Capstone Project

Modern Functional Analyst Tooling Support for the Modernisation of Legacy Mainframe Systems

CHEN YU

COMPUTER SCIENCE AY 2021/2022

ACCENTURE SINGAPORE SERVICES

SINGAPORE INSTITUTE OF TECHNOLOGY – UNIVERSITY OF GLASGOW

Industrial Background

Software Modernisation

Mainframe (COBOL, COPYBOOK, CISCS etc) to Java and Angular

Coexistence and interoperability

Motivation

Being a business and functional analyst analysing legacy codebase is a cumbersome role.

Analysist could perform their tasks more effectively with more specialised tools and guidelines.

Problem Statements

Source code interpretation are repetitive, analyst tasks lack automation.

Codebase analysis does not integrate well with existing tool chains.

Existing analysis tools being used offered little visualisation and not intuitive to interact with.

Caption Project Solution

To make a comprehensive analysis tool to simplify and keep track of **functional and requirement** data mining processes from legacy system artefacts.

The tool works as **extension** to a code editor currently in use and provides an integrated analysis environment, comparing to switching over several documentations then scrolling through thousands of lines of code.

The tool will improve the **utilities** in source code annotations, visualization and syntax highlighting. It is to be specifically **fine-tuned** to the legacy COBOL codebase we are working on.

Research Areas

Software Modernisation Process and Techniques

- M. Jha and P. Maheshwari, “Reusing Code for Modernization of Legacy Systems”, 2005
- A. J. McAllister and S. A. O’Hara, “Toward Effective Management of Large-Scale Software”, 2016
- J. B. Bowles, “Code from requirements: new productivity tools improve the reliability and maintainability of software systems”, 2004

Research Areas

Developer Tools and Development Experience

- D. Jin, "Design Issues for Software Analysis and Maintenance Tools," 13th IEEE International Workshop on Software Technology and Engineering Practice (STEP'05), 2005
- T. Besker, A. Martini and J. Bosch, "Technical Debt Cripples Software Developer Productivity: A Longitudinal Study on Developers' Daily Software Development Work", 2018
- E. Murphy-Hill et al., "What Predicts Software Developers' Productivity?", 2021

Research Areas

Design of Developer Tools and Developer Productivity

- R. Jeffries, “Designing interfaces for programmers” 1997
- Premraj, Rahul, et al. “An empirical analysis of software productivity over time”, 2005.
- M. Kern et al., “Integrating Static Code Analysis Toolchains”, 2019
- D. T. Lee, “Low-Level Developer Tools and Productivity”, 2021

Research Areas

Implementation of Developer Tools

- VS Code Extension Documentation and Samples - https://github.com/microsoft/vscode-extension-samples
- Data Driven Documents - https://d3js.org/
- Vue JS and Vuetify - https://vuejs.org/ https://vuetifyjs.com
- TensorFlow - https://www.tensorflow.org/

Target User Audience

Functional (Business) Analyst

- Little programming experience required
- Bridge between clients and developers
- Requirements Gathering
- System Design Documentations
- Data mining
- Test planning and quality assurance
- Simple Development tasks

Tool Design Goals

- Exploratory analysis of the codebase
- Familiarisation of the codebase
- Recommendation on which part is important

Autonomy of the Codebase

Single Copybook Folder

- A folder that contains full of shared copybook files
- Copybooks are like data models to specify what to pass around external calls

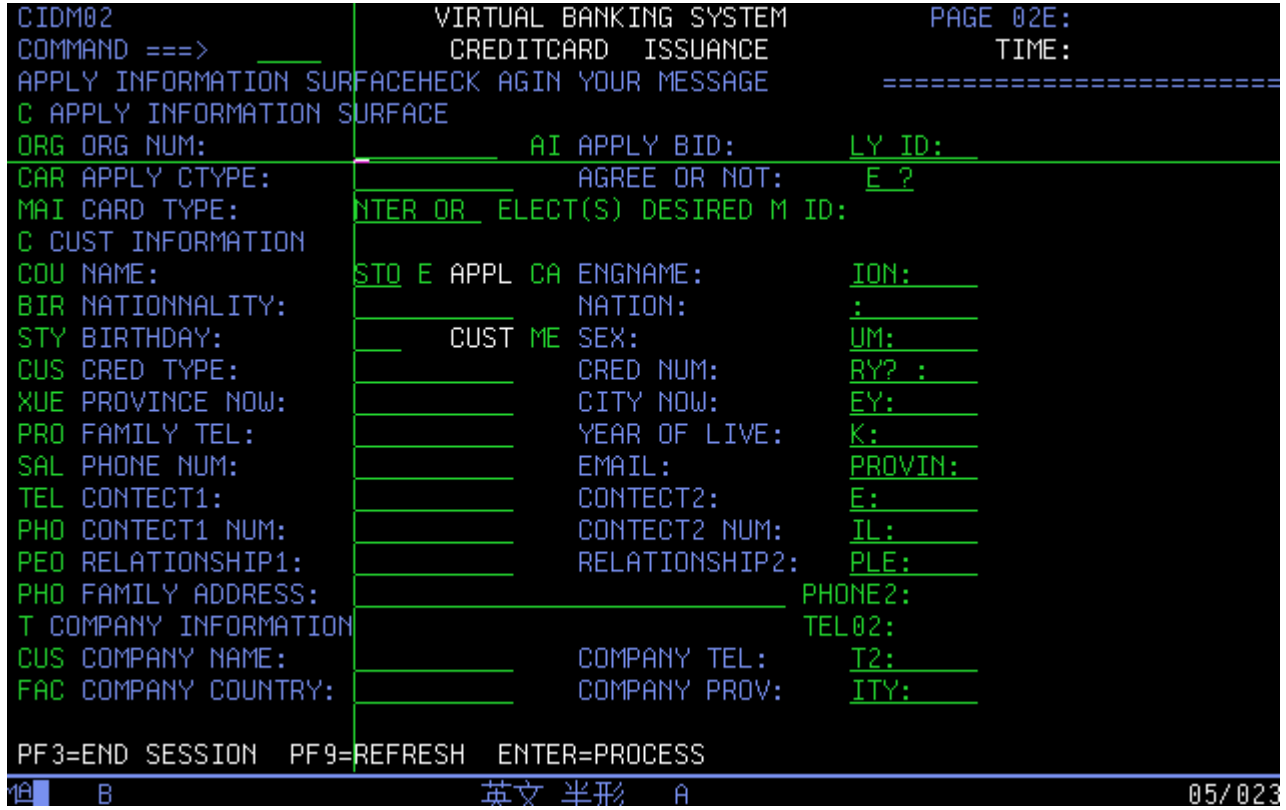
System Folders

- CICS (**C**ustomer **I**nformation **C**ontrol **S**ystem MAP
- COBOL (**CO**mmon **B**usiness **O**riented **L**anguage)

- CICS are markup files for the mainframe user interface of program
- Codes can be are structured into systems, or departments with specific business operation

Other files can consists of EZT and Batch are not covered by this tool

A COBOL Program with CICS Interface



Code sample used for the demonstration of this project is found from:

https://github.com/RookieDay/VirtualBank

Autonomy COBOL Source

Declarative Divisions

```
IDENTIFICATION DIVISION.
PROGRAM-ID. INFS001.
AUTHOR. CY.

ENVIRONMENT DIVISION.
DATA DIVISION.
*=======================*
WORKING-STORAGE SECTION.
* Framework Working Storage
COPY FWTW001.

* Routine Copybooks
COPY INFR001IO.

01 WS-VARIABLE.
   05 WS-ID                          PIC X(9).
   05 WS-DOB.
      10 WS-DOB-CCYY                  PIC 9(4).
      10 WS-DOB-MM                    PIC 9(2).
      10 WS-DOB-DD                    PIC 9(2).
   05 WS-NAME.
      10 WS-NAME-FIRST                PIC X(20).
      10 WS-NAME-LAST                 PIC X(20).
   05 WS-ADDRESS-PRIMARY.
      10 WS-ADDRESS-PRIMARY-LINE1     PIC X(100).
      10 WS-ADDRESS-PRIMARY-LINE2     PIC X(100).
      10 WS-ADDRESS-PRIMARY-PSTCDE    PIC 9(6).
* ROUTINE NAMES
77 INFR101            PIC X(08) VALUE 'INFR101'.
```

Autonomy COBOL Source

Sections of Program Logic Implementation

```
167     1030-POPULATE-APPL.
168          INITIALIZE CIC0007O-REC
169          MOVE SD-SRV-OUTPUT-DATA            TO CIC0007O-REC
170          MOVE CIC0007O-REVIEW-ID            TO REVIDO
171          MOVE CIC0007O-REVIEW-DATE          TO REVDTO
172          MOVE CIC0007O-REVIEW-RESULT        TO REVRTO
173          MOVE CIC0007O-REVIEW-REFUSE-REASON TO REVRRO
174          MOVE CIC0007O-REVIEW-COMMENT       TO COMMTO
175          .
176     *
177     1030-POPULATE-APPL-EXIT.
178          EXIT.
179     *
180     1040-ATTR-PROTECT.
181          MOVE ATTR-PROT-SKIP-MDT       TO REVIDA
182          MOVE ATTR-PROT-SKIP-MDT       TO REVDTA
183          MOVE ATTR-PROT-SKIP-MDT       TO REVRTA
184          MOVE ATTR-PROT-SKIP-MDT       TO REVRRA
185          MOVE ATTR-PROT-SKIP-MDT       TO COMMTA
186          .
187     *
188     1040-ATTR-PROTECT-EXIT.
189          EXIT.
190     *
191     2000-PRE-PROCESSING.
192     *
193     2000-PRE-PROCESSING-EXIT.
194          EXIT.
195     *
196     3000-MAIN-PROCESS.*>IMPORTANT
197          EVALUATE EIBAID ALSO TRUE
198              WHEN DFHPF1 ALSO ANY
199                   EXEC CICS
200                        XCTL PROGRAM('CIOCCIMN')
201                             RESP(WS-RESP-CODE)
202                   END-EXEC
203                   IF WS-RESP-CODE NOT = DFHRESP(NORMAL)
204                      MOVE 'PROGRAM CIOCCIMN IS NOT AVAILABLE'
205                               TO MSGO
206                      SET WS-MAP-DATAONLY TO TRUE
207                      PERFORM 3030-SEND-MAP THRU 3030-SEND-MAP-EXIT
208                   END-IF
209              WHEN DFHPF3 ALSO ANY
210                   MOVE 'THANK YOU FOR USING THE SYSTEM' TO WS-MESSA
211                   EXEC CICS
212                        SEND CONTROL
213                        CURSOR
214                        ERASE
```

Autonomy COBOL Source

External Program Calls

```
*                                                                       01290001
 3000-MAIN-PROCESS.*>IMPORTANT
     EVALUATE EIBAID ALSO TRUE                                          01310020
         WHEN DFHPF1 ALSO ANY                                           01320020
             EXEC CICS                                                  01330001
                 XCTL PROGRAM('CIOCCIMN')                               01340001
                      RESP(WS-RESP-CODE)                                01350001
             END-EXEC                                                   01360001
             IF WS-RESP-CODE NOT = DFHRESP(NORMAL)                      01370001
                MOVE 'PROGRAM CIOCCIMN IS NOT AVAILABLE'                01380001
                       TO MSGO                                          01390001
                SET WS-MAP-DATAONLY TO TRUE                             01400001
                PERFORM 3030-SEND-MAP THRU 3030-SEND-MAP-EXIT           01410001
             END-IF                                                     01420001
         WHEN DFHPF3 ALSO ANY                                           01430020
             MOVE 'THANK YOU FOR USING THE SYSTEM' TO WS-MESSAGE01440001
             EXEC CICS                                                  01450001
                 SEND CONTROL                                           01460001
                 CURSOR                                                 01470001
                 ERASE                                                  01480001
                 FREEKB                                                 01490001
                 ALARM                                                  01500001
             END-EXEC                                                   01510001
```

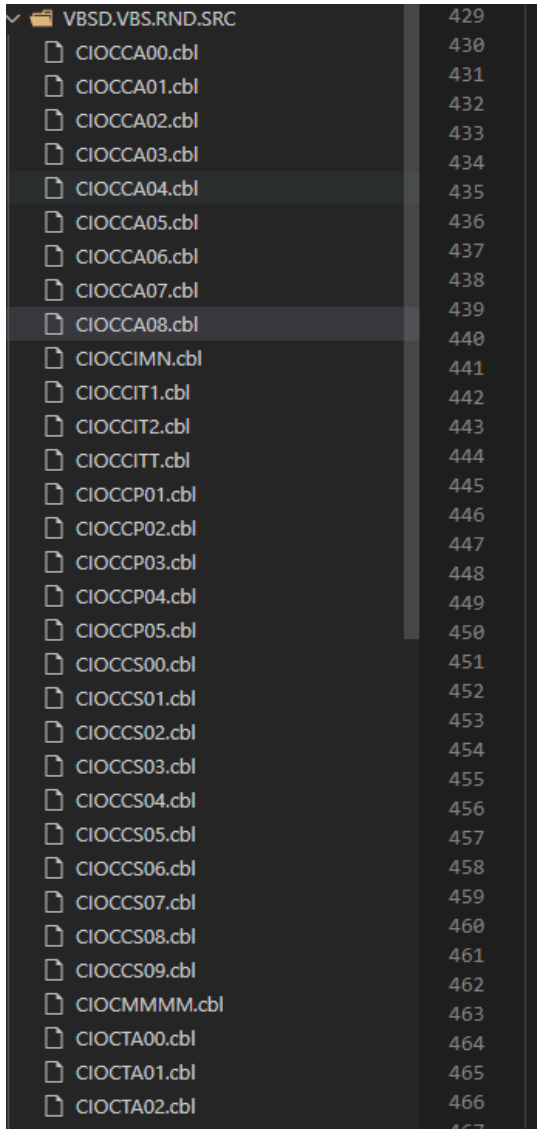
Common Challenges Faced

- Large Codebase

How to quickly determine the relationships and how are they connected?

- Overwhelming number of block of codes

Where to start?
What parts are obsolete?
What are the business logics?



Focused COBOL Analysis

Initialisation filler code?

Obsolete code?

Complex code does not mean it contains business logic.

```
1010-ASK-TIME-DATE.
*
      EXEC CICS
           ASKTIME
           ABSTIME(WS-GETTIME)
      END-EXEC
      EXEC CICS
           FORMATTIME
           ABSTIME(WS-GETTIME)
           DATESEP('/')
           YYYYMMDD(WS-DATEOUT)
      END-EXEC
      EXEC CICS
           FORMATTIME
```

```
*                                                                       01720002
1030-POPULATE-MAP.                                                      01730002
     PERFORM VARYING IX FROM 1 BY 1 UNTIL IX > 14                       01740002
       IF IX <= CIC0009O-COUNT                                          01750002
          INITIALIZE CIC0012I-REC                                       01760002
          MOVE CIC0009O-APPL-ID(IX) TO CIC0012I-APPL-ID                 01770002
          PERFORM 1040-GET-CUSTAPPL                                     01780002
             THRU 1040-GET-CUSTAPPL-EXIT                                01790002
          MOVE CIC0012O-ID             TO APPIDO(IX)                    01800002
          MOVE ATTR-PROT-SKIP-MDT      TO APPIDA(IX)                    01801011
          MOVE CIC0012O-NAME           TO NAMEO(IX)                     01810002
          MOVE CIC0012O-CUST-ID-NUMBER TO IDNUMO(IX)                    01820002
          STRING CIC0012O-LAST-DATE ' ' CIC0012O-LAST-TIME              01830002
                 DELIMITED BY SIZE   INTO LASTO(IX)                     01840002
          EVALUATE CIC0012O-STATUS                                      01850002
```

```
1000-INIT.
*     IF EIBCALEN = 0
*        MOVE 'Y' TO LK-FIRST-SEND
*        EVALUATE EIBTRNID
*            WHEN 'CI02'
*                 MOVE 1 TO LK-OPTION
*            WHEN 'CI03'
*                 MOVE 2 TO LK-OPTION
*            WHEN 'CI04'
*                 MOVE 3 TO LK-OPTION
*            WHEN 'CI05'
*                 MOVE 4 TO LK-OPTION
*        END-EVALUATE
*     END-IF
      IF LK-FIRST-SEND = 'Y'
         INITIALIZE CIC0009I-REC
         MOVE LK-OPTION TO CIC0009I-OPTION
         MOVE ZEROS TO CIC0009I-APPL-ID
         MOVE 'F8' TO CIC0009I-DIRECTION
*
```

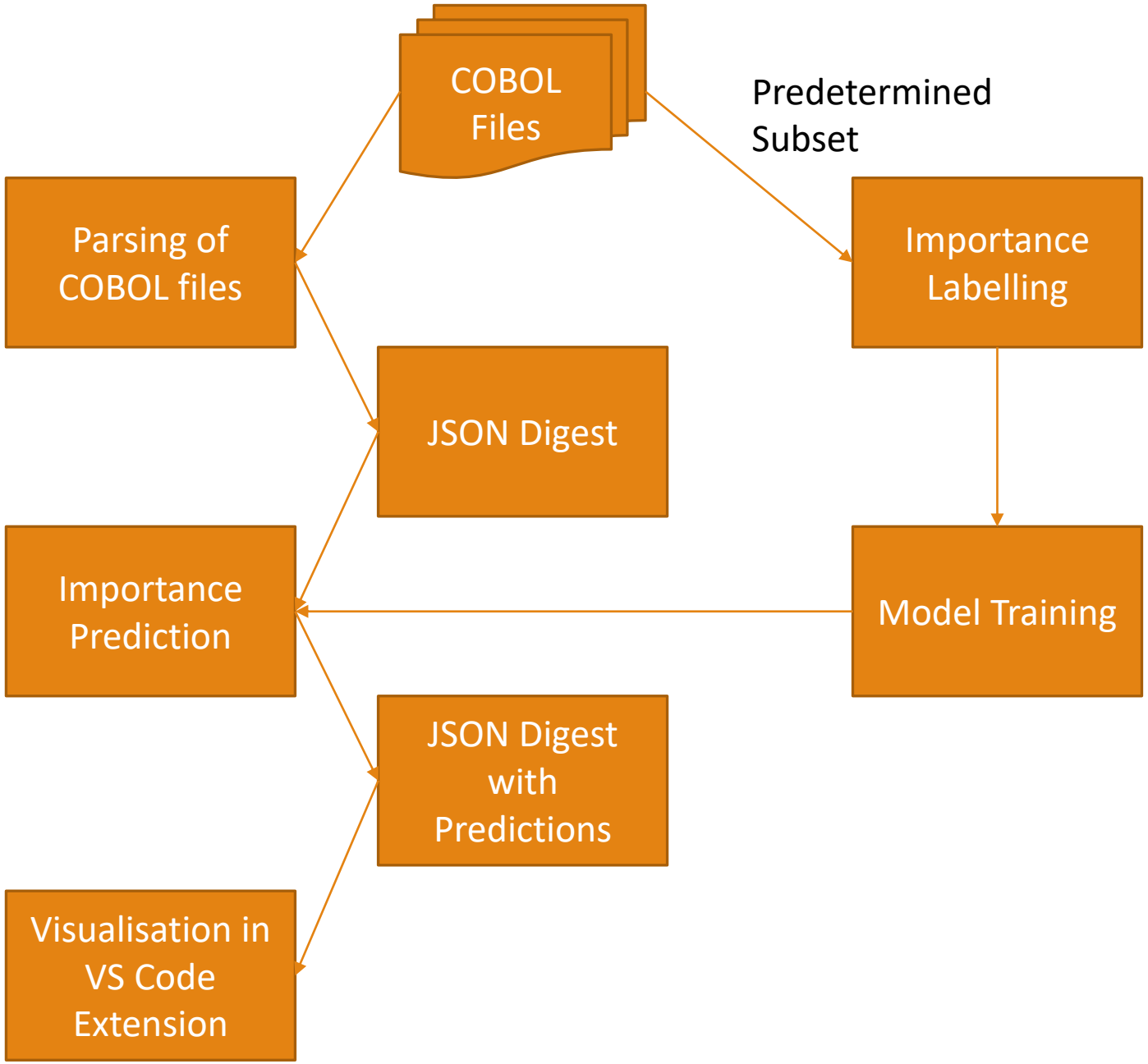
Focused COBOL Analysis

Important Sections that contains business logics which will be migrated?

```
3020-ADD-CUSTAPPL.*>IMPORTANT
    INITIALIZE SDCA-SERVICE-COMMAREA
    MOVE 'VBS.CI.CUSTAPPL.ADD' TO SD-SRV-NAME
    INITIALIZE CIC0011I-REC
    PERFORM 3040-POPULATE-CICUS
       THRU 3040-POPULATE-CICUS-EXIT
    MOVE CIC0011I-REC  TO SD-SRV-INPUT-DATA
    EXEC CICS
         LINK
         PROGRAM(WS-PGM-SRV-DRIVER)
         COMMAREA(WS-SRV-COMMAREA)
         RESP(WS-RESP-CODE)
    END-EXEC
    EVALUATE WS-RESP-CODE
        WHEN DFHRESP(NORMAL)
             IF SD-RESP-CODE EQUAL ZEROS
```

```
1020-GET-APPL-LIST.*>IMPORTANT
    INITIALIZE SDCA-SERVICE-COMMAREA
    MOVE 'VBS.CI.APPLICAN.IN2' TO SD-SRV-NAME
    MOVE CIC0009I-REC TO SD-SRV-INPUT-DATA
    EXEC CICS
         LINK
         PROGRAM(WS-PGM-SRV-DRIVER)
         COMMAREA(WS-SRV-COMMAREA)
         RESP(WS-RESP-CODE)
    END-EXEC
    EVALUATE WS-RESP-CODE
        WHEN DFHRESP(NORMAL)
*            IF SD-RESP-CODE EQUAL ZEROS
```

Implementation Architecture



JSON Digest Structure

```
63      {
64        "file": "sample/VBSD.VBS.RND.SRC/CIOCCA00.cbl",
65        "importance": true,
66        "calls": [
67          {
68            "type": "XCTL PROGRAM",
69            "target": "CIOCCIMN"
70          }
71        ],
72        "raw": "3000-MAIN-PROCESS.\nEVALUATE EIBAID\nWHEN DFHPF1\nEXEC CICS\nXCTL
73        "confidence": 0.8435518145561218
74      },
75      {
76        "file": "sample/VBSD.VBS.RND.SRC/CIOCCA00.cbl",
77        "importance": false,
78        "calls": [],
79        "raw": "3000-MAIN-PROCESS-EXIT.\nEXIT.",
80        "confidence": 0.999467087443918
81      },
82      {
83        "file": "sample/VBSD.VBS.RND.SRC/CIOCCA00.cbl",
84        "importance": true,
85        "calls": [],
86        "raw": "3010-CHECK-INPUT.\nINITIALIZE CIMENU-REC\nEVALUATE TRUE\nWHEN (CO
87        "confidence": 0.8419065475463867
88      },
89      {
```

```
export type CobolFile = {
    file: string
    presence: "source" | "specs" | "missing"
    name: string
    sections: CobolSection[]
}
```

```
export type CobolSection = {
    file: string,
    importance: boolean
    confidence: number
    calls: CobolSectionCall[]
    raw: string
}
```

```
export type CobolSectionCall = {
    type: string
    target: string
    copybook?: string
}
```

Implementation of the RNN Model

- A subset of COBOL files to be labelled for the importance of sections

```
3020-ADD-CUSTAPPL.*>IMPORTANT
    INITIALIZE SDCA-SERVICE-COMMAREA
```

```
1020-GET-APPL-LIST.*>IMPORTANT
    INITIALIZE SDCA-SERVICE-COMMAREA
```

- For this sample codebase is 687 out of 1159 sections
- 83 (12%) out of the 687 sections are marked as important
- Tokenisation of sections
- Embedding and Long short-term memory (LSTM) Layers
- Binary Classification output

Most frequent Tokens

Token	Count
to	1938
move	1595
exit	1361
map	1114
ws	962
send	928
end	872
exec	868
3030	714
perform	708
thru	696

Dataset and Training of the RNN Model

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.1, random_state = 42)

X_train, X_val, Y_train, Y_val = train_test_split(X_train, Y_train, test_size = 0.3, random_state = 42)
```

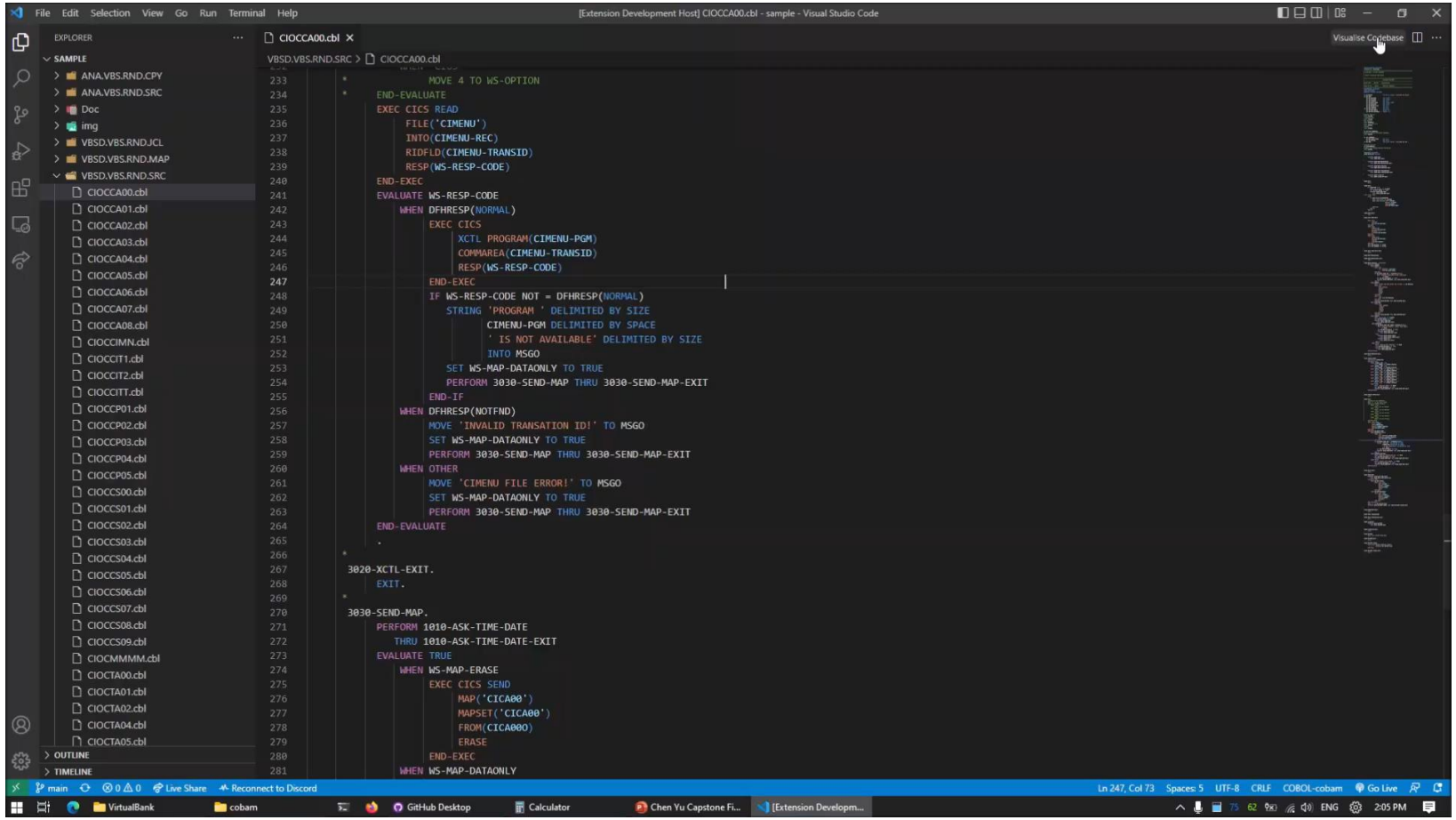
```
embed_dim = 32
lstm_out = 32

model = Sequential()
model.add(Embedding(max_fatures, embed_dim, input_length = X.shape[1]))
model.add(SpatialDropout1D(0.4))
model.add(LSTM(lstm_out, dropout=0.3))
model.add(Dense(2,activation='softmax'))
model.compile(loss = 'categorical_crossentropy',
optimizer=keras.optimizers.Adam(0.0005),metrics = ['accuracy'])
```

Dataset	Size
Total	687
Train set	432
Validation set	186
Test set	69

Codebase Visualisation

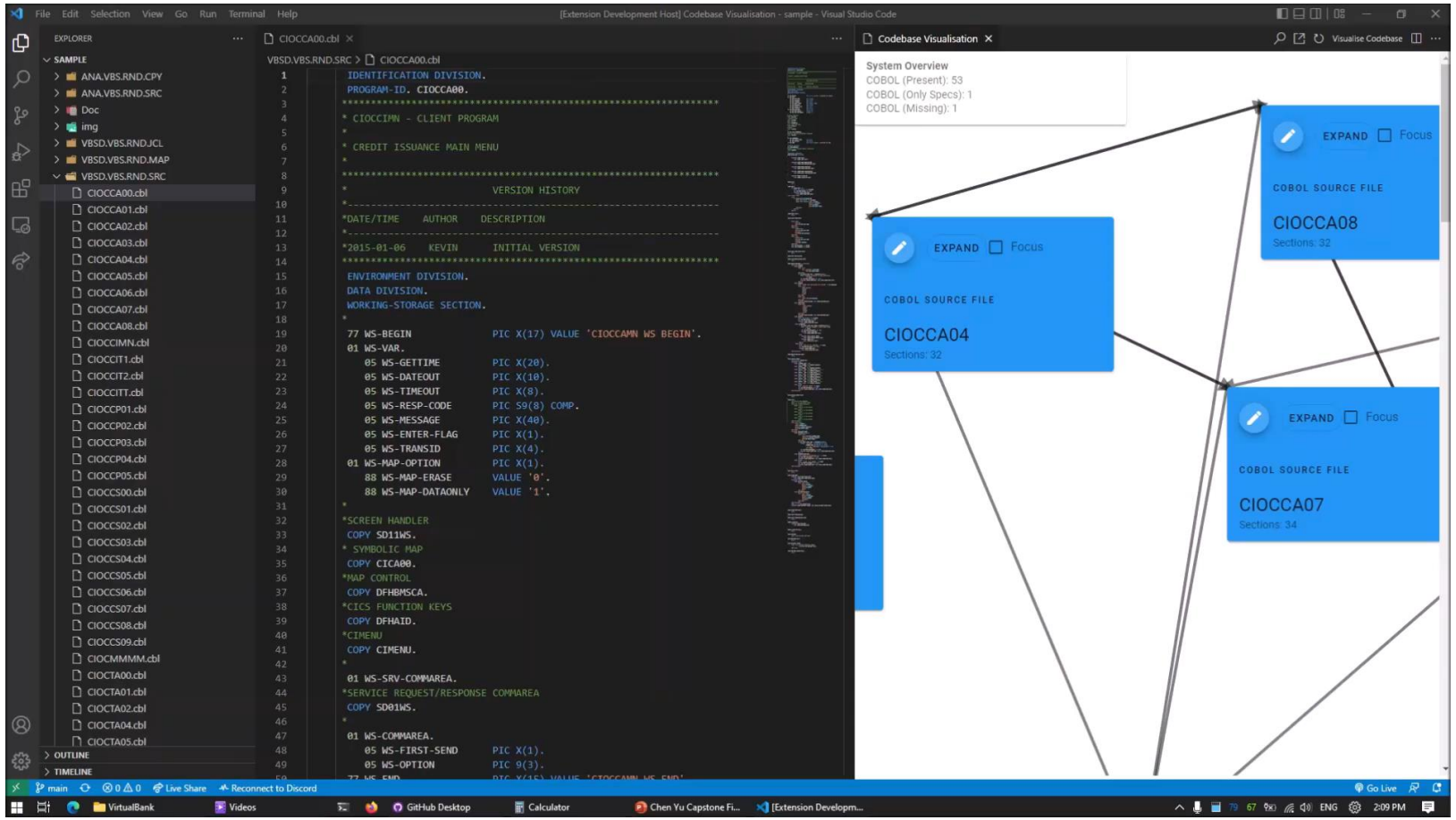
Features



Playback: https://giant.gfycat.com/HighMatureHoiho.mp4

Codebase Navigation

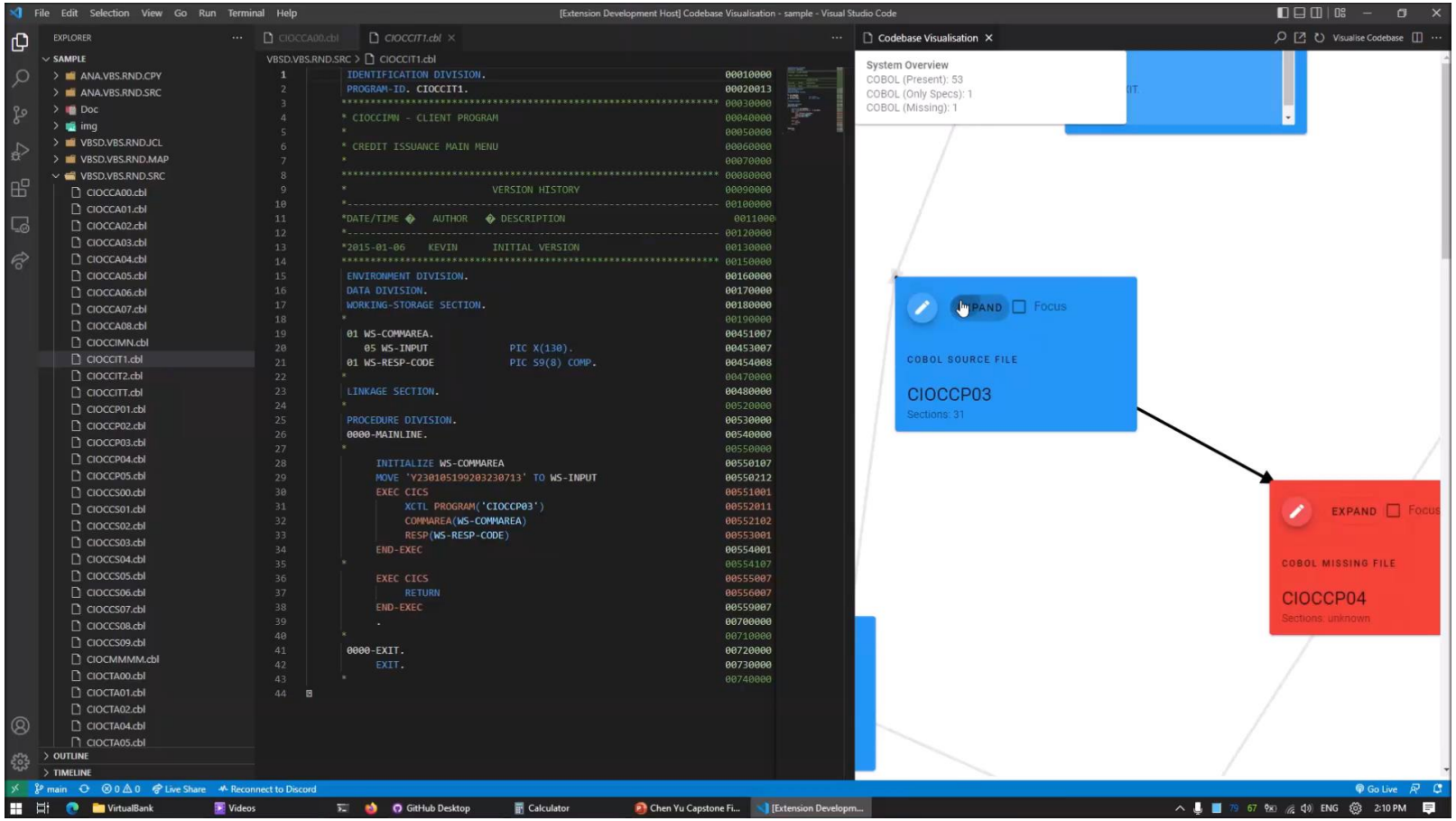
Features



Playback: https://giant.gfycat.com/EsteemedDisfiguredAardvark.mp4

Codebase Analysis with Machine Learning

Features



Playback: https://giant.gfycat.com/MeanUnequaledDoe.mp4

Evaluation

Productivity Analysis

- Dependency tracing with partial codebase

Without the tool with plain notepad++ and folder navigation:

1. Open the starting section in Notepad++ and find the name of the next external call
2. Minimise and navigate to 2nd file
3. Search for the landing section
4. Search for the next section leading to another external call
5. Minimise and navigate to 3rd file
6. Search for the landing section
7. Search for the next section leading to another external call
8. Minimise and navigate to 4rd file
9. Search for the landing section
10. Search for the next section leading to another external call
11. Minimise and navigate to the missing 5th file
12. Unable to locate the file, affirming the file is missing
13. Attempt to search for specification file in explorer and documentation repository

With the tool

1. Open VS code workspace
2. Click on visualisation code base button
3. Pan and scroll to find 1st file
4. Pan and scroll to find 2nd file
5. Pan and scroll to find 3rd file
6. Pan and scroll to find 4th file
7. Seeing the 5th file is indicated as missing source code and specification

Evaluation

Productivity Analysis

- Finding useful business logics

Without the tool with plain notepad++ and folder navigation:

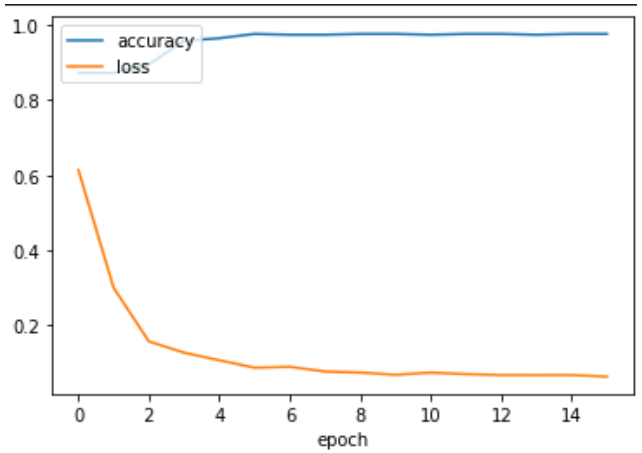
1. Open the source code in Notepad++
2. Scroll through various initial divisions such as data and working storage
3. Scroll through main section (procedure division)
4. Scroll through obsolete framework initialisation sections
5. Scroll through obsolete keyboard input handling sections
6. Scroll through obsolete user authorisation sections
7. Found first useful business function - such as retrieving personal information

With the tool

1. Open VS code workspace
2. Click on visualisation code base button
3. Pan and scroll to the node with target file
4. Look for the sections with importance indication star, click on it to jump to the code

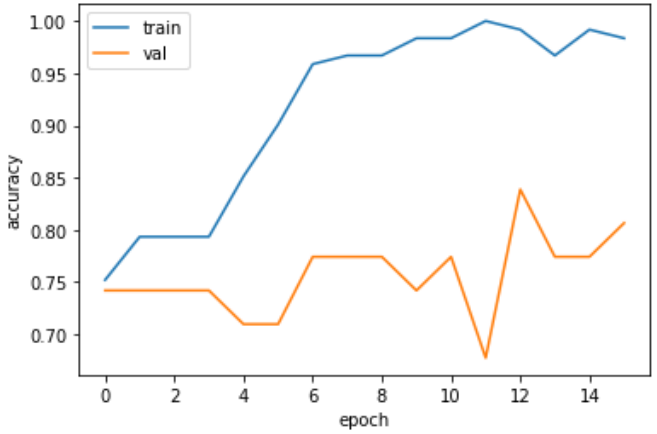
Evaluation

Importance Prediction Accuracy



Open source sample codebase

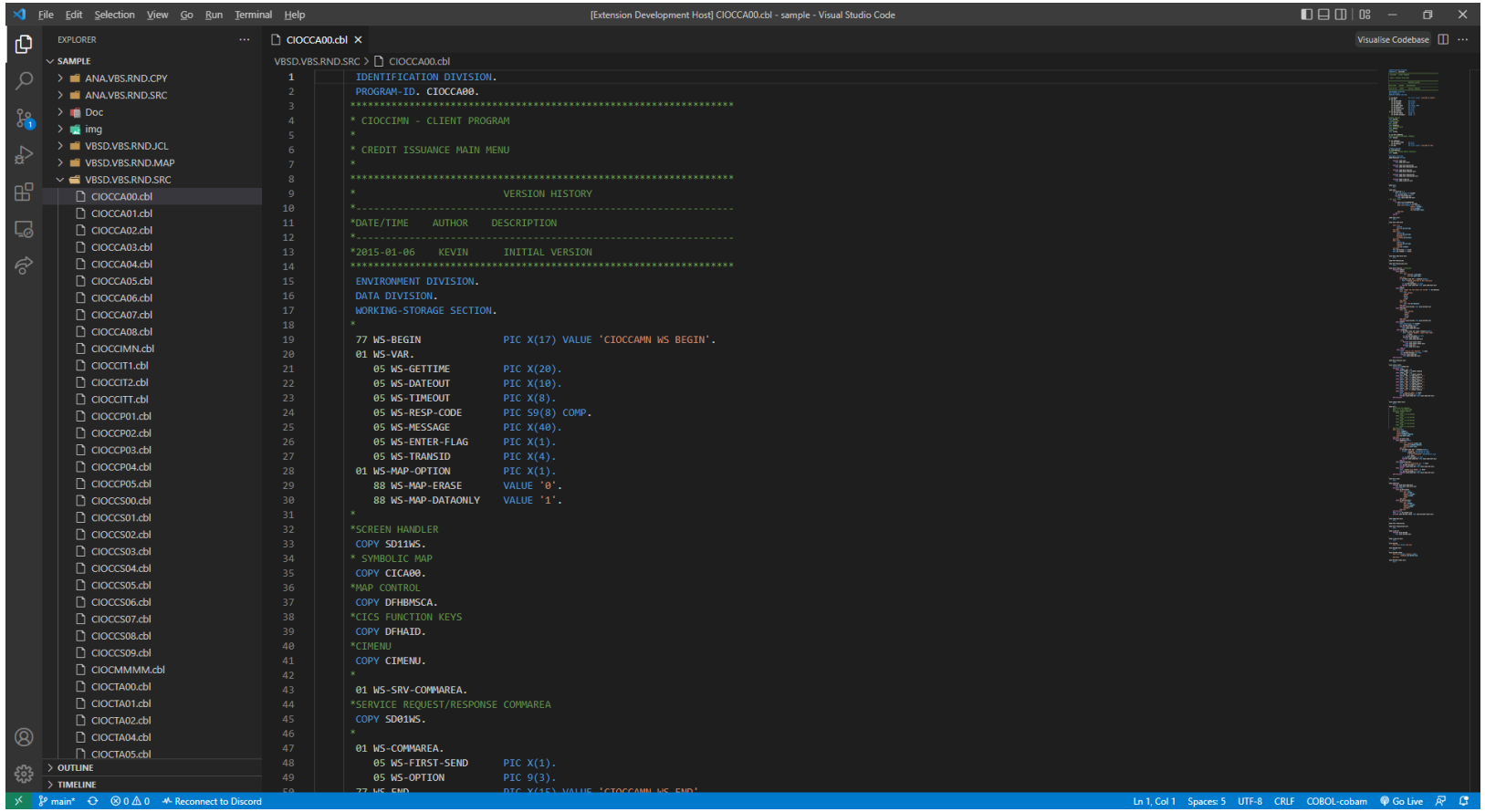
- 98% Accuracy
- 83% Mean Confidence when predicting a section is important



Company source code

- 89% Accuracy
- 69% Mean Confidence when predicting a section is important

Demo


File Edit Selection View Go Run Terminal Help
[Extension Development Host] CIOCCA00.cbl - sample - Visual Studio Code
EXPLORER
SAMPLE
ANA.VBS.RND.CPY
ANA.VBS.RND.SRC
Doc
img
VBSD.VBS.RND.JCL
VBSD.VBS.RND.MAP
VBSD.VBS.RND.SRC
CIOCCA00.cbl
CIOCCA01.cbl
CIOCCA02.cbl
CIOCCA03.cbl
CIOCCA04.cbl
CIOCCA05.cbl
CIOCCA06.cbl
CIOCCA07.cbl
CIOCCA08.cbl
CIOCCIMN.cbl
CIOCCIT1.cbl
CIOCCIT2.cbl
CIOCCITT.cbl
CIOCCP01.cbl
CIOCCP02.cbl
CIOCCP03.cbl
CIOCCP04.cbl
CIOCCP05.cbl
CIOCCS00.cbl
CIOCCS01.cbl
CIOCCS02.cbl
CIOCCS03.cbl
CIOCCS04.cbl
CIOCCS05.cbl
CIOCCS06.cbl
CIOCCS07.cbl
CIOCCS08.cbl
CIOCCS09.cbl
CIOCMMMM.cbl
CIOCTA00.cbl
CIOCTA01.cbl
CIOCTA02.cbl
CIOCTA04.cbl
CIOCTA05.cbl
OUTLINE
TIMELINE
CIOCCA00.cbl
VBSD.VBS.RND.SRC > CIOCCA00.cbl
Visualise Codebase
IDENTIFICATION DIVISION.
PROGRAM-ID. CIOCCA00.
****
* CIOCCIMN - CLIENT PROGRAM
*
* CREDIT ISSUANCE MAIN MENU
*
****
*   VERSION HISTORY
*---
*DATE/TIME   AUTHOR   DESCRIPTION
*---
*2015-01-06   KEVIN   INITIAL VERSION
****
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
*
77 WS-BEGIN   PIC X(17) VALUE 'CIOCCAMN WS BEGIN'.
01 WS-VAR.
05 WS-GETTIME   PIC X(20).
05 WS-DATEOUT   PIC X(10).
05 WS-TIMEOUT   PIC X(8).
05 WS-RESP-CODE   PIC S9(8) COMP.
05 WS-MESSAGE   PIC X(40).
05 WS-ENTER-FLAG   PIC X(1).
05 WS-TRANSID   PIC X(4).
01 WS-MAP-OPTION   PIC X(1).
88 WS-MAP-ERASE   VALUE '0'.
88 WS-MAP-DATAONLY   VALUE '1'.
*
*SCREEN HANDLER
COPY SD11WS.
* SYMBOLIC MAP
COPY CICA00.
*MAP CONTROL
COPY DFHBMSCA.
*CICS FUNCTION KEYS
COPY DFHAID.
*CIMENU
COPY CIMENU.
*
01 WS-SRV-COMMAREA.
*SERVICE REQUEST/RESPONSE COMMAREA
COPY SD01WS.
*
01 WS-COMMAREA.
05 WS-FIRST-SEND   PIC X(1).
05 WS-OPTION   PIC 9(3).
main*
0 0
Reconnect to Discord
Ln 1, Col 1
Spaces: 5
UTF-8
CRLF
COBOL-cobam
Go Live

Q and A