCODE.HDR</td>
<td>Encoding Dictionary File</td>
</tr>
<tr>
<td>[204,1] SEF.HDR</td>
<td>Subjective Evaluations File</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,3] SEFDAT.RPT</td>
<td>Output listing of the contents of the SEF</td>
</tr>
</tbody>
</table>

3.13.2.2 Baseline Diagram

Figure 3-40 is the baseline diagram for the DBRPTSEF program. The SEFRPT routine is the main driver. It opens all files, obtains all project codes from the SEF and the corresponding project names from the Encoding Dictionary, obtains the user option for the category of measure to be listed, and then writes the selected listing from the SEF. It loops through this process until a Z (control Z) is entered by the user in response to a prompt.

3.13.3 SUBROUTINE/SUBSYSTEM DESCRIPTION

The routines forming the DBRPTSEF program are grouped here by function. In each routine, the calling sequence variables are grouped according to input, input and output (if any),

3-191
and output and appear in the calling sequence in that order. In the following descriptions, each group of variables begins a new line. The calling sequence variables for the major DBRPTSEF routines are described in Section 3.13.3.5. Descriptions of the calling sequence variables for utility routines are not provided. In addition to the routines described in this section, the DBRPTSEF program also uses the following system routines: DATE, ERRSET, and SECNDS.

3.13.3.1 Process Data and Produce Formatted Listing

These 23 major routines process the SEF data and produce a formatted listing of the contents of the SEF.

ROUTINE: GTCODE

FUNCTION: Obtains all project codes from the SEF and the corresponding project names from the Encoding Dictionary and sorts them alphabetically

CALLING SEQUENCE:

CALL GTCODE (IENC, ISEF, PRCO, PROJ, IREC, ERROR)

ROUTINE: SEFRPT

FUNCTION: Main routine of the DBRPTSEF program, produces a formatted listing of the contents of the SEF organized by category of measure

CALLING SEQUENCE: None
ROUTINE: WRTAP
FUNCTION: Generates the output listing for the experience with application (AP) measure
CALLING SEQUENCE:
    CALL WRTAP (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE: WRTCO
FUNCTION: Generates the output listing for the COCOMO (CO) model measure
CALLING SEQUENCE:
    CALL WRTCO (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE: WRTCP
FUNCTION: Generates the output listing for the complexity of problem (CP) measure
CALLING SEQUENCE:
    CALL WRTCP (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE: WRTDC
FUNCTION: Generates the output listing for the documentation (DC) measure
CALLING SEQUENCE:
    CALL WRTDC (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE: WRTEX
FUNCTION: Generates the output listing for the external influences on project (EX) measure
CALLING SEQUENCE:
    CALL WRTEX (ISEF, IRPT, PROJ, PRCO, IREC)
ROUTINE:  WRTIN
FUNCTION:  Generates the output listing for the internal influences on project (IN) measure
CALLING SEQUENCE:

   CALL WRTIN (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE:  WRTMG
FUNCTION:  Generates the output listing for the effectiveness of management (MG) measure
CALLING SEQUENCE:

   CALL WRTMG (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE:  WRTMS
FUNCTION:  Generates the output listing for the miscellaneous (MS) measure
CALLING SEQUENCE:

   CALL WRTMS (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE:  WRTMT
FUNCTION:  Generates the output listing for the practices and techniques (MT) measure
CALLING SEQUENCE:

   CALL WRTMT (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE:  WRTPF
FUNCTION:  Generates the output listing for the performance of team (PF) measure
CALLING SEQUENCE:

   CALL WRTPF (ISEF, IRPT, PROJ, PRCO, IREC)
ROUTINE: WRTPP

FUNCTION: Generates the output listing for the product/process performance (PP) measure

CALLING SEQUENCE:

    CALL WRTPP (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE: WRTPR

FUNCTION: Generates the output listing for the software product (PR) measure

CALLING SEQUENCE:

    CALL WRTPR (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE: WRTPS

FUNCTION: Generates the output listing for the PRICE S3 (PS) model measure

CALLING SEQUENCE:

    CALL WRTPS (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE: WRTRA

FUNCTION: Generates the output listing for the resources available (RA) measure

CALLING SEQUENCE:

    CALL WRTRA (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE: WRTRK

FUNCTION: Generates the output listing for the team rank (RK) measure

CALLING SEQUENCE:

    CALL WRTRK (ISEF, IRPT, PROJ, PRCO, IREC)

3-197

8818
ROUTINE:  WRTSW

FUNCTION:  Generates the output listing for the code breakdown (SW) measure

CALLING SEQUENCE:

    CALL WRTSW (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE:  WRTTS

FUNCTION:  Generates the output listing for the tools (TS) measure

CALLING SEQUENCE:

    CALL WRTTS (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE:  WRTWF

FUNCTION:  Generates the output listing for the Walston-Felix (WF) model measure

CALLING SEQUENCE:

    CALL WRTWF (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE:  WRTYA

FUNCTION:  Generates the output listing for the years of applicable experience (YA) measure

CALLING SEQUENCE:

    CALL WRTYA (ISEF, IRPT, PROJ, PRCO, IREC)

ROUTINE:  WRTYE

FUNCTION:  Generates the output listing for the years of environment experience (YE) measure

CALLING SEQUENCE:

    CALL WRTYE (ISEF, IRPT, PROJ, PRCO, IREC)

3-198

8818
ROUTINE: WRTYP
FUNCTION: Generates the output listing for the years of professional experience (YP) measure

CALLING SEQUENCE:

   CALL WRTYP (ISEF, IRPT, PROJ, PRCO, IREC)

3.13.3.2 Input and Output Routines
These five routines perform either input or output functions.

ROUTINE: FENCA
FUNCTION: Finds the description field on the Encoding Dictionary for the given type and code

CALLING SEQUENCE:

   CALL FENCA (IENCF, TYPE, CODE, NAME, REST, FOUND)

ROUTINE: FOPEN
FUNCTION: Opens an indexed file

CALLING SEQUENCE:

   CALL FOPEN (IUNIT, FILNAM, ERROR)

ROUTINE: FREAD
FUNCTION: Reads one indexed record

CALLING SEQUENCE:

   CALL FREAD (IUNIT, KEYVAL, KEYLEN, LRECL, BUFFER, ERROR)

3-199
8818
ROUTINE: HEADR

FUNCTION: Prints a two-line title for each report page, including the date and page number

CALLING SEQUENCE:

CALL HEADR (IRPT, TITLE1, TITLE2, IPAGE)

ROUTINE: RDSEF

FUNCTION: Reads one record from the SEF

CALLING SEQUENCE:

CALL RDSEF (ISEF, KVAL, ERROR, BUF, LRECL)

3.13.3.3 Sort Routine

This routine provides a sort function.

ROUTINE: SORT8

FUNCTION: Generates an array of indices to alphabetize the given name array

CALLING SEQUENCE:

CALL SORT8 (MAX, NSORT, NAMES, SRTKEY)

3.13.3.4 Routines Performing String Movement or Comparison

These three routines deal with string movement or comparison.

ROUTINE: CHARGT (LOGICAL*1 FUNCTION)

FUNCTION: Determines if the first string is alphabetically after the second

CALLING SEQUENCE:

CHARGT (STRNG1, STRNG2, LEN)
ROUTINE: MATCHS (LOGICAL*1 FUNCTION)
FUNCTION: Determines whether two input strings match
CALLING SEQUENCE:

MATCHS (ARRAY1, ARRAY2, NBYTES)

ROUTINE: MOVE
FUNCTION: Moves a given number of bytes from one address to another
CALLING SEQUENCE:

CALL MOVE (A, B, LEN)

3.13.3.5 Variable Description
The variables in the calling sequences of major DBRPTSEF routines are described below

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUF(578)</td>
<td>L*1</td>
<td>The SEF record buffer</td>
</tr>
<tr>
<td>ERROR</td>
<td>L*1</td>
<td>Error flag</td>
</tr>
<tr>
<td>IENC</td>
<td>I*2</td>
<td>FORTRAN unit number for the Encoding Dictionary</td>
</tr>
<tr>
<td>IPAGE</td>
<td>I*2</td>
<td>Page number</td>
</tr>
<tr>
<td>IREC</td>
<td>I*2</td>
<td>Number of projects</td>
</tr>
<tr>
<td>IRPT</td>
<td>I*2</td>
<td>FORTRAN unit number for the output report file</td>
</tr>
<tr>
<td>ISEF</td>
<td>I*2</td>
<td>FORTRAN unit number for the SEF</td>
</tr>
<tr>
<td>KVAL(3)</td>
<td>L*1</td>
<td>Key value</td>
</tr>
<tr>
<td>PRCO(70)</td>
<td>I*2</td>
<td>Array of project codes</td>
</tr>
<tr>
<td>PROJ(70)</td>
<td>R*8</td>
<td>Array of project names</td>
</tr>
<tr>
<td>TITLE1(40)</td>
<td>L*1</td>
<td>First title line for each report page</td>
</tr>
<tr>
<td>TITLE2(50)</td>
<td>L*1</td>
<td>Second title line for each report page</td>
</tr>
</tbody>
</table>
3.13.4 TASK BUILD PROCEDURE

3.13.4.1 Command Procedures

The DBRPTSEF program can be generated from the source code by executing the command procedure DBSEFGEN.CMD under UIC [204,6]. This command procedure references three command procedures--DBSEFFPP.CMD, DBSEFFOR.CMD, and DBRPTSEF.TKB--all under UIC [204,6]. Figure 3-41 is a listing of DBSEFGEN.CMD, the command procedure to precompile, compile, and task build the DBRPTSEF program. The DBRPTSEF task is generated by executing the following command:

@*[204,6]DBSEFGEN

3.13.4.2 Overlay Structure

The DBRPTSEF program is overlaid to reduce the memory space requirement. Figure 3-42 is a listing of the Overlay Descriptor Language file, [204,6]DBRPTSEF.ODL, needed to build the DBRPTSEF program task image. The system libraries RMS11M.ODL and RMS12X.ODL are needed for the overlay.
Figure 3-41. DBRPTSEF Task Generation Command Procedure (DBSEFGEN.CMD) (1 of 2)
COMPILE FORTRAN ROUTINES

*OBSEFFOR

COMMAND PROCEDURE TO COMPILE ALL FORTRAN ROUTINES FOR THE SUBJECTIVE EVALUATIONS FILE LISTING PROGRAM (DBRPTSEF) (P. LO 9/9/82)

ROUTINE WITH PREFIX SF

FOR/F4P/OBJECT: [204.6]SFGTCODE [204.6]SFHEADR
FOR/F4P/OBJECT: [204.6]SFHEADR
FOR/F4P/OBJECT: [204.6]SFROSEF
FOR/F4P/OBJECT: [204.6]SFSEFRPT
FOR/F4P/OBJECT: [204.6]SFWRDC
FOR/F4P/OBJECT: [204.6]SFWRTEX
FOR/F4P/OBJECT: [204.6]SFWRTPF

ROUTINE WITH PREFIX DM, RC, OR UT

FOR/F4P/OBJECT: [204.6]RCSORT8
FOR/F4P/OBJECT: [204.7]UTCHARGT
FOR/F4P/OBJECT: [204.7]UTrFNCA
FOR/F4P/OBJECT: [204.7]UTFOPEN
FOR/F4P/OBJECT: [204.7]UTFREAD
FOR/F4P/OBJECT: [204.7]UTMATCHS
FOR/F4P/OBJECT: [204.7]UTMOVE
FOR/F4P/OBJECT: [204.18]DMZFILL

TASK BUILD THE DBRPTSEF PROGRAM

TKB *[204.6]DBRPTSEF.TKB

COMMAND PROCEDURE TO BUILD THE SUBJECTIVE EVALUATIONS FILE LISTING PROGRAM (DBRPTSEF) TASK IMAGE (P. LO 9/9/82)

DBRPTSEF*[204.6]DBRPTSEF.DDL/MP

MAXBUF=578

Figure 3-41. DBRPTSEF Task Generation Command Procedure (DBSEFGEN.CMD) (2 of 2)
THE OVERLAY STRUCTURE FOR THE SUBJECTIVE EVALUATIONS FILE LISTING
PROGRAM (DBRPTSEF)
(P. LO 9/9/82)

.ROOT RMSROT-OTSROT-ROOT.OTSALL.RMSALL

$ROOT: .FCTR [204.6]SFSEFRPT-[204.7]UTFOPEN-[204.7]UTMATCHS-R1
R1: .FCTR [204.6]SHEADR-[204.6]SFREDSEF-[204.15]DMZFILL-R2
R2: .FCTR *(CODE,WRT)

CODE: .FCTR [204.6]SFGT_CODE-*READ.SORT*

READ: .FCTR [204.7]UTFENC A-[204.7]UTFMOVE-[204.7]UTFREAD 1
SORT: .FCTR [204.6]RC_SORT8-[204.7]UTFCHARGT


A: .FCTR [204.6]SFWRTAP
B: .FCTR [204.6]SFWRTC0
C: .FCTR [204.6]SFWRTCP
D: .FCTR [204.6]SFWRTDC
E: .FCTR [204.6]SFWRTEX
F: .FCTR [204.6]SFWRTIN
G: .FCTR [204.6]SFWRTMG
H: .FCTR [204.6]SFWRTMS
I: .FCTR [204.6]SFWRTMT
J: .FCTR [204.6]SFWRTPF
K: .FCTR [204.6]SFWRTPF
L: .FCTR [204.6]SFWRTPP
M: .FCTR [204.6]SFWRTPR
N: .FCTR [204.6]SFWRTPS
O: .FCTR [204.6]SFWRTA
P: .FCTR [204.6]SFWRTSK
Q: .FCTR [204.6]SFWRTSW
R: .FCTR [204.6]SFWRTTS
S: .FCTR [204.6]SFWRTWF
T: .FCTR [204.6]SFWRTYA
U: .FCTR [204.6]SFWRTYE

$LB:[1,1]RMS11M.ODL
$LB:[1,1]RMS12X.ODL
.END

Figure 3-42. DBRPTSEF Program Overlay Descriptor Language File (DBRPTSEF.ODL)
3.14 SUBJECTIVE EVALUATIONS DIRECTORY FILE LISTING PROCEDURE (DBRPTDIR)

3.14.1 INTRODUCTION

The Subjective Evaluations Directory File Listing Procedure (DBRPTDIR) lists the contents of the Subjective Evaluations Directory (DIR) file by using DATATRIEVE (Reference 4).

3.14.2 FILES ACCESSED

The DBRPTDIR procedure accesses one input file and one output file as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]DIR.HDR</td>
<td>Subjective Evaluations Directory File</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEFDIR.RPT</td>
<td>Output listing file</td>
</tr>
</tbody>
</table>

3.14.3 DATATRIEVE COMMAND FILE

Figure 3-43 is a listing of DBRPTDIR.DTR under UIC [204,4], a DATATRIEVE command file that generates a listing of the contents of the DIR file.
SET DICTIONARY [204, 1] QUERY.DIC:
READY SEFDIR:
FIND E IN SEFDIR SORTED BY CODE:
REPORT ALL CURRENT ON SEFDIR.RPT
SET REPORT-NAME="SUBJECTIVE EVALUATIONS DIRECTORY INFORMATION (DIR.HDR)"
PRINT CODE ("CODE").
NAME ("MEASURE"/"NAME"),
MIN-VALUE ("MIN"/"VALUE"),
MAX-VALUE ("MAX"/"VALUE"),
DATA-REC-NO ("REC"/"SEQ"),
BYTE-LOC ("BYTE"/"LOC"),
DESCRIPTION ("DESCRIPTION")
REPORT END

YOUR REPORT IS ON FILES 'SEFDIR.RPT'
PLEASE PRINT THIS FILE.

Figure 3-43. DBRPTDIR DATATRIEVE Command File (DBRPTDIR.DTR)
3.15 ENCODING DICTIONARY LISTING PROCEDURE (DBRPTENC)

3.15.1 INTRODUCTION

The Encoding Dictionary Listing Procedure (DBRPTENC) produces a listing of the contents of the Encoding Dictionary File by using DATATRIEVE (Reference 4).

3.15.2 FILES ACCESSED

The DBRPTENC procedure accesses one input file and one output file as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]ENCODE.HDR</td>
<td>Encoding Dictionary (ENC) file</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENC.RPT</td>
<td>Output listing file of the Encoding Dictionary</td>
</tr>
</tbody>
</table>

3.15.3 DATATRIEVE COMMAND FILE

Figure 3-44 is a listing of DBRPTENC.DTR under UIC [204,4], the DATATRIEVE command file that generates a listing of the contents of the ENC File.
SET DICTIONARY [204,1]QUERY.DIC;
READ ENC;
FIND E IN ENC;
REPORT CURRENT SORTED BY TYPE, CODE ON ENC.RPT
SET REPORT-NAME='ENCODING DICTIONARY (ENCODE_HDR)',LINES-PAGE=60,
   COLUMNS-PAGE=90
AT TOP OF TYPE PRINT SKIP;
PRINT COL 10(TYPE USING ZZ9, COL 18CODE USING XXXXX,
   COL 24, NAME, COL 40, REST
REPORT END;
!
! YOUR REPORT IS ON FILE 'ENC.RPT'

Figure 3-44. DBRPTENC DATATRIEVE Command File
(DBRPTENC.DTR)
3.16 PHASE DATES FILE LISTING PROCEDURE (DBRPTHDR)

3.16.1 INTRODUCTION

The Phase Dates File Listing Procedure (DBRPTHDR) produces a listing of the contents of the Phase Dates (HDR) file by using DATATRIEVE (Reference 4).

3.16.2 FILES ACCESSED

The DBRPTHDR procedure accesses one input file and one output file as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]HEADER.HDR</td>
<td>HDR file</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR.RPT</td>
<td>Output listing file of the HDR file</td>
</tr>
</tbody>
</table>

3.16.3 DATATRIEVE COMMAND FILE

Figure 3-45 is a listing of [204,4]DBRPTHDR.DTR, the DATATRIEVE command file that generates a listing of the contents of the HDR file.
SET DICTIONARY [204,1]QUERY.DIC:
READY HDR;
FIND S IN HDR WITH DES1 > 0
REPORT ALL CURRENT SORTED BY NAME ON HDR.RPT
SET REPORT-NAME="HEADER DATA (FILE [204,1]HEADER.HDR)", LINES-PAGE=65,
COLUMN-PAGE=78
PRINT NAME ("PROJECT"),
    DES1 ("DESIGN" /"START"/"DATE") USING ZZZZZZ,
    CODE1 ("CODE &"/"UNIT TEST" /"START"/"DATE") USING ZZZZZZ,
    SYS1 ("SYSTEM"/"TEST" /"START"/"DATE") USING ZZZZZZ,
    ACC1 ("ACCEPTANCE"/"TEST" /"START"/"DATE") USING ZZZZZZ,
    CLEAN1 ("CLEANUP" /"START"/"DATE") USING ZZZZZZ,
    CLEAN2 ("CLEANUP" /"END" /"DATE") USING ZZZZZZ
REPORT END
| YOUR REPORT IS ON FILE 'HDR.RPT'

Figure 3-45. DBRPHDR DATATRIEVE Command File (DBRPHDR.DTR)
3.17 FILE NAME AND STATUS FILE LISTING PROCEDURE (DBRPTSTS)

3.17.1 INTRODUCTION

The File Name and Status File Listing Procedure (DBRPTSTS) produces a listing of the contents of the File Name and Status (STS) file by using DATATRIEVE (Reference 4).

3.17.2 FILES accessed

The DBRPTSTS procedure accesses one input file and one output file as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]STAT.HDR</td>
<td>STS file</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAT.RPT</td>
<td>Output listing file of the STS file</td>
</tr>
</tbody>
</table>

3.17.3 DATATRIEVE COMMAND FILE

Figure 3-46 is a listing of [204,4]DBRPTSTS.DTR, the DATATRIEVE command file that generates a listing of the contents of the STS file.
SET DICTIONARY [204,1]QUERY.DIC
READY STAT
FIND S IN STAT
REPORT CURRENT SORTED BY PROJ ON STAT.RPT
  SET REPORT-NAME="DIRECTORY FILE - STAT.DAT",
  COLUMNS-PAGE=90
  PRINT FILE USING ZZ, NAME, CREATE, BACKUP, UPDATE,
  NREC USING ZZZZZ
  AT TOP OF PROJ PRINT SKIP,"PROJECT =",PROJ USING ZZ
REPORT END
!
! YOUR REPORT IS ON FILE STAT.RPT

Figure 3-46. DBRPTSTS DATATRIEVE Command File
  (DBRPTSTS.DTR)
3.18  ESTIMATED STATISTICS FILE LISTING PROCEDURE (DBRPTEST)

3.18.1  INTRODUCTION

The Estimated Statistics File Listing Procedure (DBRPTEST) produces a listing of the contents of the Estimated Statistics (EST) file by using DATATRIEVE (Reference 4).

3.18.2  FILES ACCESSED

The DBRPTEST procedure accesses one input file and two output files as described below.

<table>
<thead>
<tr>
<th>Input File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[204,1]EST.HDR</td>
<td>EST file</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EST1.RPT</td>
<td>Part one of the output report of the contents of the EST file</td>
</tr>
<tr>
<td>EST2.RPT</td>
<td>Part two of the output report of the contents of the EST file</td>
</tr>
</tbody>
</table>

3.18.3  DATATRIEVE COMMAND FILE

Figure 3-47 is a listing of DBRPTEST.DTR under UIC [204,4], the DATATRIEVE command file that generates the listings of the contents of the EST file.
Figure 3-47. DBRPTTEST DATATRIEVE Command File (DBRPTTEST.DTR)
REFERENCES


BIBLIOGRAPHY OF SEL LITERATURE

The technical papers, memorandums, and documents listed in this bibliography are organized into two groups. The first group is composed of documents issued by the Software Engineering Laboratory (SEL) during its research and development activities. The second group includes materials that were published elsewhere but pertain to SEL activities.

SEL-Originated Documents

SEL-76-001, *Proceedings From the First Summer Software Engineering Workshop*, August 1976


SEL-77-003, *Structured FORTRAN Preprocessor (SFORT)*, B. Chu and D. S. Wilson, September 1977

SEL-77-004, *GSFC NAVPAK Design Specifications Languages Study*, P. A. Scheffer and C. E. Velez, October 1977


†This document superseded by revised document.
SEL-78-004, Structured FORTRAN Preprocessor (SFORT)

SEL-78-005, Proceedings From the Third Summer Software Engineering Workshop, September 1978


SEL-78-007, Applicability of the Rayleigh Curve to the SEL Environment, T. E. Mapp, December 1978


SEL-79-005, Proceedings From the Fourth Summer Software Engineering Workshop, November 1979


†This document superseded by revised document.

S-2

8818

SEL-80-005, A Study of the Musa Reliability Model, A. M. Miller, November 1980

SEL-80-006, Proceedings From the Fifth Annual Software Engineering Workshop, November 1980


†SEL-81-001, Guide to Data Collection, V. E. Church, D. N. Card, F. E. McGarry, et al., September 1981

SEL-81-101, Guide to Data Collection, V. E. Church, D. N. Card, F. E. McGarry, et al., August 1982


†SEL-81-102, Software Engineering Laboratory (SEL) Data Base Organization and User's Guide Revision 1, P. Lo and D. Wyckoff, March 1983 (superseded by July 1983 version of SEL-81-102)


†This document superseded by revised document.

S-3

†SEL-81-007, Software Engineering Laboratory (SEL) Compendium of Tools, W. J. Decker, E. J. Smith, A. L. Green, et al., February 1981


SEL-81-012, The Rayleigh Curve As a Model for Effort Distribution Over the Life of Medium Scale Software Systems, G. O. Picasso, December 1981

SEL-81-013, Proceedings From the Sixth Annual Software Engineering Workshop, December 1981

SEL-81-014, Automated Collection of Software Engineering Data in the Software Engineering Laboratory (SEL), A. L. Green, W. J. Decker, and F. E. McGarry, September 1981


SEL-82-002, FORTRAN Static Source Code Analyzer Program (SAP) System Description, W. A. Taylor and W. J. Decker, August 1982

SEL-82-003, Software Engineering Laboratory (SEL) Data Base Reporting Software User's Guide and System Description, P. Lo, September 1982 (superseded by August 1983 version of SEL-82-003)

†This document superseded by revised document.

S-4
SEL-82-004, Collected Software Engineering Papers: Volume 1, July 1982

SEL-82-005, Glossary of Software Engineering Laboratory Terms, M. G. Rohleder, December 1982

SEL-82-006, Annotated Bibliography of Software Engineering Laboratory (SEL) Literature, D. N. Card, November 1982

SEL-82-007, Proceedings From the Seventh Annual Software Engineering Workshop, December 1982

SEL-82-008, Evaluating Software Development by Analysis of Changes: The Data From the Software Engineering Laboratory, V. R. Basili and D. M. Weiss, December 1982

SEL-Related Literature


Basili, V. R., "SEL Relationships for Programming Measurement and Estimation," University of Maryland, Technical Memorandum, October 1979


†† This article also appears in SEL-82-004, Collected Software Engineering Papers: Volume 1, July 1982.


†† This article also appears in SEL-82-004, Collected Software Engineering Papers: Volume 1, July 1982.


Freburger, K., "A Model of the Software Life Cycle" (paper prepared for the University of Maryland, December 1978)

Higher Order Software, Inc., TR-9, A Demonstration of AXES for NAVPAK, M. Hamilton and S. Zeldin, September 1977 (also designated SEL-77-005)

Hislop, G., "Some Tests of Halstead Measures" (paper prepared for the University of Maryland, December 1978)

Lange, S. F., "A Child's Garden of Complexity Measures" (paper prepared for the University of Maryland, December 1978)

Miller, A. M., "A Survey of Several Reliability Models" (paper prepared for the University of Maryland, December 1978)

National Aeronautics and Space Administration (NASA), NASA Software Research Technology Workshop (proceedings), March 1980


‡‡This article also appears in SEL-82-004, Collected Software Engineering Papers: Volume 1, July 1982.

S-7

Reiter, R. W., "The Nature, Organization, Measurement, and Management of Software Complexity" (paper prepared for the University of Maryland, December 1976)


††This article also appears in SEL-82-004, Collected Software Engineering Papers: Volume 1, July 1982.