TOWARD A 'HALF-MACHINE, HALF-HUMAN' TRANSLATOR FOR LOCAL MALAY: A SIMPLE TRANSLATION APP

Wilham George Louhenapessy[1], Sunardo Panjaitan[2]
[1]Signifikan Solusi Integrasi (Signtegra), Dosen Tamu CIT Jakarta / UNPAR
[2]Jurusan Digital Economy Double Degree Univ. Padjadjaran & Tek. Informatika Univ. BINUS
wilham@signtegra.com; wilham.george@gmail.com

ABSTRAK

Penerjemahan dari satu bahasa ke bahasa lain membutuhkan sumber daya yang besar, termasuk tenaga kerja dan waktu, yang sering kali menyebabkan penundaan yang signifikan. Salah satu solusi yang mungkin adalah penerjemahan mesin, yang mengotomatiskan sebagian dari proses tersebut. Berbagai alat komputasi, seperti Google Translate, Babelfish, dan DocTranslator, telah dikembangkan untuk membantu tugas penerjemahan umum. Akan tetapi, alat-alat ini tidak mendukung banyak bahasa lokal, meskipun permintaan untuk penerjemahan tersebut tinggi. Di Indonesia sendiri, terdapat ratusan bahasa lokal, sehingga hal ini menjadi kebutuhan yang mendesak. Tantangan ini memotivasi penulis untuk mengembangkan mesin penerjemahan sederhana khusus untuk bahasa Melayu lokal.

Penulis pertama (Louhenapessy) mulai mengerjakan algoritma penerjemahan mesin pada tahun 2002, menggunakan FORTRAN untuk menerjemahkan bahasa Inggris ke bahasa Melayu Ambon (Louhenapessy 2006, 2007, 2008, 2009). Algoritma tersebut menerapkan diagram tata bahasa logis dan kriteria do-looping untuk menyusun proses penerjemahan. Kemudian, penulis kedua (Panjaitan) mengonversi algoritma tersebut ke VB.NET dan Python, sehingga meningkatkan kejelasan dan aksesibilitas output (Panjaitan 2015, 2018, 2024). Modifikasi ini membuat sistem lebih mudah digunakan sekaligus mempertahankan integritas komputasinya.

Algoritma umum untuk menerjemahkan bahasa Inggris ke bahasa Melayu lokal telah dikembangkan dan diuji menggunakan beberapa kalimat dari manuskrip Alkitab bahasa Inggris. Langkah pertama dalam algoritma tersebut adalah meniru tata bahasa Inggris menggunakan struktur yang sesuai dalam bahasa Melayu lokal. Langkah kedua melibatkan pengecekan kosakata untuk memastikan terjemahan yang akurat. Hasilnya menunjukkan bahwa algoritma tersebut berfungsi dan menciptakan gaya penerjemahan khusus, mirip dengan Amplified Bible. Algoritma tersebut menambahkan makna tambahan—seperti sinonim, penjelasan, dan kemungkinan terjemahan lain dalam tanda kurung—sehingga pembaca dapat lebih memahami makna yang lebih dalam dari teks Alkitab asli dalam bahasa Yunani atau Ibrani, terutama jika satu kata bahasa Inggris tidak cukup untuk menjelaskannya. Selain itu, kamus bahasa Inggris-bahasa Melayu lokal yang terbatas telah disusun berdasarkan kalimat-kalimat uji, yang berkontribusi pada peningkatan akurasi dan konsistensi.

Meskipun proyek ini masih berlangsung, makalah ini menguraikan algoritma penerjemahan dan penerapannya pada bahasa Melayu setempat. Penulis bermaksud untuk menilai apakah sistem semacam itu dapat mengotomatiskan setidaknya sebagian dari proses penerjemahan. Jika berhasil, metode ini dapat membantu menjembatani kesenjangan bahasa bagi banyak komunitas lokal, membuat informasi lebih mudah diakses dalam bahasa Melayu setempat.

Kata kunci: *Algoritma Mesin Penterjemah, Melayu Ambon, Fortran, Python*

ABSTRACT

Translation from one language to another requires substantial resources, including manpower and time, often causing significant delays. One possible solution is machine translation, which automates part of the process. Various computational tools, such as Google Translate, Babelfish, and DocTranslator, have been developed to assist general translation tasks. However, these tools do not support many local languages, despite the high demand for such translations. In Indonesia alone, there are hundreds of local languages, making this an urgent need. This challenge motivated the authors to develop a simple translation machine specifically for local Malay languages.

The first author (Louhenapessy) began working on a machine translation algorithm in 2002, using FORTRAN to translate English into Ambon Malay (See Appendix A, Louhenapessy 2006, 2007, 2008, 2009). The algorithm applies logical-grammar diagrams and do-looping criteria to structure the translation process. Later, the second author (Panjaitan) converted the algorithm into VB.NET and Python, improving the clarity and accessibility of the output (Panjaitan 2015, 2018, 2024). These modifications made the system easier to use while retaining its computational integrity.

A general algorithm for translating English into local Malay has been developed and tested using several sentences from English Bible manuscripts. The first step in the algorithm is to mimic English grammar using the corresponding structures in local Malay. The second step involves checking the vocabulary for accurate translation. The results show that the algorithm works and creates a special translation style, like the Amplified Bible. It adds extra meanings—like synonyms, explanations, and other possible translations in brackets—so readers can better understand the deeper meaning of the original Bible texts in Greek or Hebrew, especially when one English word

isn't enough to explain it. In addition, a limited English-to-local Malay dictionary has been compiled based on the test sentences, contributing to improved accuracy and consistency.
Although the project is still in progress, this paper outlines the translation algorithm and its application to local Malay language. The authors aim to assess whether such a system can automate at least part of the translation process. If successful, this method could help bridge the linguistic gap for many local communities, making information more accessible in their native local Malay languages
Kata kunci: *Keywords: Machine Translation Algorithm, Ambon Malay, Fortran, Python*

INTRODUCTION

Language translation across global languages has seen major technological advancements. However, local Malay[1] languages remain largely unsupported by mainstream machine translation tools such as Google Translate or DocTranslator (or earlier by Babelfish). This paper addresses the need for a tailored solution: a computational system that translates English into local Malay languages, starting with Ambon Malay.

Background and Motivation

The first author began this work in 2002, developing a machine translation algorithm using FORTRAN. This algorithm focused on converting English into Ambon Malay by using logical-grammar diagrams and looping structures. Publications from 2006 to 2009 document this early work. Later, the second author rewrote the algorithm in VB.NET and Python, improving the system's usability and readability without compromising its logic.

A significant step forward occurred in 2015 with the migration of the "Half Machine Half Human" algorithm from FORTRAN to Visual Basic. This transition not only modernized the technological foundation but also paved the way for a crucial architectural improvement: the extraction of the English-Malay Ambon dictionary into a separate file. This separation proved to be a game-changer. It allowed for easier expansion, modification, and maintenance of the vocabulary without requiring alterations to the core translation logic. The system became more modular and adaptable. (Panjaitan 2015 and 2018).

The relentless march of technology and the continuous pursuit of optimization led to another significant evolution in 2024. Recognizing the power and versatility of Python, we undertook the task of porting the Visual Basic code to this widely adopted language. The Python implementation not only offered enhanced performance and a richer ecosystem of libraries but also provided a more robust platform for future development and integration with other language processing tools. (Panjaitan 2024).

Throughout these technological shifts, the underlying principle of the "Half Machine Half Human" approach remained constant: a synergistic partnership between computational power and human expertise. The machine handles the rapid look-up and initial suggestion of translations based on the dictionary, while human input remains essential for disambiguation, ensuring cultural appropriateness, and refining the nuanced meaning of the text.

METHODOLOGY

The translation approach is built on two primary steps:

- Grammar Emulation: The system mimics English grammar structures using equivalents in local Malay.
- Vocabulary Matching: An internal dictionary is used to match English words with appropriate Malay counterparts.

[1] Examples of Local Malay are Manado Malay, Papua Malay, and Kupang Malay. They are indeed spoken in Indonesia. It's important to understand that Malay has evolved into many distinct varieties across the Indonesian archipelago. These varieties are often called "Malay" by their speakers but have differences in vocabulary, pronunciation, and grammar.
Manado Malay: This variety is spoken in and around the city of Manado in North Sulawesi.
Papua Malay: Also known as Irian Jaya Malay, this is a creole language spoken in Papua and West Papua provinces. It has been influenced by local Papuan languages.
Kupang Malay: This variety is spoken in and around Kupang, the capital of East Nusa Tenggara province.
The existence of these varieties highlights the linguistic diversity of Indonesia and the way Malay has adapted and changed as it spread throughout the region.

When ambiguity arises, the system inserts alternate meanings or explanations in brackets, creating a style akin to the Amplified Bible. The algorithm's core relies on rule-based translation, implemented with clear syntax mapping and conditional logic. For instance, it detects subject-verb-object (SVO) patterns in English and rearranges them to match Malay sentence construction.

To illustrate the fundamental steps involved in this translation process, consider the following simplified diagram:

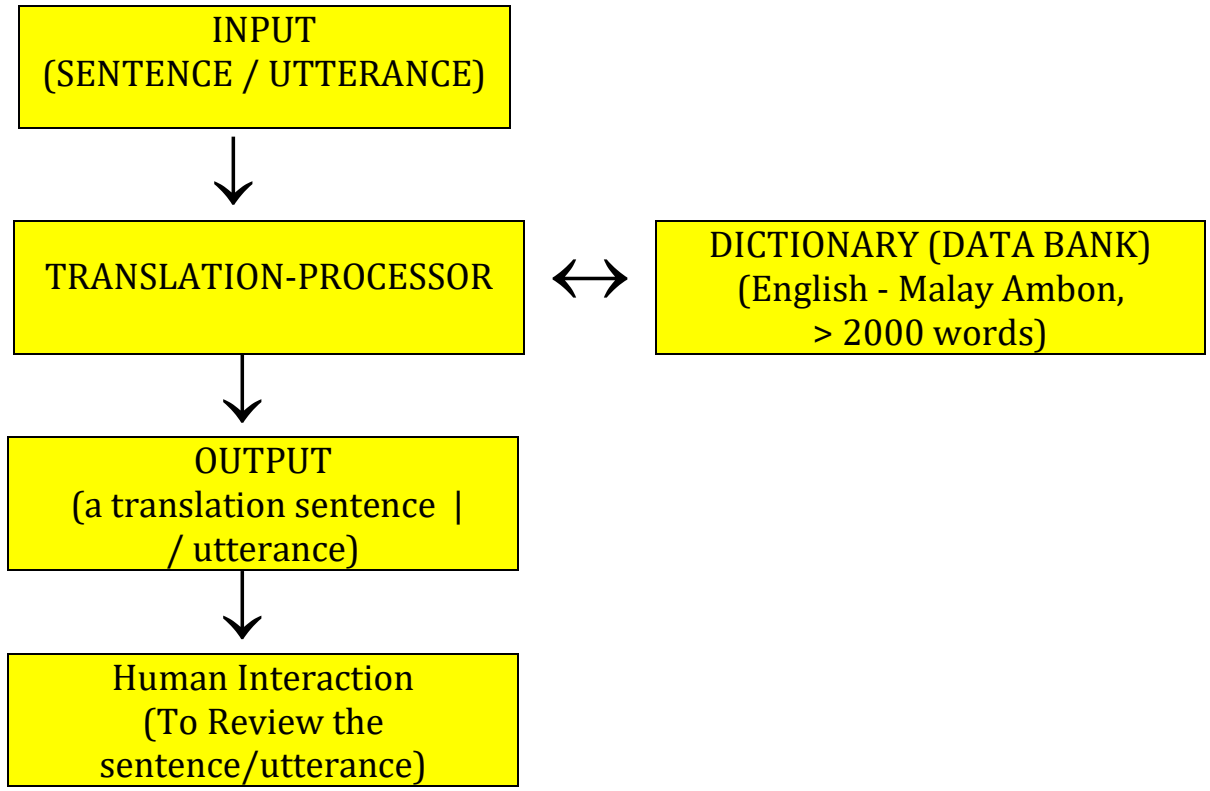

Figure 1. Translation Process[2]

Description of the Translation Process

The process begins with the **input** of a sentence or utterance in the source language (in this case, English). This input is then fed into the **translation processor**, which forms the core of the **Half-Machine, Half-Human algorithm**. The primary function of the translation processor is to analyse the input and identify corresponding translations for its constituent words. To achieve this, it interacts dynamically with the **dictionary (data bank)**, which contains a growing collection of English words and their corresponding

[2] Explanation of the Flowchart Elements:

- INPUT (SENTENCE / UTTERANCE): This is the starting point. It represents the sentence or phrase in the original language (English) that you want to translate.
- TRANSLATION-PROCESSOR: This is the "engine" of your translation system. It's the algorithm or code that does the work of converting the input into the target language.
- DICTIONARY (DATA BANK) (English - Malay Ambon, >2000 words): This is a crucial resource. It stores the vocabulary, the words and their translations, that the Translation-Processor uses. The double-ended arrow between the "Translation-Processor" and the "Dictionary" indicates that the processor both retrieves information from (looks up words) and potentially updates (though less likely in a simple translation) the dictionary.
- OUTPUT (a translation sentence / utterance): This is the pre-final result, of the computer translated sentence or phrase in the target language (Malay Ambon).
- Human Interaction: To review the output (from the machine), and this is the final result, the translated sentence or phrase in the target language (Malay Ambon).

translations in Malay Ambon—currently exceeding 2,000 entries. The translation processor queries this dictionary to retrieve potential translations for each word in the input sentence. After processing the input and consulting the **dictionary**, the system generates the **output**: a translated sentence or utterance in the target language (Malay Ambon). The double-headed arrow between the translation processor and the dictionary represents the continuous exchange of information between these two components throughout the translation process.

Most recently, in 2025, the English–Malay Ambon dictionary underwent substantial enrichment. Recognizing the importance of comprehensive lexical coverage, we focused on expanding the dictionary to include a wider range of linguistic categories—verbs, adjectives, adverbs, and nouns. This expansion is intended to enhance the algorithm's ability to handle more complex sentence structures and convey a broader spectrum of meaning.

Testing and Results

To test the algorithm, the authors used a sample of English Bible sentences. These texts were chosen because of their formal language structure and semantic richness. The output included bracketed alternatives and synonyms, increasing interpretability for local readers (Appendix B). A limited English–Malay Ambon dictionary was also generated, enhancing word-level accuracy. The translation outputs showed that the algorithm can produce meaningful results, even with limited data. The bracketed amplification technique made complex theological or abstract ideas more accessible to local speakers unfamiliar with Greek or Hebrew concepts.

DISCUSSION

The current system operates as a hybrid: part machine, part human—a "half-machine, half-human" translator. It automates structural and lexical translation but allows human editors to refine and contextualize the results. This division ensures that nuanced language remains faithful to its cultural context, a task that full automation still struggles to achieve.

Conclusion and Future Work

The preliminary results are promising. The hybrid system effectively bridges the gap between machine efficiency and human insight.

Future work includes:

- Expanding the dictionary
- Integrating speech-to-text functionality
- Testing with other local Malay dialects
- User interface development for mobile platforms
- Using AI tools in enrichment of the Dictionary file

Such innovations may help preserve local languages, improve information accessibility, and reduce dependency on human translators for routine tasks.

CLOSING

The journey of the "Half Machine Half Human" algorithm—from its humble beginnings in FORTRAN to its current form in Python, enhanced by a more comprehensive dictionary—underscores the dynamic nature of translation in the digital age. It illustrates the power of combining computational tools with deep linguistic knowledge to bridge language barriers, especially for languages that lack the technological support enjoyed by more widely spoken ones. This ongoing development testifies to the enduring value of both human insight and technological innovation in achieving effective, meaningful translation.

REFERENCES

Albahari, J., & Albahari, B. 2021. *C# 9.0 in a Nutshell.* (Ref. for .NET framework.

Bolton, R., Riupassa, M and J. Tjia. 2005. *Pedoman Membaca dan Menulis Bahasa Ambon (Edisi Percobaan) (booklet)*. 24 pages.

Chelladurai, Xavier. 2024. FORTRAN 77 and Numerical Methods Paperback – New Age International Publisher

Joss Moorkens, Joss, Way. A., Lankford, S. 2024. *Automating Translation.* Routledge

Kadir, Abdul. 2018. *Dasar pemrograman python 3 : panduan untuk mempelajari python dengan cepat dan mudah bagi pemula*. CV Andi Offset

Koehn, Philipp 2012 (Online Publication Date), Statistical Machine Translation, Cambridge University Press, Cambridge, UK.-

Louhenapessy, Wilham G. 2007. *Tuhan Yesus Ajar Katong Dolo.* Majalah ASSAU, PIKOM GPM. Sept.-Oct. 2007 Ed. Publish by: PIKOM GPM, Ambon. Indonesia. Page. 38-39. ISSN 1412-7881.

Louhenapessy, Wilham G. 2008. *Tuhan Yesus Ajar Katong Dolo: Kelahiran Tuhan Yesus (Kajadiang)* Majalah ASSAU, PIKOM GPM. Nov.-Dec. 2008 Ed.. Publish by: PIKOM GPM, Ambon. Indonesia: ISSN 1412-7881.

Louhenapessy, Wilham G. 2008. *Kres Bajalang Jau Par Cari Dame Yang Tuhan Yesus Su Kasi.* Majalah ASSAU. Jan.-Feb. 2008 Ed. Publish by: PIKOM GPM, Ambon. Indonesia: ISSN 1412-7881.

Louhenapessy, Wilham G. 2007. *Tulisan-tulisan dan rekaman-audio bahasa Melayu-Ambon anak-anak di Maluku Tengah (Seram Utara, Telutih, Tuhaha dll):* Seminar Nasional Bahasa dan Sastra Indonesia 2007. Universitas PATTIMURA, PGSD, 25 Oktober 2007

Louhenapessy, Wilham G. 2008. *The Malay-Ambon Writings and Audio-recordings of Children in Central Maluku (Seram Utara, Telutih, Tuhaha etc.)*: Heritage Language Literacy Development in SE Asia. PBTM, Atma Jaya University, SIL International, KD, International Reading Association and IDAC, JULY 22-23 2008. in Jakarta, INDONESIA.

Microsoft Docs. 2023. *Visual Basic Guide.* https://learn.microsoft.com/en-us/dotnet/visual-basic/

Minde, Don van. 1997. *Melayu Ambong: Phonology, Morphology, Syntax.* Research School CNWS, Leiden University. The Netherland: 390 pages. ISBN 90-73782-94-5

Panjaitan, Sunardo. 2015, 2018 and 2024 (communications and discussions)

Sweigart, A. (2019). Automate the Boring Stuff with Python. No Starch Press.

Van Rossum, G. 2009. *The Python Language Reference Manual.* Python Software Foundation.

Website 2025. https://thepythoncode.com/article/translate-text-in-python

Zelle, J. (2017). Python Programming: An Introduction to Computer Science. Franklin, Beedle & Associates Inc.

Curriculum Vitae

Complete Name	Institution	Education	Research Interests
Wilham George Louhenapessy M.Sc. Ph.D.	SIGNTEGRA	Ph.D.	Computational Machine Translation
Sunarso Panjaitan	Jurusan Digital Economy Double Degree Univ. Padjadjaran (Jurusan Ekonomi Terapan) dan dan Univ.Bina Nusantara)	Jurusan Teknik Informatika	Computational Machine Translation

Appendix A

```
      PROGRAM HalfMach_HHpre7and650
c ------------------------------------------------------------- c
c Program by Dr. Wilham George Louhenapessy
C This program is to help Dr Louhenapessy
C to do English translation into Malay Ambon dialect
C Renew R8 in 3rd Aug 2005
C Based on previous work (R3) of Thursday 4th April 2004
C ============================================================= c
      CHARACTER*5000 K4
      CHARACTER*5000 K5
      CHARACTER*5000 K6
      CHARACTER*5000 K7
      CHARACTER*34 ENG_dW0R(1500)
      CHARACTER*40 MAB_dW0R(1500)
      CHARACTER*34 ENGW0R(50)
      CHARACTER*34 ENGJawa(50,50)
      CHARACTER*40 MABW0R(50)
      CHARACTER*12 FILE
      CHARACTER*12 JAWAB

      DIMENSI0N IJawa(50)
C-----0PEN FILES

      WRITE(*,1000)
      WRITE(*,1010)
      READ(*,1020)FILE
c     1000 F0RMAT(//T10,'Wilham FIRST TRY Thu 20 May 2004',/,
 1000 F0RMAT(//T10,'Wilham at Cilandak Wed 3rd August 2005',/,
     .T10,46(1H=),/,T10,'-------------------------------------'
     .,/,T10,'Half Machine Half Human',/,T16
     .,/,T10,'- - - - - - - - - - - - -',/,T16,
     .'C0PYRIGHT Wilham G. Louhenapessy '/T22,'Ver 1.0, Aug 2005'
     ./T22,22(1H-)//)
 1010 F0RMAT(/' Masukan nama file INPUT DATA [Maks. 8 huruf]'/)
 1020 F0RMAT(A)
       0PEN(UNIT=1,FILE='ENGword1.txt',STATUS='unknown')
       0PEN(UNIT=2,FILE='BTWword1.txt',STATUS='unknown')
c      CALL FILNAM(FILE,'dat')
c      0PEN(UNIT=1,FILE=FILE,STATUS='unknown')
      CALL FILNAM(FILE,'out')
      0PEN(UNIT=3,FILE=FILE,STATUS='unknown')
      CALL FILNAM(FILE,'txt')
      0PEN(UNIT=4,FILE=FILE,STATUS='unknown')

      CALL INPUT(ENG_dW0R,MAB_dW0R)

      NLine = 650

      write(3,*) 'Number of line in dictio=',NLine
c      WRITE(*,5000)
c 5000 F0RMAT(/' Put a line number that The UTTERANCE interest you'/)
c 5000 F0RMAT(/' type any character please')

c      do 150 I=1,

      Jawa = 1
 100  Iw = 1
      Jawa  = Jawa + 1
c buat sementara tuk make layar setop / pause
c pausdfds
c      read(*,*) sdss
C      WRITE(*,'(A\)') ' to continue type any character PLEASE: '
C      READ (*,'(A)') outputfile

 125  READ(4,2010)  ENGW0R(Iw)
      ENGJawa(Jawa,Iw) = ENGW0R(Iw)
      ENGJawa(Jawa,Iw) = ENGW0R(Iw)
```

```
C     write(*,1035) ENGW0R(Iw)
2010  F0RMAT(A20)
c     write(3,*) 'Iw =',Iw,'   engwor[iw]=', ENGW0R(Iw)
      if(ENGW0R(Iw).E@.'999') then
      Verse = 0
C     Equation for translation
      [redacted]
      goto 150
      else
         if(ENGW0R(Iw).E@.'9999') then
C     Equation for translation
      [redacted]
         goto 150
         else
         endif
      Iw = Iw + 1
      goto 125
      endif
 150  continue
      NoofCha = Iw - 1
      NoofCh2 = Jawa - 1
      write(3,*) '1NoofCha=',NoofCha,'[   ]NoofCh2=',NoofCh2,'['
c     write(*,*) '1NoofCha=',NoofCha,'[   ]NoofCh2=',NoofCh2,'['
      write(*,*) '12345678901 2345678901 2345678901 2345678901'
C BENTAR
c data engword dari 1 pet... 16 kata
C NoofCha ----> sama dengan 16
      do 200 J=1,NoofCha
      do 300 K=1,NLine
C     First "IF-THEN"
      IF  [redacted]THEN
C     FIRST EQUATION 1
      [redacted]
      G0T0 200
      ELSE
C     FIRST EQUATION 2
      [redacted]

      ENDIF
 300  C0NTINUE
 200  C0NTINUE
c     write(3,*) 'NoofCha = ',NoofCha
c     NoofCha = 16
C     SEC0ND EQUATI0N 1 & 2
      [redacted]
      J=NoofCha-1
      do 400 I=0,NoofCha-1
c     write(3,*) '....',I,NoofCha,J
      IDELTA = J-I
C     SEC0ND "IF-THEN"
      IF [redacted] THEN
      goto 400
      else
C     SEC0ND EQUATI0N 3 & 4
      [redacted]
      end if
 400  continue
      WRITE(*,1030) K5
      write(3,1030) K5
      write(3,1030) K4
c [sementara]
```

```
c sementara berikut di   PAUSE
      write(*,1030) K4
      write(3,*) ' ----------- new verse ----'
1030 FORMAT(A1000)
1035 FORMAT(A100)
c1030  FORMAT(A250)
      if(Verse.eq.1) goto 500
      GOTO 100
C     stop
 500  continue
      write(3,*) ' NEW LOOPING TO CHECK ... '
      write(3,*) '    '
      write(3,*) '    '
      write(3,*) 'after 500 CONTINUE'
      write(*,*) 'how many utterance ',Jawa-1
      write(3,*) 'how many utterance ',Jawa-1

c to choose interest UTTERANCE no X
c                   and automatically X - 1 and X + 1
      write(*,*) 'You have',Jawa-1,'    utterances'
      write(*,*) 'What number you might interest with [choose 1 ONLY]'
      read(*,*) Ninteres
      Nminu1  = Ninteres - 2
      Nplus1  = Ninteres
      do 8200 Jawa=Nminu1,Nplus1
c     write(3,*) 'Utterance No',Jawa      c
c     write(3,*) 'with number of WORDS is',IJawa(Jawa)
      K6 = ENGJawa(Jawa,IJawa(Jawa))
      J2=IJawa(Jawa)-1
      do 8400 I2=0,IJawa(Jawa)-1
c     write(3,*) '....',I,NoofCha,J
      IDELTA2 = J2-I2
      if(IDELTA2.EQ.0) then
      goto 8400
      else
      K6 = ENGJawa(Jawa,(J2-I2))//K6
      IF(Jawa.eq.Ninteres) then
      K7 = K6
      else
      endif

      end if
8400  continue
      WRITE(*,1030) K6
      write(3,1030) K6

8200  CONTINUE

      WRITE(*,'(A\)') ' Is there any refference to be clarifying '
      READ (*,'(A)') JAWAB
      if(JAWAB.EQ.'YES') goto 7777
      if(JAWAB.EQ.'yes') goto 7777

      write(*,*) 'stop sini before cek continue'
      stop

7777  WRITE(*,*)' In what number the word to be clarifying appear?'
      READ(*,*) IClarify

      WRITE(*,1030) K7
      write(3,1030) K7

      WRITE(*,'(A\)') ' WHAT YOU SHOULD CHANGE THEN INTO?'
      READ (*,'(A)') CLARYFY
      END
C*************************************************************
      SUBROUTINE INPUT (ENG_dWOR,MAB_dWOR)
c      IMPLICIT REAL*8(A-H,O-Z)
```

```
C-----GLOBAL DIMENSION
      CHARACTER*34 ENG_dWOR(1500)
      CHARACTER*40 MAB_dWOR(1500)

       ENG_dWOR(1)='[away]'
       ENG_dWOR(2)='[day]'
       ....(3), (4) and so on .....
       ENG_dWOR(650)='awake'
c  ---------------------------------------
       MAB_dWOR(1)='[pigi]'
       MAB_dWOR(2)='[hari]'
       ....(3), (4) and so on .....
       MAB_dWOR(650)='....'
      RETURN
      END
C*****FILNAM*****************************************************************
      SUBROUTINE FILNAM (FILE,EXT)
C
      IMPLICIT REAL*8(A-H,O-Z)
C
      CHARACTER*1 FILE(12),EXT(3)
      DO 500 I=1,8
      II=I
      IF(FILE(I).EQ.'.') GO TO 520
  500 CONTINUE
      II=9
  520 FILE(II)='.'
      DO 550 I=1,3
  550 FILE(II+I)=EXT(I)
      RETURN
      END
```

Appendix B