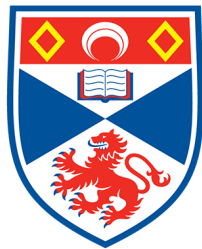


# TranslPrime

A L2 Reading Assistance Application For Browser-Based  
Reading



University of  
St Andrews

Yuxuan Wang

160000139

Supervisor: Dr. Mark-Jan Nederhof

Date of Submission: 21/05/2021

## **Abstract**

With the thriving of Internet and web-based technologies, increased attention is given to second-language (L2) reading and learning using browser. People are more exposed to different kinds of L2 texts on the Internet and consequently a substantial amount of L2 reading and learning assistance applications are developed.

However, little amount of applications are developed for targeting both L2 reading assistance and L2 learning assistance in the context of browser reading. This research project aims to develop and implement a web browser extension, TranslPrime, for providing timely and reactive L2 reading and learning assistance.

This report discusses the background knowledge of L2 reading and learning. It also documents the development approach, design decisions, implementation considerations and evaluations of TranslPrime.

## **Declaration**

I declare that the material submitted for assessment is my own work except where credit is explicitly given to others by citation or acknowledgement. This work was performed during the current academic year except where otherwise stated.

The main text of this project report is 8165 words long, including project specification and plan.

In submitting this project report to the University of St Andrews, I give permission for it to be made available for use in accordance with the regulations of the University Library. I also give permission for the title and abstract to be published and for copies of the report to be made and supplied at cost to any bona fide library or research worker, and to be made available on the World Wide Web. I retain the copyright in this work.

## **Acknowledgement**

I would like to extend my sincere thanks to my supervisor Dr. Mark-Jan Nederhof for his valuable assistance at every stage of the research project especially during the lockdown period. The discussions on L2 reading competence and the research direction were invaluable for this project.

I would also like to thank all participants for evaluating TranslPrime and providing their valuable suggestions.

*Yuxuan Wang*

# Contents

<b>1 Introduction</b>	<b>5</b>
1.1 Objectives & achievements	5
1.2 Report structure	6
<b>2 Context Survey</b>	<b>7</b>
2.1 Background	7
2.1.1 Reading competency	7
2.1.2 L2 reading	8
2.1.3 L2 reading difficulty & assistance	9
2.2 State-of-the-art	10
2.2.1 Translation applications	11
2.2.2 TaaS applications	12
2.2.3 L2 reading & learning assistance applications	12
2.2.4 L2 proficiency assessments	13
<b>3 Software Engineering</b>	<b>14</b>
3.1 Requirements specification	14
3.1.1 Requirement adjustments	14
3.1.2 Assessment subsystem cancellation	15
3.1.3 Functional requirements	15
3.1.4 Nonfunctional requirements	16
3.2 Software engineering process	17
3.3 Ethics	17
<b>4 Design</b>	<b>19</b>
4.1 Base layer: browser extension framework	19
4.1.1 Browser extension & browser engine	19
4.1.2 Browser extension components	20
4.2 Top layer: translation module	21
4.2.1 Text tokenisation and translation rules	22
4.2.2 API “plug-and-socket” solution	22
4.3 Top layer: note module	23
4.3.1 Note structure & storage	23
<b>5 Implementation</b>	<b>25</b>
5.1 Browser extension	25
5.1.1 Manifest access limitations & Content Security Policy (CSP)	25
5.1.2 Manifest V3	25
5.1.3 Asynchronous event-based communication approach	26
5.1.4 Assistance menu construction & injection	27
5.2 Translation module	28
5.2.1 Source & target language choices	28
5.2.2 Text tokenisation	29
5.2.3 API “plug-and-socket” implementation	29

5.2.4 Assistance menu: translation display . . . . .	29
5.2.5 Default APIs & API plug interface . . . . .	30
5.3 Note module . . . . .	31
5.3.1 Note display . . . . .	32
5.4 User assistance . . . . .	33
<b>6 Evaluation &amp; Critical Appraisal</b>	<b>34</b>
6.1 User evaluation result . . . . .	34
6.2 Heuristic evaluation . . . . .	35
6.3 Known design and implementation defects . . . . .	36
<b>7 Conclusion</b>	<b>38</b>

# 1 Introduction

## 1.1 Objectives & achievements

This project provides a second-language (L2) reading assistance application for browser-based reading called TranslPrime. It is aimed to reinforce users' L2 reading competence by providing L2 translations and supporting L2 learning activities.

With the development of NLP technologies and the thriving of L2 learning, a substantial amount of translation applications and language learning applications are developed. The translation applications are capable of providing comprehensive linguistic explanations for words, sentences and even articles that are written in different natural languages. Alternatively, the language education applications are capable of providing personalised online language lessons from different literacy aspects. Therefore, although people are more likely to encounter prints written in various foreign languages, the costs for understanding L2 texts or learning a new L2 are continuously reduced.

Nevertheless, in the context of browser-based reading (i.e., reading articles or files using a web browser), little number of applications are capable of providing translation and language learning assistance at the same time. It is too time-consuming for readers to copy-and-paste texts from the browser and wait for translations using an individual translation application. On the other hand, the language learning application often uses fixed material for educational purposes and most of these applications do not support arbitrary article translation.

Therefore, TranslPrime is designed and implemented as a hybrid application aimed for both L2 reading and learning assistance. It is a web browser extension that provides an additional reading assistance menu when reader selects a piece of texts in the browser. The assistance menu provides translation services in terms of whole sentence translation (i.e., translating the meaning of a whole sentence from the source language to the target language) and word lexical explanations (i.e., tokenising a sentence into distinct words and providing the paraphrase for each word), together with a note input interface that allows users to take arbitrary notes along with the translations.

The final deliverables of TranslPrime research include:

- The context survey on L2 reading assistance applications, including:
  - The definition of L2 reading competence and what makes L2 reading difficult from the perspective of applied linguistics.
  - The state-of-the-art research on current L2 reading and learning assistance research and applications.
- The web browser extension based on Chromium framework that provides

L2 reading and learning assistance.

- The analysis of software usability in terms of user evaluation and heuristic evaluation.

## **1.2 Report structure**

This project report includes the following chapters:

- Context survey: the context survey on L2 reading competence and difficulties, and state-of-the-art research in both applied linguistics and computer science fields.
- Software engineering: the requirement specification, software development process and other software engineering aspects of TranslPrime.
- Design & implementation: the principle design decisions and implementation considerations of TranslPrime.
- Evaluation: the application analysis in terms of user evaluation and heuristic evaluation.

## 2 Context Survey

### 2.1 Background

#### 2.1.1 Reading competency

*...when we think of the different purposes for reading and the varying processes that are called into play, it is evident that no single statement is going to capture the complexity of reading.* [1]

Reading is one of the inborn skills of the human being to receive and process information. It serves not only as the “input interface” of human cognition but also the foundation of cognitive behaviours beyond input processing:

- First of all, reading is an irreplaceable process of **comprehending** the writer’s intention. In terms of applied linguistics, reading comprehension involves making graphemic-phonemic connections (i.e., associating texts with their vocal sounds) and recognising grammatical structures using morphological, syntactic and semantic knowledge [1]. Specifically, Koda [2] indicates that reading comprehension occurs when readers extract information from texts and integrates information with what is already known.
- On the basis of comprehension, reading is one of the progressively evolved **learning** skills: learning implies identifying new concepts from information and acquiring new knowledge through reflection and critique. Consequently, the accumulation of knowledge is going to induce the development of reading abilities.
- Reading is also inherently a learning process with subconscious text **evaluations** [1] in two directions: both writers (e.g., *do I still have an interest in the texts?* or *do I agree with what the writer intended to convey?*) and readers themselves (e.g., *how well can I understand what the writer conveys?*).

In the daily life, reading is one of the survival skills for social activities and communications because people are surrounded by a substantial amount of texts in the modern society. The competency of reading has diverse definitions from different perspectives due to the fact that the contexts and purposes of reading are subject to change. Consequently, it is impossible to conclude a trivial definition for reading and reading competence regardless of reading contexts and purposes.

Defining the nature of reading competence from various perspectives is an important topic for reading research. Among various research perspectives, the cognitive perspective emphasises the interactions between readers and texts. Therefore, it is the most suitable perspective within the context of this report.

In Koda’s research [2], reading can be subdivided into three consecutive processes from the cognitive perspective, as depicted by figure 1:



1. *Decoding* is the process of extracting information from texts using linguistic knowledge.
2. *Text-information building* is the process of integrating extracted information to reveal text meanings.
3. *Situation-model construction* is the process of synthesising integrated information with readers' prior knowledge.

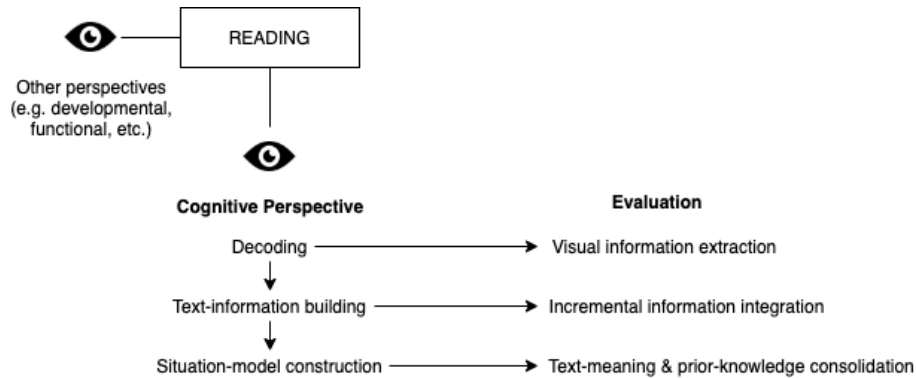


Figure 1: Reading competence and evaluation in cognitive perspective

### 2.1.2 L2 reading

...But it is also fair to say that, for millions of people, *L2 reading skills* represent a significant concern as these people negotiate careers and seek advancement in modern economies. [1]

The prosperity of Internet and network-based communications has greatly expedited the global information flow. People live in modern society are more exposed to texts and information that are expressed in foreign languages. Consequently, foreign language reading and learning have received increased public attentions and research interests. Computer-scientific research on aiding foreign language reading and learning has also increased both in number and in quality unprecedentedly.

In applied linguistics, the term *L2 reading*, as the opposite of *L1 reading*, refers to reading in a non-L1 language. More specifically:

- The term *L1* refers to the *first language* or the *native language*, which is the language(s) one used the most frequently or the first language one learnt to read and comprehend.

- The term *L2* refers to the collection of *second language* and *foreign language*. Although these terms are defined in several ways in different linguistic research, reading in L2 generally implies reading using the language(s) that is learnt or acquired consciously[3]. Hence, the term *L2 readers* can be used to refer to the group of readers who are literate in their L1s trying to read in another language(s).

The differences between L1 reading and L2 reading do not merely consist of linguistic differences: L2 reading entails cross-lingual information processing while L1 reading only involves monolingual information processing. That said, it does not imply that L2 reading is "more difficult" than L1 reading, as L2 readers are capable of using their prior reading and literacy experience. Besides, some factors that can affect the L2 reading comprehension[4] includes: vocabulary knowledge, topic familiarity, situational interests and reading strategies.

On the other hand, as no two leaves are alike in a forest, different readers have completely dissimilar reading competence. Beside aforementioned L2 readers, there are young children who are not yet skilled in L1 and learn to read a L2 as a school lesson; there are also people who have poor literacy skills and immigrate to other countries due to inevitable reasons. These L2 readers are likely to have more difficult L2 reading and learning experiences compared to literate L2 readers.

### 2.1.3 L2 reading difficulty & assistance

From the cognitive perspective, the difficulties of reading can be deemed as cognitive barriers that interrupt the regular reading flow:

- In the decoding process, the information extraction can be interrupted by unrecognisable linguistic structures. For instance, some words or grammatical structures cannot be understood by the readers.
- In the text-information building process, the information integration can be interrupted by incoherent context or text misinterpretations. For instance, a piece of text is selectively skimmed or garbled by the readers
- In the situation-model construction process, the information combination can be interrupted by insufficient prior-knowledge or knowledge misinterpretations. For instance, a complex sentence in a scientific article may not be successfully interpreted by a nonscholastic reader who understand the meaning of words in the sentence.

In addition to these cognitive interruptions, L2 reading further increases the reading difficulties by adding cross-lingual interference. As an illustration, Grabe[5] describes the linguistic difference between the L1 and the L2 for a L2 reader as *linguistic distance*. The "longer" the linguistic distance (i.e., the more distinct the two languages), the more likely the language processing interference will appear.

Therefore, providing L2 reading assistance implies **helping L2 readers to reaching high level of L2 reading competence by removing the cognitive barriers and reducing the language processing interference**. In order to keeping the cognitive processes fluently and avoiding any further interrupts, the assistance should be provided **timely**, the assistance texts should express the meaning of the original text **precisely**. Figure 2, as an extension of figure 1, depicts L2 difficulties and corresponding assistance examples.

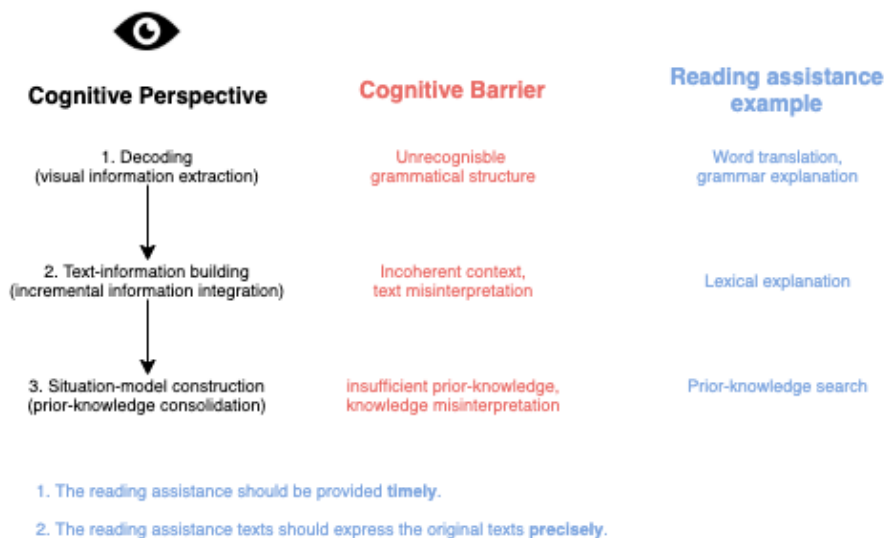


Figure 2: L2 reading difficulties & assistance examples

## 2.2 State-of-the-art

Before computer-based translation technologies are broadly applied, people were limited to using dictionaries or encyclopedias, or asking multilinguals, to translate a piece of foreign texts. The process of finding accurate translation resource, searching the translation and comprehending the translation in the original context would significantly interrupt cognitive processes and consequently eliminate readers' interests.

In the last two decades, the large-scale application of NLP and web-based technologies had significantly changed the evolution of L2 reading and learning assistance:

- The evolution of ML technologies, especially the rapid development of artificial neural network approach, had considerably improved the accuracy of text extraction and text translation under diverse scenarios.

- The emergence of cloud service had given rise to the development of **Translation-as-a-Service (TaaS)** applications which are capable of providing a wide variety of translation services based on different languages or educational purposes.
- The advancement of big data and storage technologies had enabled larger and more precise corpora to be developed.

### 2.2.1 Translation applications

The term “translation” generally refers to the process of reading a piece of text of a particular language (*source language*)  $T_{src}$  and yielding a piece of text in another language (*target language*)  $T_{tar}$  that has the equivalent linguistic meaning. For L2 readers, the source language is often the language of texts and the target language is often readers’ respective L1(s).

From the linguistic perspective,  $T_{tar}$  can be presented in multiple forms. For instance, when an English reader is reading a French article and observes an unrecognisable word  $T_{src} = Pomme$  and searches it in a translation application, the translation result can be presented in various ways: in terms of direct word-to-word translation, the translation could be  $T_{tar} = Apple$ ; While in terms of lexical explanation, the translation could be  $T_{tar} = A\ red\ round\ fruit\ with\ sweet\ taste$ .

Today, most of the translation applications provides comprehensive translations consisting of multiple  $T_{tar}$  for a given  $T_{src}$ . These services can be delivered through individual executables, web applications or programming APIs. Furthermore, these translation applications can be categorised into three clusters:

1. *Programming-focused application*: these applications provide a moderate amount of linguistic information with flexible programming supports. E.g., Google Translate<sup>[5]</sup>, which is one of the most popular translation platforms that is built within the Chromium, provides simple word-to-word translation with brief linguistic definitions and examples on its web application. On the other hand, it provide comprehensive translation API supports that allow programmers to write various kinds of customised translation applications.
2. *Language-focused application*: these applications focus on the linguistic knowledge behind the translation. E.g., Cambridge Dictionary<sup>[6]</sup> provides thorough linguistic supports including word paraphrases, grammatical explanations and thesauri.
3. *Public-contributed application*: these applications are usually in the form of web applications and the resource of translation is typically contributed by its users. E.g., Wiktionary<sup>[7]</sup>.

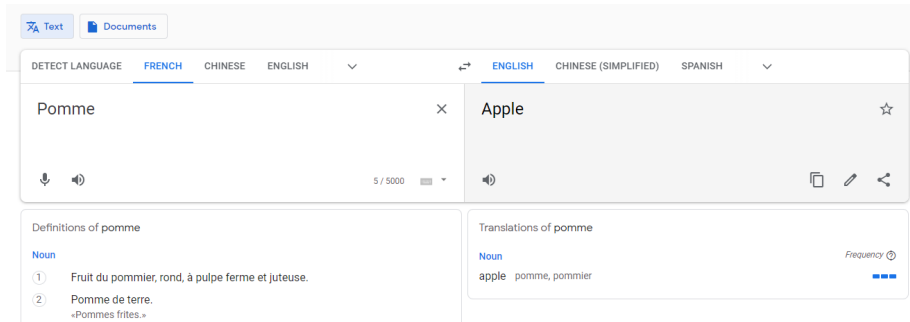


Figure 3: Google Translate[5] example

### 2.2.2 TaaS applications

Nowadays, many cloud service providers (e.g., Google Cloud, Microsoft Azure and Amazon AWS) provides customised cloud training services that enable users to train their models using the computational resource in the cloud. Some cloud service providers also provide pre-trained NLP models as web services. For instance, Google AutoML Translation[8] allows user to upload customised data and provide pre-trained model supports for accurate translations. Although these ML-based services save a lot of time and computational resource for users, they are still “expensive” in terms of financial and learning costs.

### 2.2.3 L2 reading & learning assistance applications

At the same time, many applications dedicated to provide L2 reading and learning assistance are developed for academic research and business purposes. E.g., a NLP-based reading assistance tool for nonnative English readers[9], this tool provides more comprehensive English reading assistance than TranslPrime using NLP technologies and server-side language supports.

On the other hand, many language educational organisations have developed numerous amount of language teaching applications for L2 learners in different countries, ages and literacy levels. E.g., Duolingo[10], a popular language learning application, provides different levels of online language lessons and dedicated reading comprehension training.

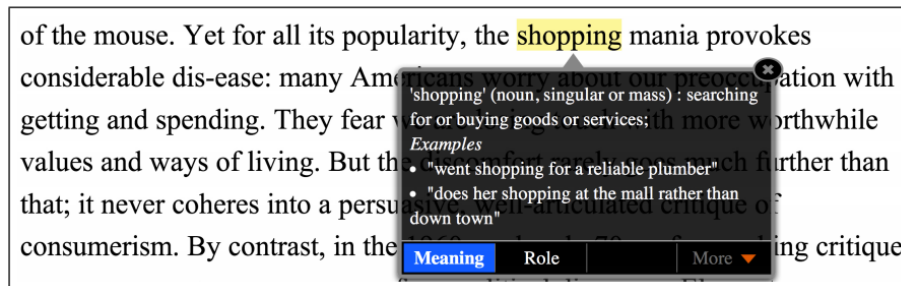


Figure 1: Looking up content word meaning

Figure 4: An example of English reading assistance tool [9]

#### 2.2.4 L2 proficiency assessments

Traditionally, the language proficiency assessments are designed by linguists and education experts. The language assessments usually adopt some kinds of linguistic guidelines and classify examinees into different proficiency levels. Among different kinds of assessments, reading competency assessment is usually carried out as one part of the assessment.

Recently, many educational organisations have designed online assessments for language proficiency. These assessments are convenient for L2 learners to self-assess their L2 proficiency and produce accurate self-estimations on their L2 reading competence.

### 3 Software Engineering

The context survey manifests that a desirable L2 reading assistance application for browser reading should prioritise following attributes:

- **Lightweight:** under the premise of providing complete and efficient L2 reading assistance functions, the application should have the following features:
  - The application should be kept small in volume and fast in speed, so that it can provide responsive L2 reading assistance services.
  - The application should avoid complicate computations and be as “independent” as possible from external dependencies, so that it does not heavily rely on external services which may interfere readers.
- **Usability:** the application should provide user-friendly GUIs and easy-to-use services, so that the cost of using and learning can be minimised.
- **Reliability:** the application should provide the following reliable services:
  - Reliable assistance service: the application should always provide L2 reading assistance services regardless the reading medium (i.e., whether selected texts come from webpage or files).
  - Reliable translation service: the application should provide accurate translations.

In this chapter, the software engineering process will be demonstrated in three sections. The requirement specification section documents all requirement engineering activities and modifications; The software engineering process section illustrates software development approaches and the development timeline; The ethics section documents all ethical considerations of this project.

#### 3.1 Requirements specification

##### 3.1.1 Requirement adjustments

- At the early stage, the application was designed to be “*a hybrid system composed of three interconnected subsystem with regards to the three aspects of L2 reading*”. The final implementation composed two subsystems (i.e., the translation module and the note module). The assessment subsystem was removed from the requirements and the reasons for cancellation will be discussed in **next subsection**.
- One of the initial requirements specified that the application should provide API substitution for dictionaries, so that readers are able to import and use other dictionary resource. The final implementation implemented this feature but the API substitution were required at the code implementation level. This requirement may be difficult for nonprogrammer users.

### 3.1.2 Assessment subsystem cancellation

The assessment subsystem was included in the initial requirement set, it was aimed to provide reading competence assessments and help readers to evaluate their L2 reading proficiency. However, this subsystem is cancelled due to the following constraints:

- Evaluation implementation: it is impossible to implement the language proficiency evaluation without prior-knowledge of applied linguistics. Although there are many unified language assessment standards such as CEFR [11], the implementation of L2 reading competence assessment are still already beyond the discipline of computer science.
- Internal resource limitation: although a NLP model can be built to analyse readers' behaviours through ML technologies, the computational resource required to build the model would be apparently unaffordable for only using the browser engine.
- External resource limitation: although many TaaS providers like Google AutoML Translation [8] have off-the-shelf NLP products that are capable of analysing reader data in short times, these services are too “expensive” in terms of financial and learning costs for this project.

As a result, related context survey for language proficiency assessment is preserved and presented in the **L2 proficiency assessment** section.

### 3.1.3 Functional requirements

All functional requirements are classified into three groups: the application framework requirements identifies fundamental architectural behaviours of the application; The translation requirements specifies the standards and desired functionalities for translating reading texts; The note-taking requirements specifies the functionalities related to providing note input interfaces and storing notes.

#### 1. Application framework

- The application should be capable of reading data from the browser and writing data to modify browser's behaviours.
- The application should be capable of using (and only using) browser's resource to implement the logic of L2 reading assistance.

#### 2. Translation functionality

- The application should allow user to select a group of texts from arbitrary webpages or files. It should also allow multiple types of selection strategies, such as dragging and copy-and-paste.
- Given a set of input strings  $T_{src}$  from the user, the application should be capable of providing a set of translated strings  $T_{tar}$  for:



- **Whole sentence translation** service (taking  $T_{src}$  as a whole sentence and producing  $T_{tar}$  that is equivalent in terms of semantic context)
- **Word lexical explanation** service (taking  $T_{src}$  as a set of words and producing  $T_{tar}$  with each  $t_{tar} \in T_{tar}$  is the lexical explanation of its corresponding  $t_{src} \in T_{src}$ )
- The application should use appropriate dictionary resource to translate selected texts.
- The application should allow the translation result to be displayed on browser’s GUI, so that the reader can see the L2 reading assistance along the side of original texts.

### 3. Note-taking functionality

- The application should allow user to input arbitrary strings as **notes** along the side of original texts or translated texts. These notes should be organised using object-oriented approach.
- The application should allow user to visualise, edit and delete their notes.

#### 3.1.4 Nonfunctional requirements

All nonfunctional requirements can be derived from aforementioned desired attributes of L2 reading assistance applications, including:

1. **Simplicity** for lightweight
  - The application should not rely on any individual translation API. Instead, it should provide service “sockets” and “plug” translation APIs, so that different APIs can be “plugged and unplugged” easily.
  - The application should provide concise L2 reading assistance information, rather than providing comprehensive but bulk information.
2. **Efficiency** for lightweight: the application should prioritise data processing speed in text extraction, API invocation and other kinds of component communications, so that the L2 reading assistance can be provided in the shortest time.
3. **Usability**: the application should provide clean and easy-to-use UI for user interactions. It should also provide user guidance within the application.
4. **Reliability**: the application should use appropriate dictionary resource to carry out translation works. The translation text should precisely express the meaning of the original texts.

5. **Security:** the application should read and write data within user's permission. It should use an independent running environment supported by the browser, so that user's data in this application cannot be accessed by potential malicious programs.

### 3.2 Software engineering process

This project is derived from the prototype “*An aid to learning and assessing to read foreign languages*”. An initial set of requirements were set out in DOER after some initial background investigations and supervisory guidance.

The complete software development process lasted for approximately 5 months (from January 2021 to May 2021) with 4 months of actual software development and 1 month of evaluation and documentation. An Agile-based method for individual development was used throughout the development:

- The weekly progress was reported to the supervisor through mail communication and online meeting.
- The priority of requirements were adjusted dynamically in accordance with the development progress. Some new requirements were added and some inadequate requirements were removed.
- An interim demonstration was arranged and a minimum viable product was delivered at the middle of development period.
- An user evaluation and a heuristic evaluation was conducted at the end of development. The detail of evaluations will be discussed in the evaluation chapter.

The development timeline Gantt chart is attached in the appendices.

### 3.3 Ethics

This project follows the ethics guidance from Student Handbook issued by the School of Computer Science. At the preparation stage, a self ethical assessment was conducted using the preliminary self-assessment form. After discussed with the project supervisor, it was decided that an anonymous user evaluation is required to evaluate the application, and the artefact evaluation form was sufficient to cover all necessary ethical considerations for this project.

The ethical application was firstly denied by the ethics convener due to the ambiguous description on user evaluation. After sending the clarification to the project supervisor and the ethics convener, the artefact evaluation form was approved. The copy of artefact evaluation form is attached in the appendices.

Due to the COVID-19 restrictions, the evaluation strictly followed the interim guidance for research provided by the University. The evaluation was conducted

online using Qualtrics and all participants joined the evaluation voluntarily and anonymously.

## 4 Design

In this chapter, the design decisions of TranslPrime will be demonstrated.

TranslPrime is designed to be a 2-layer web browser extension that acts as a “proxy” between the browser GUI and translation APIs. The base layer (layer 1) is the **browser extension framework** which is the application backbone. It establishes the connection between application and browser, giving application the permission to access various computational resource within the browser. The top layer (layer 2) consists of two components: the **translation module** handles all translation tasks by extracting selected texts, tokenising texts, invoking different translation APIs to translate tokens and displaying assistance menu on the browser GUI.; The **note module** handles all note-taking tasks such as reading input strings, constructing data structures and displaying the notes if users want to access notes.

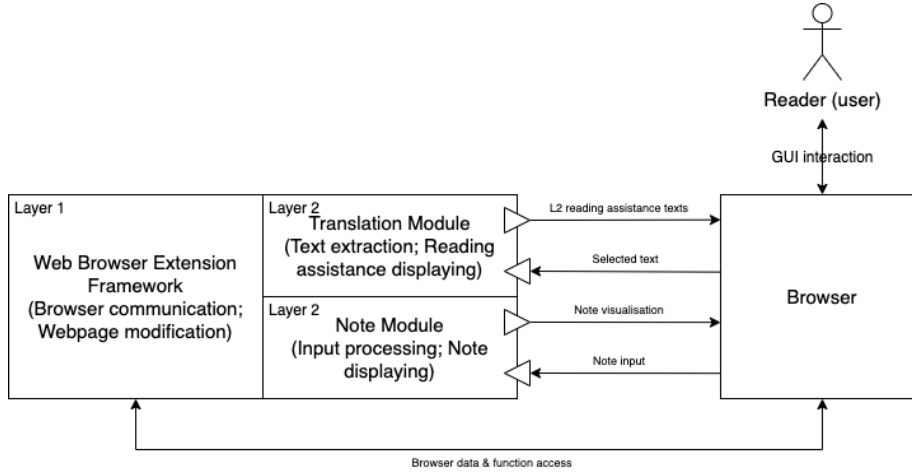


Figure 5: Software architecture of TranslPrime

### 4.1 Base layer: browser extension framework

#### 4.1.1 Browser extension & browser engine

A *browser extension* is a software module that provides additional features and functions to the web browser [12]. Besides its own set of functions, it is also capable of accessing and utilising the resource provided by the browser:

- A *browser engine* is the core component of a browser. It transforms the webpage from coarse files (i.e., HTML, CSS and JavaScript files) into interactive graphical representations. For instance, Blink [13] is the open-source browser engine for Chromium project which has been used to develop Google Chrome and Microsoft Edge.

- Once installed, the browser extension will be run in a separate sandboxed execution environment [12]. It is capable of accessing functions and resource from browser engine through **ExtensionAPI** which is a set of APIs designated for browser extension. Figure 6 displays a simplified architecture pattern for browser engine and browser extension.

Therefore, it is decided that the application will be designed and implemented as a browser extension rather than individual software or web application, because of the following reasons:

- Browser extension is inherently designed to be small and fast for adding functions on top of the original webpage. This feature typically fits the requirement of being a lightweight application.
- Compared to other applications, browser extension has the highest efficiency of assessing and processing data in the browser engine because it has a direct access through **ExtensionAPI**. Furthermore, browser extension can use ExtensionAPI to access web contents through constructing HTTP requests.

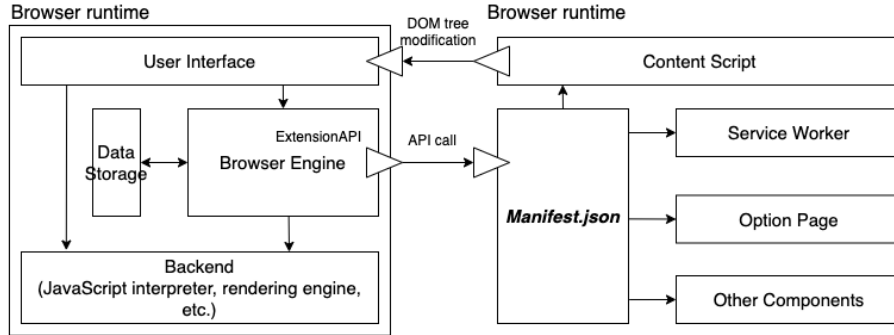


Figure 6: Browser engine & browser extension architecture

#### 4.1.2 Browser extension components

Despite the fact that there are many browser engines and their derivatives, the architecture patterns for browser extensions resemble almost the same. Generally, a browser extension consists of a collection of cohesive components that are implemented using web development technologies: HTML and CSS for constructing DOM tree, JavaScript for implementing background logics.

All browser extensions are required to include a manifest file **manifest.json** that describes the extension metadata such as resource access permissions and other included components. For TranslPrime, the following components are included:

1. *Service worker* (also known as *background script*): it is a script running at the background, acting as the “server-side” of extension. Its primary tasks include maintaining the application state and handling different kinds of events triggered by inter-component communications and network communications.
2. *Content script*: it is a script running in the context of opened webpages, acting as the “client-side” of extension. It is able to manipulate the Document-Object Model (DOM) of the webpage and thus providing additional functions through adding, removing or modifying HTML elements.
3. *Popup page & option page*: these pages allow the user to customise the settings and behaviours of the browser extension.

Figure 7 depicts the internal architecture of TranslPrime.

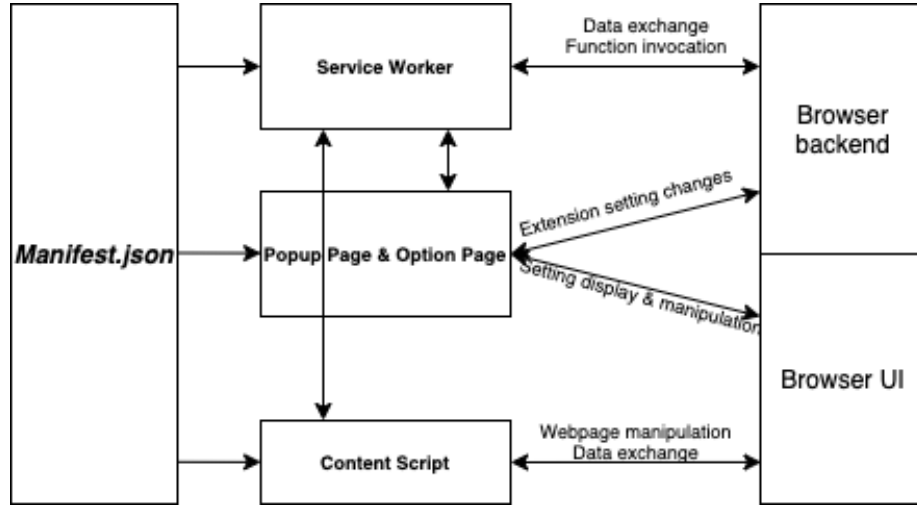


Figure 7: Internal architecture of TranslPrime

## 4.2 Top layer: translation module

The translation module is built for four L2 reading assistance tasks:

1. Text extraction: extracting selected texts selected by the user.
2. Text tokenisation: tokenising the texts and identifying the source language.
3. Text translation: invoking different translation APIs.
4. Text delivery: processing the API response and delivering the data package from the backend to the UI.
5. Text display: displaying translated texts on the assistance menu.

#### 4.2.1 Text tokenisation and translation rules

After extraction process, selected texts should be tokenised and stored into a dedicated data structure. In the tokenisation process, punctuations should be removed and a long sentence should be tokenised on word basis if needed. The following rules are set for tokenising a piece of text into the token data structure:

- If the user selects only one word for translation, the tokeniser should put this word directly into the data structure. The translation result should be an equivalent word in the target language.
- If the user selects a phrase or a sentence (consisting of at least two words), the tokeniser should treat them as an entity and put the string into the data structure without processing. The translation result should be an equivalent phrase or sentence in the target language.
- If the user selects a word, phrase or sentence for paraphrase, the tokeniser should tokenise texts into individual words and store all distinct words into the data structure. The translation result should be an object array with each object explaining the lexical meaning of its corresponding word.

By using the tokeniser, texts from different sources can be filtered and stored in a standard way in the service worker. Also, the tokeniser provides a general input interface for *API plugs*.

#### 4.2.2 API “plug-and-socket” solution

The solution for “API sockets and plugs” consists of three parts:

1. Using object-oriented approach on constructing translation APIs. Each translation API can be initialised as an independent object with the registration data stored as private attributes.
2. Providing *input adaptor* for each API object. The input adaptor can be implemented as an object method that takes the token data structure as input and provides an URL or multiple URLs that can be directly used for API calls.
3. Providing *output adaptor* for each API object. The output adaptor can be implemented also as an object method that takes the API response as input and provides standardised data objects.

Each API object can be considered as an individual *API plug* that connects the source language and the target language. The service worker should provide *API sockets* for different translation functions. Once a translation API is “plugged in”, its translation service should be immediately ready to use.

Hence, translation APIs can be standardised (as the data formats inside the service worker are standardised) and can be “plugged in and out” without relying on any services. Also, new translation API can be easily added by creating new *API plug* objects.

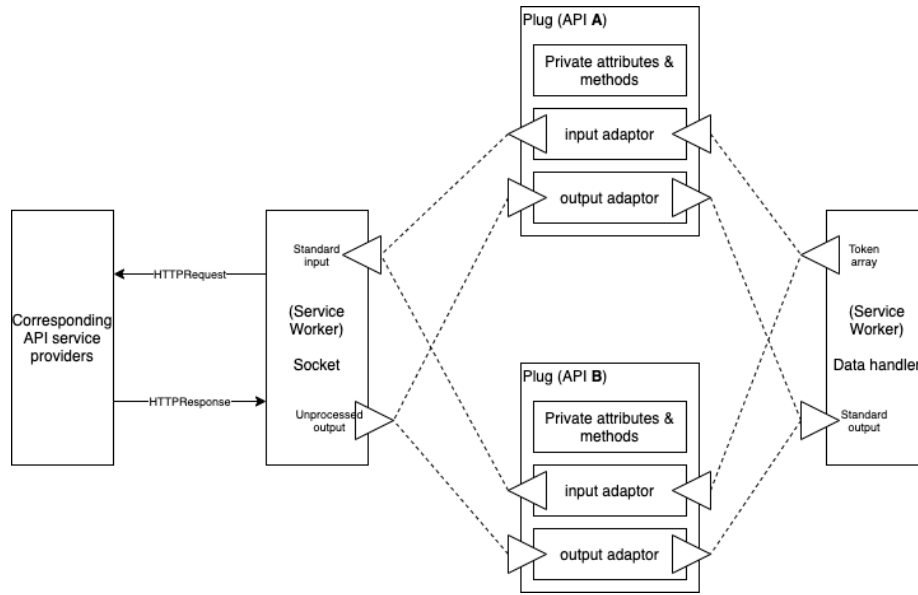


Figure 8: API “Plug-and-socket” approach

### 4.3 Top layer: note module

The note module is built for note-taking tasks:

- Note input recognition: reading and storing the user input together with other necessary information in a dedicated data structure.
- Note display: displaying existing notes if the keywords of a note match the translation result.
- Note modification: allowing users to visualise, modify and delete existing notes.

#### 4.3.1 Note structure & storage

The data structure should consist of the following attributes:

- Unique ID: the unique identifier that will be used as the index of note object.
- Note: the input string from the user.
- Note keyword(s): a word or multiple words that act as the search index for note display. During the translation process, if the keyword(s) of a note matches the selected texts, that note will be displayed together with the translation result.



A note taken by the user will be stored persistently in the local storage of the browser, these notes should have standard modification interfaces that can only be accessed by TranslPrime. Figure 9 depicts the architecture of note module.

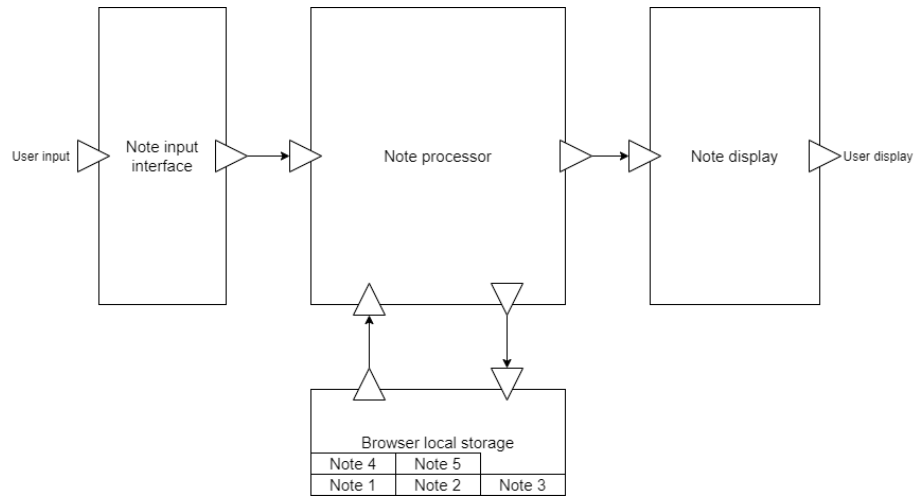


Figure 9: Note module architecture

## 5 Implementation

### 5.1 Browser extension

TranslPrime is implemented for browsers using Chromium[14] framework such as Google Chrome, Microsoft Edge and Opera. Therefore, TranslPrime can be directly installed on these browsers without considering adaptivity issues.

#### 5.1.1 Manifest access limitations & Content Security Policy (CSP)

Since browser extensions allowed to access resource and data stored in the browser engine, many data regulations and permission policies have been set out to prevent malicious behaviours. TranslPrime has implemented the following regulations:

- The access permissions on browser resource and data have been explicitly specified. In *manifest.json*, three access permissions required for the implementation are explicitly declared:
  1. **tabs**, access to *chrome.tabs* objects: each *chrome.tabs* object is a webpage tab opened in the browser. This permission is used for appending new HTML elements to the DOM tree of the current webpage.
  2. **storage**, access to *chrome.storage* objects: this permission is used for accessing the local storage of the browser for reading and writing global variables.
  3. **contextMenus**, access to *chrome.contextMenus* objects: this permission is used for injecting additional choices in browser's context menu (i.e., the right-clicking menu inside the browser).
- CSP[15], a policy that prevents Cross-Site Scripting (XSS) attacks, has been implemented: no external JavaScript libraries or plugins are imported in TranslPrime. The only external dependency used is Bootstrap 5.0[16] for decorating the CSS style for popup and option pages.

As an exchange, TranslPrime is implemented using only traditional web development technologies, including HTML, CSS/Bootstrap and vanilla JavaScript.

#### 5.1.2 Manifest V3

In November 2020, Google released the third edition of manifest standard for browser extension[17]. Rather than implementing Manifest V2 which is used by most of the browser extension developers, TranslPrime implements Manifest V3 with the following considerations:

- Manifest V3 uses *service worker* instead of *background script*. Compared to background script that persists in the background, the service worker is also a script but can be terminated if not used and restarted again.

It is also capable of acting as a network proxy between other extension components and translation APIs. In short, using service worker saves more time and browser resource compared to using background script.

- Manifest V3 allows the use of ***Promise*** which is a JavaScript object for handling asynchronous and deferred communications. Many ***Extension-API*** functions are also updated and become asynchronous functions. The use of asynchronous communication with proper handlers can boost the processing and reaction speed of the application.

### 5.1.3 Asynchronous event-based communication approach

Many ***ExtensionAPI*** functions are implemented using event-based approach with asynchronous communication pattern. As a result, all inter-component communications and network communications are implemented using asynchronous event-based communication approach:

- Service worker: once the extension is successfully installed, an initialisation event will be triggered. Extra options will be injected into browser's context menu and default global variables will be stored within browser's local storage. The events triggered by clicking context menu options are also handled by the service worker: for translation events, it will read the selected texts from the browser, request the corresponding translation API and deliver the processed result from API response to the content script; For a note-taking event, it simply notifies the content script to display the note input interface.
- Content script: there is only one event handler registered in the content script. It handles the message from the service worker consisting of translation results. It's primary is to construct the assistance menu by rearranging assistance texts and injecting new HTML elements into the DOM tree.
- Popup/Option page: the event handlers in these pages are only used for detecting setting changes and broadcasting these changes to other components by modifying global variables.

Due to the fact that JavaScript is a single thread language, the use of asynchronous communication can significantly boost the processing speed when paired with event-based approach. Therefore, many functions and message exchanges are implemented using asynchronous approach:

- Many functions require callback function as parameters. These callback functions will be invoked after some time-consuming functions are executed. E.g., reading a global variable from the browser storage, processing HTTP responses from translation APIs, etc.
- In service worker, ***fetchAll()*** function is a particular function implemented using many JavaScript asynchronous technologies. It is capable

of reading an array of URLs, constructing HTTP requests for each URL and processing HTTP responses using callback functions.

- In addition to callbacks, ***Promise*** objects are also used in order to handle asynchronous communications more efficiently.

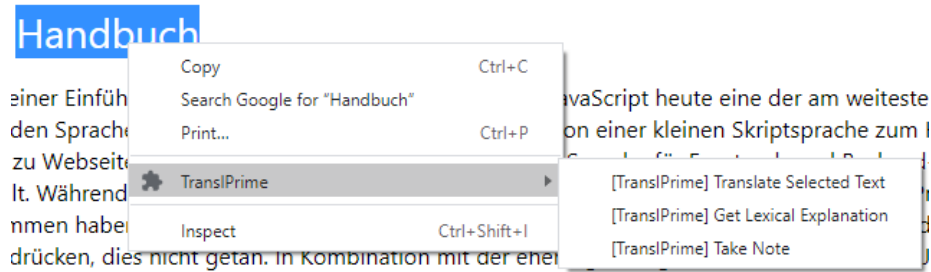


Figure 10: Extra context menu options, injected by the service worker

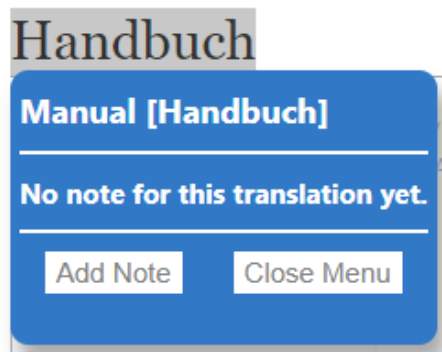


Figure 11: Reading assistance menu, injected by the content script

#### 5.1.4 Assistance menu construction & injection

The assistance menu is an additional *div* element displayed on top of the browser GUI (shown in figure 11). The menu is constructed using recursive injection approach:

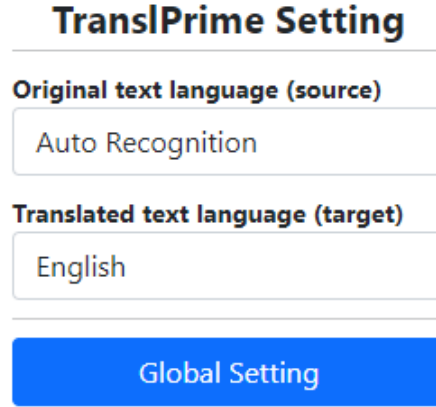
- In the content script, a class ***TranslMenuElement*** is implemented. It can be seen as the logical subclass of JavaScript ***HTMLElement***. Each ***TranslMenuElement*** instance is an HTML element appeared in the assistance menu.
- Since ***HTMLElements*** are arranged in tree structure, ***TranslMenuElements*** are also implemented using the tree data structure: each object consists of an array ***TranslMenuElement.#children*** that stores its children elements.

- All attributes of an **HTML***Element*, such as classes, styles and unique element ID, are also included in **TranslMenuElement** as object attributes.

The **TranslMenuElement** tree root will be stored in the content script. During the rendering process, the **TranslMenuElement** will be transformed to corresponding **HTML***Elements* and be injected into the DOM tree of the webpage.

## 5.2 Translation module

The translation module uses all extension components in the implementation: the selected texts  $T_{src}$  are translated to  $T_{tar}$  by the service worker; Then  $T_{tar}$  is processed by the content script and displayed on the assistance menu. The popup page and option page provides language selections that allow users to change the source language and the target language.



**TranslPrime Setting**

---

**Original text language (source)**

Auto Recognition

**Translated text language (target)**

English

---

Global Setting

Figure 12: Popup page that allows user to change source and target languages

### 5.2.1 Source & target language choices

Users can easily change the source language and the target language by selecting their desired languages in two ways:

- In the popup page (opened through clicking the “T” icon located in the browser’s address bar), the languages can be directly switched as figure 12 shown.
- In the option page (opened through either clicking the “Global Setting” button in the popup page or opening the option page from browser extension import interface), there are two dedicated options for switching languages.

The final implementation of languages only include English, German and French because of paraphrase API limitations. However, the source language supports an “auto recognition” option which ignores the source language and directly passes the selected text to the translation API. As a result, TranslPrime is able to recognise texts in different languages, but it is only capable of providing translations in English, French and German.

### 5.2.2 Text tokenisation

The text tokenisation is performed by a class in the content script called ***Tokeniser***. A new ***Tokeniser*** object will be constructed whenever a new piece of text is selected for translation, it will also be passed as parameters for *input adaptor* and *output adaptor* of the *API plug*.

### 5.2.3 API “plug-and-socket” implementation

For each translation API that is registered in the service worker, a corresponding ***TranslationAPI*** will be constructed. Each ***TranslationAPI*** object acts as a *API plug* that takes multiple parameters:

- The API name, HTTP domain and private information will be stored as object attributes. The private information are stored as a private attribute which cannot be accessed by other scripts.
- The *input adaptor* and *output adaptor* are implemented as two callback functions stored in the object. ***TranslationAPI.#converter*** is the *input adaptor* that takes itself and a ***Tokeniser*** object and generates URLs for further HTTP requests; ***TranslationAPI.#handler*** is the *output adaptor* that takes the HTTP responses and the original ***Tokeniser*** object and generates standardised output for content script.

On the other hand, the *API sockets* are implemented as service worker variables called ***translateAPI*** and ***paraphraseAPI***. Different ***TranslationAPI*** object can be “plugged in and out” through simple variable assignments.

### 5.2.4 Assistance menu: translation display

For whole sentence translation (i.e., selecting texts and choosing “Translate Selected Text”), both translated texts and original texts will be displayed on the assistance menu. The original texts will be included inside a square bracket, as shown in figure 13.

For word lexical explanation (i.e., selecting texts and choosing “Get Lexical Explanation”), the original texts, the translated texts and their lexical explanations will be displayed in a list order, as shown in figure 14.

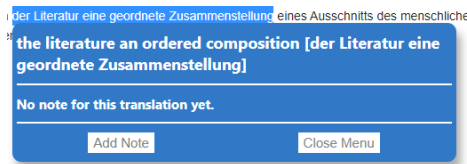


Figure 13: Assistance menu for “Translate Selected Text”

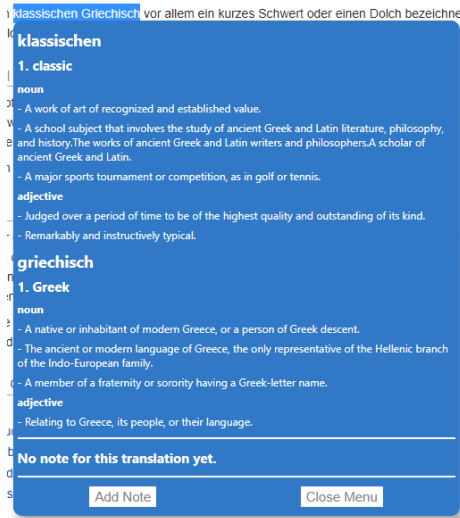


Figure 14: Assistance menu for “Get Lexical Explanation”

### 5.2.5 Default APIs & API plug interface

Due to the fact that most of the translation APIs are charged services, two free translation APIs are used as the default configuration.

The default *translateAPI* uses the Baidu translation open platform<sup>[18]</sup> and the *API plug* is implemented as *BAIDUTranslateAPI*. Although it provide free translation service , there is an access limitation that the free service only accepts one translation request per second. As a drawback, this limitation will significantly increase the processing time when user makes multiple translation requests in a short period.

The default *paraphraseAPI* uses the Free Dictionary API<sup>[19]</sup> that provides free paraphrase services, the *API plug* is implemented as *FreeDictionaryAPI*. As one of its drawbacks, it only provides paraphrases for limited number of languages, this is also the major reason why TranslPrime only supports three target languages: although some languages such as Chinese are supported for the translation part, the paraphrase part does not support them. Rather than

providing a lot of language choices and leaving the paraphrases as “unavailable services”, the approach of providing complete services is used by TranslPrime.

There is also an obsolete translation API ***MSTranslatorAPI*** that uses Microsoft Azure Translator V3.0 service[20]. Due to the fact that Microsoft only provides limited access to their translation service and the free service trials have been used up in the development process. This *API plug* is only used for the demonstration purpose of *API plug*. The API can be substituted in the option page, but no translation result will be delivered.

### 5.3 Note module

The note module uses content script as the note input interface and option page as the note display. The note data structure is implemented with the following attributes:

- ID: the uniqueness of identifier is implemented using the UNIX timestamp when the note is taken.
- Note: the note input is stored using the JavaScript template string approach, so that special characters such as Chinese characters can be stored. Also, the note is capable of handling multiple lines of input.
- Keyword: the keyword is sequence of word(s) separated by commas, the keyword will be used as the search index when the assistance menu is called. If the translation result contains the keyword, the associated note will be displayed in the note display section. The matching of keyword uses
- Category: in addition to the keyword, TranslPrime also supports categorising notes using customised categories. In the option page, user can also search the notes using categories.

The note assistance menu can be called with or without text selection. The user can either directly right-clicking the empty space of webpage or selecting a piece of text as keyword(s).



### 5.3.1 Note display

In the option page, the notes in the storage can be visualised by user. The option page also allows filters notes based on their categories or keywords. The user is also capable of selecting their notes and editing the attributes of notes in the option page.

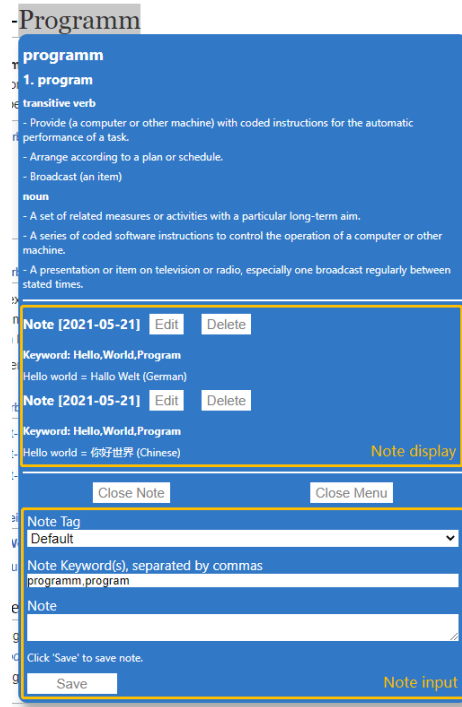


Figure 15: Note input and display example

Note (keyword(s): hello world)
[2021-05-21] [Hello,World,Program] Hello world = Hallo Welt (German)
[2021-05-21] [Hello,World,Program] Hello world = 你好世界 (Chinese)

Figure 16: Note search result in option page

## 5.4 User assistance

TranslPrime also provide user guidance in the option page. Once the application is successfully installed, the option page will be automatically opened and user can open the guidance for translation module and note module and learn how to use TranslPrime.

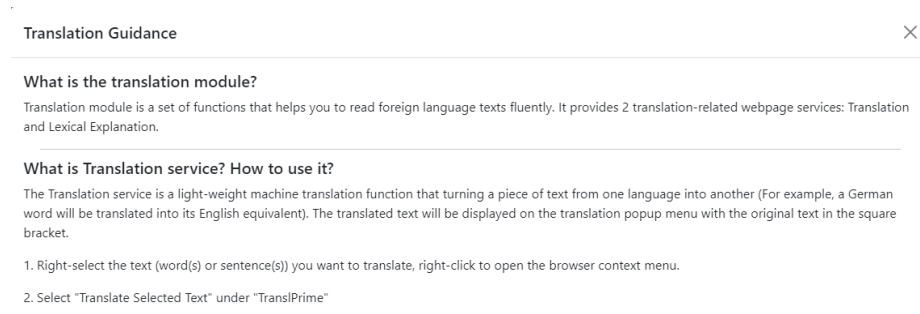


Figure 17: User guidance example

## 6 Evaluation & Critical Appraisal

At the end of application development, two separate evaluations were carried out to evaluate the quality and usability of TranslPrime:

1. A public user evaluation was conducted among some university L2 readers and learners. They were invited online to read a short article in their respective L2s with the assistance of TranslPrime using different browsers. The evaluation feedback are reported in the form of online questionnaires. The feedback results generated by Qualtrics are attached in the final submission extras. A more visualisable summarised evaluation result is attached in the appendices.
2. Since TranslPrime involves UI interactions, Nielsen’s heuristic approach [21] conducted to self-inspect the usability of TranslPrime. In this evaluation, the usability and the performance of TranslPrime are evaluated against 10 usability heuristics.

### 6.1 User evaluation result

At the end of evaluation, there are a total of 11 participants and 5 complete questionnaires are collected, 6 participants left the evaluation unfinished with partially finished questionnaires.

The evaluation questionnaire contains 3 type of questions: single/multiple choices, scoring (the score is ranged from 0 to 5, with 0 being the worst and 5 being the best) and arbitrary input for feedback and suggestions.

From the evaluation result, it can be observed that:

- Most of the participants uses L2 as a learning tool for academic purposes (due to the fact that all participants are university students).
- Translation module: the participants are overall satisfied with the whole sentence translation function. The mean scores of translation speed, accuracy and helpfulness are 4.1, 4.0 and 4.4 over 5 respectively.
- Translation module: the participants are overall satisfied with the accuracy and helpfulness of paraphrase (word lexical explanation) (with the means of 4.4 and 4.2 respectively). However, the paraphrase speed only has the mean of 3.6 with the minimal value of 2.0. This indicates that the speed for providing paraphrases still need to be improved.
- Translation module: most participants think the translation module is usable but still can be improved in terms of improving the ways of calling out assistance menu and the clarity of assistance menu UI.
- Note module: the participants are overall satisfied with the note module (mean: 4.0). Most participants think that it is convenient despite the fact

some participants give the minimal score of 1.0, indicating that the note input interface still has a lot to space to be improved.

- Overall evaluation: overall, 80% of participants think TranslPrime has greatly improved their L2 reading experience through the assistance from the translation and note modules. The mean scores for overall usability (the evaluation on whether TranslPrime is user-friendly and convenient), efficiency (the evaluation on whether TranslPrime provides fast responses) and linguistic efficiency (the evaluation on the default translation and paraphrase services) are **3.8**, **4.2** and **4.0** respectively.

## 6.2 Heuristic evaluation

The heuristic evaluation is a self-inspection on the usability of TranslPrime. The evaluation is performed by critically compare the usability properties of TranslPrime against 10 heuristics from Nielsen.

1. *Visibility of system status (the system should always keep users informed about what is going on)*: all TranslPrime functions are provided via the assistance menu injected in the webpage. Since TranslPrime uses asynchronous event-based approach, the system feedback can be seen by user within reasonable time. However, there are some rare cases where the assistance menu cannot be displayed due to DOM tree injection failure.
2. *Match between system and the real world (the system should speak the users' language, with words, phrases and concepts familiar to the user)*: In terms of browser context menu, the injected options are labelled explicitly and the option description text directly describe the translation functions; In terms of assistance menu, all visible texts in the assistance menu, the popup and the option page are presented using plain English.
3. *User control and freedom (users often choose system functions by mistake and will need a clearly marked "emergence exit" to leave the unwanted state)*: The assistance menu has the "Close menu" button that removes the assistance menu. In the case of misclicking, the menu can be easily closed by clicking the button.
4. *Consistency and standards (users should not have wonder whether different words, situations or actions mean the same thing)*: TranslPrime uses consistent terminologies for its functionalities: "Translate Selected Text", "Get lexical explanation" and "Take note".
5. *Error prevention (either eliminate error-prone conditions or check for them and present users with a confirmation option)*: there is a dedicated error handling functions **`printErrors()`** in the service worker. It is capable of handling the edge and invalid cases in the data processing by logging the errors in browser's console; For the option page, invalid operations (e.g., using keyword searching without any keywords or attempting

to remove “Default” category which by default cannot be removed) will be alerted by the browser.

6. *Recognition rather than recall (minimise the user’s memory load by making objects, actions, and options visible)*: in terms of assistance menu, the note-taking button and closing button are explicitly displayed; In terms of option page, all functions are presented using explicit HTML elements, such as buttons, input blocks and selections.
7. *Flexibility and efficiency of use (the system can cater both inexperienced and experienced users)*: for inexperienced users, TranslPrime provides a user manual for installation and built-in user guidance in the option page,
8. *Aesthetic and minimalist design (dialogues should not contain information which is irrelevant or rarely needed)*: TranslPrime only displays one assistance menu on the webpage. The default assistance menu uses blue background and white fonts in order to keep the menu noticeable.
9. *Help user to recognise, diagnose, and recover from errors (error messages should be expressed in plain language)*: the error message alerted in the option page uses plain English to describe the errors.
10. *Help and documentation (provide help and documentation)*: both external documentations (the user manual) and internal built-in helps (user guidance in the option page) are provided to help users.

In conclusion, for each heuristics there is at least one solution for usability problems. Although there are some messages that will not be directly shown to the users through assistance menu and options page, TranslPrime is still capable of providing UI with usability guarantees.

### 6.3 Known design and implementation defects

- Browser extension framework: if the webpage is already overridden by some kinds of web applications with customised context menu, TranslPrime will not be able to inject assistance menu into the web application; This could be fixed by detecting the browser environment or finding other ways to provide translation service (e.g., adding a “search bar” in the popup so that the text can always be searched).
- Translation module: the paraphrase function can be extremely slow if a long piece of text is selected because of the API limitations. The default paraphrase API does not have direct translation service. In order to get translation, it needs to translate the tokens into the target language and then perform the paraphrase task. This could be fixed by using more advanced paraphrase APIs.
- Translation module: the Microsoft Translator V3.0 API cannot be used due to the trials reason (demonstrated in **Default APIs & API plug**

**interface**) section. This could be fixed by using an activated Microsoft Azure account with new application IDs and keys.

## 7 Conclusion

Overall, TranslPrime have the ability to resolve the cognitive interrupts appeared in the L2 reading process. It is capable of providing translation service and note service to help L2 readers improving their L2 reading competence. Although many functionalities and supports can be improved in advance, TranslPrime has met the project expectation of providing all primary objectives and some secondary objectives. (Just as one of the user feedback states: *Overall, it is a functional translation extension, although far from those top-notch technology company products, it can do its job.*)

The future work for this research project includes but not limited to:

1. Using finer translation APIs to provide more accurate and comprehensive translate services. The NLP-based reading assistance project [9] can be a very good guidance.
2. Offering more aesthetic and user-friendly UI for assistance menu.
3. Developing more L2 reading competence assistance functions.

From this research project, I have gained comprehensive understanding of L2 reading competence and L2 reading assistance from both applied linguistics and computer science perspectives; I have also learnt how to develop high-quality browser extension that can modify the browser's behaviour to provide practical functions for application users.

## References

- [1] Grabe, W. (2008). *Reading in a Second Language: Moving from Theory to Practice* (Cambridge Applied Linguistics). Cambridge: Cambridge University Press. doi:10.1017/CBO9781139150484
- [2] Koda, K. (2005). *Insights into Second Language Reading: A Cross-Linguistic Approach* (Cambridge Applied Linguistics). Cambridge: Cambridge University Press. doi:10.1017/CBO9781139524841
- [3] Krashen, S. (1982). *Principles and practice in second language acquisition*
- [4] Olmez, F. (2016). *Exploring the interaction of L2 reading comprehension with text- and learner-related factors*
- [5] Google Translate. <https://translate.google.com>
- [6] Cambridge Dictionary. <https://dictionary.cambridge.org/>
- [7] Wiktionary. <https://www.wiktionary.org/>
- [8] Cloud Translation. <https://cloud.google.com/translate>
- [9] Azab, M., Salama, A., Oflazer, K., Shima, H., Araki, J., & Mitamura, T. (2013, September). *An NLP-based reading tool for aiding non-native English readers*. In Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013 (pp. 41-48).
- [10] Stories - Duolingo. <https://en.duolingo.com/stories>
- [11] Common European Framework of Reference for Languages: Learning, Teaching, Assessment (CEFR). <https://www.coe.int/en/web/common-european-framework-reference-languages>
- [12] What are extensions?. <https://developer.chrome.com/docs/extensions/mv3/overview/>
- [13] third\_party/blink. [https://chromium.googlesource.com/chromium/src/+refs/heads/main/third\\_party/blink](https://chromium.googlesource.com/chromium/src/+refs/heads/main/third_party/blink)
- [14] The Chromium Projects. <https://www.chromium.org/>
- [15] Content Security Policy. <https://developers.google.com/web/fundamentals/security/csp>
- [16] Bootstrap. <https://getbootstrap.com/>
- [17] Welcome to Manifest V3. Chrome Developers. <https://developer.chrome.com/docs/extensions/mv3/intro/>
- [18] Baidu Translation Open Platform, <https://api.fanyi.baidu.com/>



- [19] *Free Dictionary API*, <https://dictionaryapi.dev/>
- [20] *Microsoft Translator service - Azure Cognitive Services*, <https://docs.microsoft.com/en-us/azure/cognitive-services/translator/reference/v3-0-reference>
- [21] Nielsen, J., & Molich, R. (1990, March). *Heuristic evaluation of user interfaces*. In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 249-256).

## **Appendix**

1. User manual (8 pages)
2. User evaluation result (4 pages)
3. DOER (4 pages)
4. Software development timeline (1 page)
5. Artefact evaluation form (1 page)

# TranslPrime: User Manual

## Installation

TranslPrime is a web browser extension based on Chromium browsers. It is not published to any kinds of web extension markets due to university policy restrictions.

To install TranslPrime in Google Chrome:

1. Open the **Extension** management page by either clicking the extension icon at the right-top corner of the page (in the address bar), or clicking **More tools** -> **Extension** in the setting menu.
2. Enable **Developer mode** at the right-top corner of the page.
3. Import all the files in *translator* directory into the browser.
4. Once the installation is ready, the option page will be displayed.

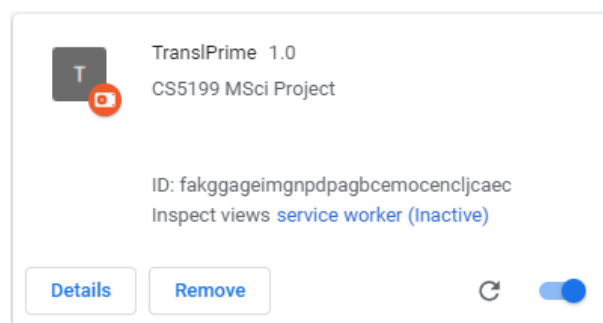


Figure 1. Chrome extension

To install TranslPrime in Microsoft Edge:

1. Open the **Extension** management page by clicking **Extension** in the setting menu.
2. Enable **Developer mode** at the left-bottom corner of the page.
3. Import all the files in *translator* directory into the browser.
4. Once the installation is ready, the option page will be displayed.

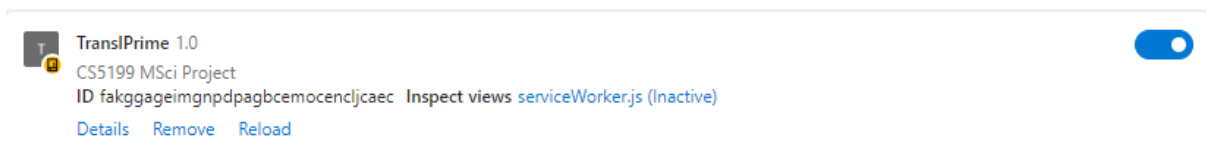


Figure 2. MS Edge extension



Figure 3. The extension icon for TranslPrime

## Translation Module

The translation module of TranslPrime is capable of providing whole sentence translations and word lexical explanations.

### Whole sentence translation

This function can translate the text from the source language into the target language:

1. Right-dragging to select the text that will be translated.
2. Right-clicking to select **[TranslPrime] -> [TranslPrime] Translate Selected Text**.
3. The translated texts will be displayed on the assistance menu.

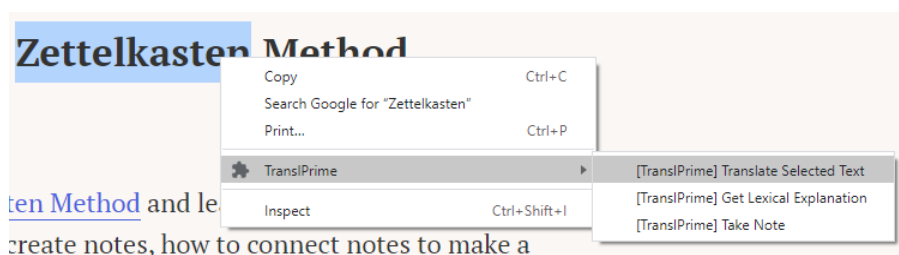


Figure 4. Context menu selection

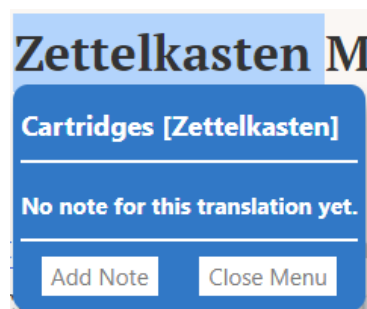


Figure 5. Single word translation (German -> English)

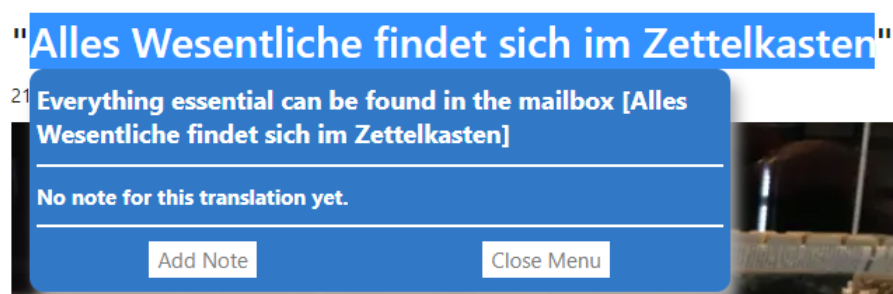


Figure 6. Sentence translation (German -> English)

## Word lexical explanation

This function can tokenise the selected string and provide lexical explanation for each word.

1. Right-dragging to select the text that will be paraphrased.
2. Right-clicking to select **[TranslPrime] -> [TranslPrime] Get Lexical Explanation**.
3. The paraphrased texts will be displayed on the assistance menu.

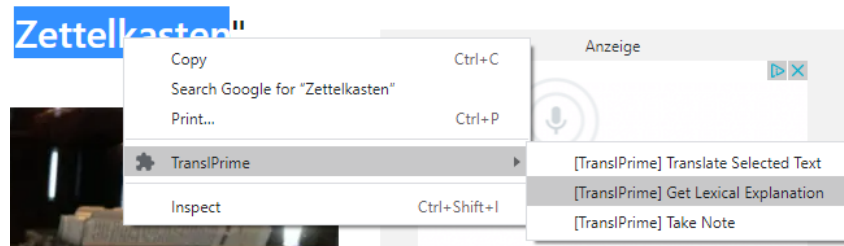


Figure 7. Context menu selection

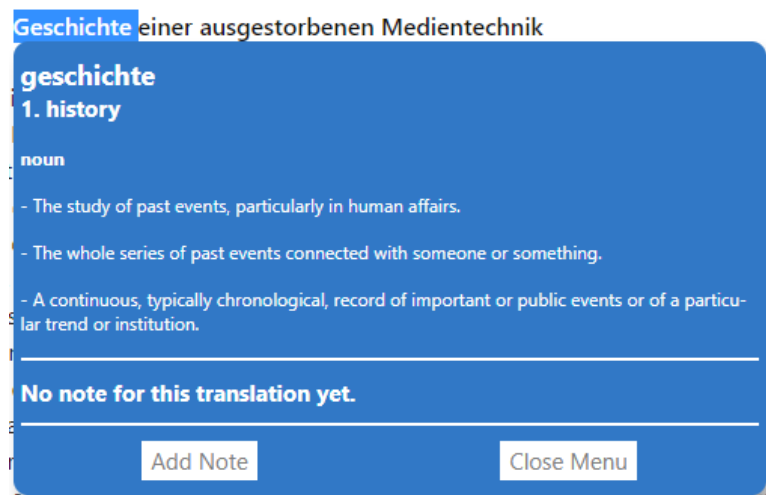


Figure 8. Single word paraphrase (German -> English)

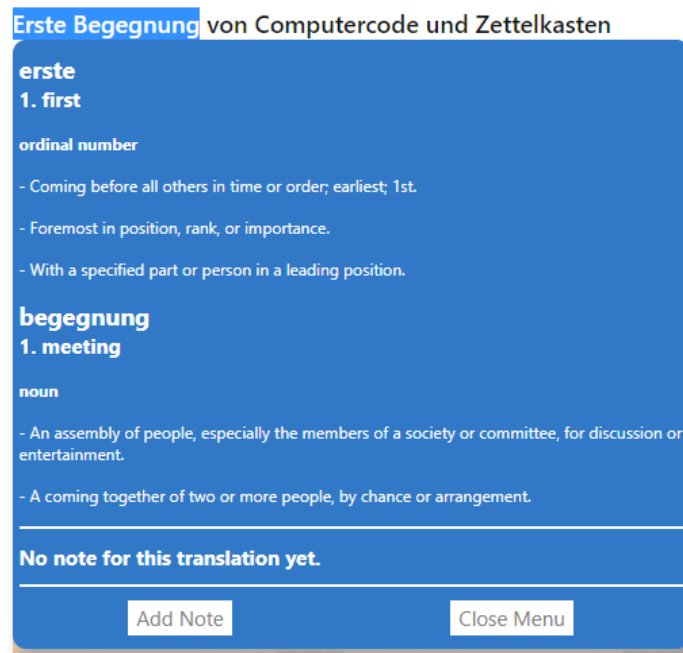


Figure 9. Multiple words paraphrase (German -> English)

The source language and the target language either in the popup page by clicking the **T** icon in the address bar, or in the option page.

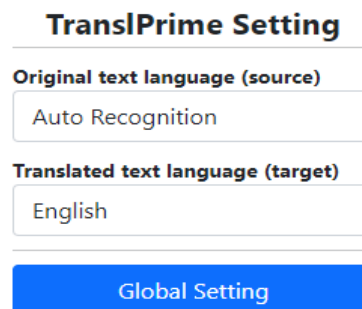


Figure 10. Popup page

For more information, a user guidance for translation module is provided in the option page that can be opened in the option page.

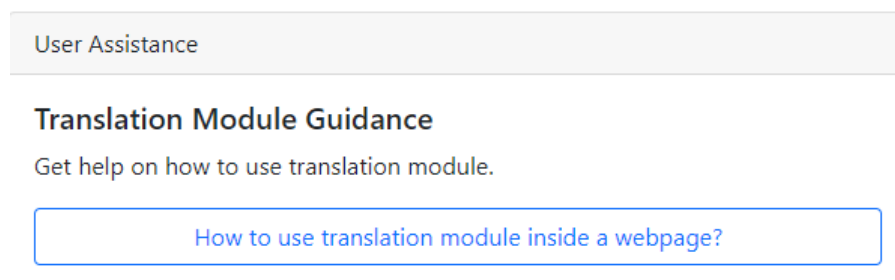


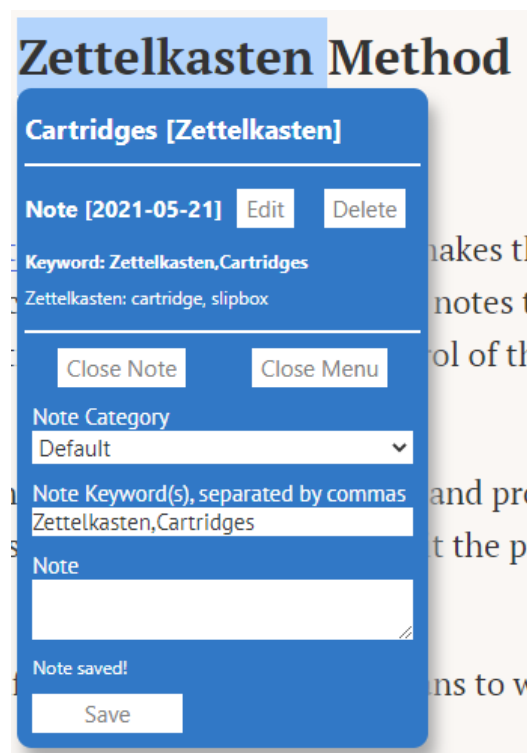
Figure 11. Translation module guidance

## Note Module

The note module of TranslPrime is capable of providing note-taking functions through the assistance menu. A note has a category, optional keyword(s) that are separated by commas and the note body.

### Take note with assistance menu

1. Clicking the **Add Note** button, the note input interface will be automatically expanded.
2. Clicking **Save Note** to save the note.



The screenshot shows a blue modal window titled "Cartridges [Zettelkasten]" overlaid on a document titled "Zettelkasten Method". The modal contains the following elements:

- A header bar with the title "Cartridges [Zettelkasten]" and a close button.
- A section for "Note [2021-05-21]" with "Edit" and "Delete" buttons.
- A "Keyword: Zettelkasten,Cartridges" field with a sub-field "Zettelkasten: cartridge, slipbox".
- Buttons for "Close Note" and "Close Menu".
- A "Note Category" dropdown menu set to "Default".
- A "Note Keyword(s), separated by commas" field containing "Zettelkasten,Cartridges".
- A "Note" text area.
- A "Note saved!" status message and a "Save" button.

Figure 12. Taking note within assistance menu

### Take note without assistance menu

There are two approaches to take note without translating anything first

1. Right-dragging to select the text and right-clicking to select **[TranslPrime] -> [TranslPrime] Take Note**
2. Right-clicking any empty space on the web page and selecting **[TranslPrime] -> [TranslPrime] Take Note**

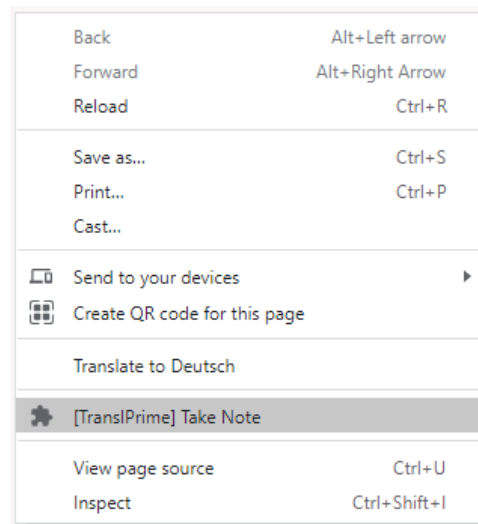


Figure 13. Arbitrary note-taking

### Note access, modification and deletion

Once stored, the note can be accessed, edited and deleted in both the assistance menu (as shown in figure 12) and the option page. The option page also provides additional note-taking functions:

1. Searching and visualising notes: existing notes can be displayed by their categories or keyword(s).
2. Modifying note categories: additional note categories can be added and removed.

Figure 14. Note search and note-taking related functions

For more information, a user guidance for note module is provided in the option page that can be opened in the option page.



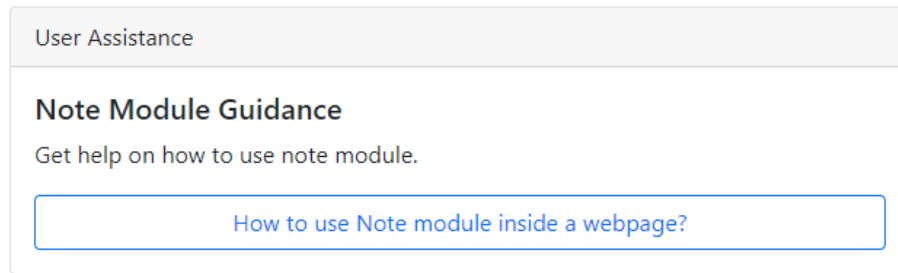


Figure 15. Note module guidance

## Translation API substitution

TranslPrime uses an “API socket and plug” approach that allows the translation service API to be freely substituted. As a demonstration, there is a *MSTranslatorAPI* variable template in *serviceWorker.js*.

### Construct and use new API plug

1. Make a new *TranslationAPI* object with the following parameters:
  - a. A boolean variable specifying whether it is for translation service or paraphrase service.
  - b. A string specifying its name displayed in the option page.
  - c. A string specifying API domain in the form of *http(s)://.....*
  - d. A “input adaptor” that takes this API and a tokeniser consisting of selected text information and generates *HTTPRequest* related data objects.
2. Substitute the value of *translateAPI* socket or *paraphraseAPI* socket by variable assignments.

The API and object attributes are explicitly commented in the code and also documented in the README file.

State	Participant L1	Purpose of L2 reading	Reading assistance tools used	L2 Article chosen	Whole sentence translation evaluation	Lexical explanation evaluation
Finished	English	Entertaining	Online translation Grammar correction L2 education	Chinese	Speed: 4/5 Accuracy: 4/5 Helpfulness: 5/5	Speed: 2/5 Accuracy: 5/5 Helpfulness: 3/5
Finished	English	Academic	Online translation Grammar correction	Chinese	Speed: 4/5 Accuracy: 5/5 Helpfulness: 4/5	Speed: 2/5 Accuracy: 5/5 Helpfulness: 4/5
Finished	Chinese	Academic	Online translation Grammar correction L2 education	German	Speed: 4.5/5 Accuracy: 3.5/5 Helpfulness: 4/5	Speed: 5/5 Accuracy: 4/5 Helpfulness: 4/5
Finished	Chinese	Academic	Online translation Grammar correction L2 education	English	Speed: 5/5 Accuracy: 4/5 Helpfulness: 5/5	Speed: 5/5 Accuracy: 4/5 Helpfulness: 5/5
Finished	Chinese	Academic	Online translation Grammar correction	German	Speed: 3/5 Accuracy: 3.5/5 Helpfulness: 4/5	Speed: 4/5 Accuracy: 4/5 Helpfulness: 5/5
Unfinished	Chinese	Academic	Online translation	(No answers are provided for questions after this question)		
Unfinished	(No answers are provided for questions after this question)					
Unfinished	(No answers are provided for questions after this question)					
Unfinished	(No answers are provided for questions after this question)					
Unfinished	(No answers are provided for questions after this question)					
Unfinished	(No answers are provided for questions after this question)					

[illegible]

[illegible]

Suggestions
<p>Awesome work!</p> <p>Though its not user-friendly (for me), it still helps me with some translation.</p>
/
<p>Same as before (feedback on translation)</p> <p>翻译插件很棒，平时阅读L2文章时可以提供快捷且相对准确的翻译，能够提供很大的帮助。 (The extension is convenient, when I read a L2 article it is able to provide quick and relatively accurate translations, it can provide a lot of helps)</p>
/

# CS5199 DOER

Student ID: 160000139

Student name: Yuxuan Wang

## Description

### Project title

An aid to learning and assessing to read foreign languages

### Project description

Reading is a fundamental skill to acquire information from all kinds of text resources. For second language (L2) readers and learners, reading is also an inevitable challenge of gaining access to more knowledge and becoming skilled L2 users. Therefore, developing reading ability has become an essential requirement of L2 acquisition.

In the case of linguistics, reading is commonly understood as a multifaceted process to **comprehend** the meaning of the text using readers' morphological, syntactic and semantic knowledge<sup>[ref]</sup>. Reading is also an implicit process of **learning** and **evaluation**, as readers not only evaluate the texts and their authors by asking themselves "*do I still have an interest in the texts?*" or "*do I agree with the authors?*", but also evaluate their strengths and weaknesses in their reading abilities by asking themselves "*how well can I understand what the authors say?*" or "*what have I learnt from the texts?*". Different individuals have diverse educational backgrounds, they also have their reading habits, interests and purposes so that they tend to find and read texts they are interested in.

On the other hand, all readers, even skilled readers, will encounter different levels of reading barriers which distract their attention and reduce their reading efficiency. These reading barriers are much more intractable for L2 readers and learners to address: L2 readers with different native languages (L1) need to understand how their L1s are different from the L2 they are reading. These L1-L2 variations are going to inevitably affect the efficiency and the motivation of L2 reading.

As a result, the development of aids in reading and L2 reading is always a popular topic on language learning and computer science. Nowadays, the prosperity of the Internet and the machine translation technology has significantly changed how people read and develop their reading ability by providing countless linguistic and educational supports in diverse forms. These supports can be classified into three categories concerning the aspects of reading to address the L2 reading barriers and improve reading efficiency:

1. Material support: in the past, people needed to spend large amounts of time in the library or bookstore to search for their interested materials. Today, the Internet and its byproducts, including various search engines, websites and platforms, have provided abundant amounts of multilingual resources that can meet the readability requirements for different readers. In addition to resource support, many language learning software, like Spreeder<sup>[ref]</sup>, also provide dedicated reading ability training.

2. Language assessment support: the majority of reading ability assessments are normally carried out as a part of comprehensive offline language assessments. Recently, some educational organisations have started to provide online computer-based language assessments, such as IELTS<sup>[ref]</sup> and TOEFL<sup>[ref]</sup>. Moreover, many language learning software providers, like Duolingo<sup>[ref]</sup>, start to integrate individual language ability assessments into their applications. These assessments are convenient for L2 readers to self-assess and produce accurate self-estimations on their L2 reading ability at different scales.
3. Translation support: advanced machine translation technologies have successfully transformed traditional bulky dictionaries into light-weighted, user-friendly programs. Not only can these ever-evolving programs provide fast and precise multilingual translations on vocabularies, but also sentences and even paragraphs.

Each of these supports meets only a single demand and is lacking integration, L2 readers need to spend additional time opening up multiple individual applications, learning how to use them separately and combining different applications to satisfy their needs.

Consequently, hybrid translation applications like Google Translate<sup>[ref]</sup> and Dictionary Everywhere<sup>[ref]</sup> are used more frequently today due to their integrated and user-friendly nature, although they purely focus on the translation works and lack assistance on L2 reading ability assessments.

Therefore, an integrated aiding tool for improving L2 reading efficiency and providing L2 reading ability assessments can be developed. This tool is expected to reinforce users' reading ability with regards to the three aspects of L2 reading. In the case of design, it is expected to be implemented as a user-friendly reliable hybrid translation program with additional reading ability evaluation functionalities.

## Objectives

This project is an extension on “An aid to learning to read foreign languages”<sup>[ref]</sup>, it aims at developing an integrated language translation and assessment aiding tool targeted at L2 readers to improve their L2 reading ability.

The final software artefact is expected to be a hybrid system composed of three interconnected subsystems with regards to the three aspects of L2 reading:

1. Text extraction and translation (translation system): this subsystem should be able to read the selected text, in the source language, from arbitrary text resources and translate by providing linguistic information in the target language.
2. Note attachment (note system): this subsystem should allow the readers to attach reading notes in parallel with the translation result.
3. Reading ability assessment (assessment system): this subsystem should provide reading ability estimations in accordance with readers' translation history and recorded notes.

The final software artefact is expected to be implemented as a web browser extension program using HTML, CSS, JavaScript/Node.js and other Internet-related technologies.

### Primary objectives



[Translation system] The application shall extract the text selected by the user from arbitrary webpages.

- It shall allow multiple types of selection strategy, e.g. right-drag, copy-and-paste, etc.

[Translation system] The application shall translate input text into corresponding output text.

- It shall display the translated text to the user through small pop-ups or dialogue frames.
- It shall allow users to specify the context of input and output languages, and use appropriate dictionary resources in the translation process.
- *Usability: it should be able to search the text in the dictionary in a short time.*
- *Reliability: it should be able to deliver precise translation results to the users.*

[Note system] The application shall provide an additional input field in parallel with the translated information for note-taking purposes.

- It shall store the *<originText, translateText, noteText>* (note pair) permanently as file or database entries so that the *noteText* can be displayed again when users select *originText* or *translateText* multiple times.
- *Usability: it should allow users to write notes easily and quickly.*

[Assessment system] The application shall be able to evaluate users' reading ability using existing note pairs data and translation histories.

- For a particular user, the reading ability evaluation shall take all his/her note pairs and searching histories, evaluate the relative language level according to linguistic theories and display the evaluation result in terms of textual or numeral reports.
- *Reliability: the evaluation should provide authentic and accurate results to reflect users' real reading ability levels.*

[Software] The application shall support all browsers based on the Chromium browser framework<sup>[ref]</sup> (e.g. Google Chrome, Microsoft Edge, Opera, etc.)

- The browser extension shall be imported into the browser without reporting errors.
- *Reliability/Security: the application shall not expose any personal information of the readers*

[User evaluation] The application shall be evaluated by L2 readers and proved to be useful in helping improve L2 reading efficiency.

- The participants will be invited to read an L2 article with the aid of this application
- *User evaluation on translation: the participants should be able to fluently read the article by translating their unfamiliar words and sentences.*
- *User evaluation on note-taking: the participants should be able to take note easily*
- *User evaluation on evaluation: the evaluation result should be precise and reflect the real reading ability level.*

## **Secondary/Tertiary objectives**

[Translation system] The application should provide general dictionary interfaces at the backend so that users are able to import other dictionaries.

[Assessment system] The application should provide the evaluation result with respect to some international linguistic standards, such as CEFR<sup>[ref]</sup>.

[Software] The application should support other popular browsers, such as Firefox.

## Ethics

This project involves a user evaluation to assess the reliability and performance of software artefact. Participants will be invited to evaluate the artefact voluntarily and anonymously following the school's online evaluation rules<sup>[\[ref\]](#)</sup>.

The ethical issues have been evaluated against the school's preliminary self-assessment form and artefact ethical approval application. The artefact evaluation form will be signed and uploaded through MMS.

## Resource

This project requires no additional software or hardware, the artefact development can be completed using the current configuration of school lab machines.

Time	Month	January		Febuary			March			April				May			
	Semester time	Week 1 (1.25)	Week 2 (2.1)	Week 3 (2.8)	Week 4 (2.15)	Week 5 (2.22)	Week 6 (3.1)	Week 7 (3.8)	Week 8 (3.15)	Vacation	Week 9 (4.5)	Week 10 (4.12)	Week 11 (4.19)	Week 12 (4.26)	Week 13 (5.3)	Week 14 (5.10)	Week 15+ (5.17)
Project Preparation	DOER																
	Context survey																
Code Implementation	Browser extension framework																
	Translation module: translation API adaption																
	Translation module: text display																
	Note module																
Documentation	User evaluation																
	Final report																

UNIVERSITY OF ST ANDREWS  
TEACHING AND RESEARCH ETHICS COMMITTEE (UTREC)  
SCHOOL OF COMPUTER SCIENCE  
ARTIFACT EVALUATION FORM

Title of project

An aid to learning and accessing to read foreign languages

Name of researcher(s)

Yuxuan Wang

Name of supervisor

Mark-Jan Nederhof

Self audit has been conducted YES ☒ NO ☐

This project is covered by the ethical application CS12476 (amended for 2019/20 due to COVID-19)

Signature Student or Researcher

Yuxuan Wang

Print Name

Yuxuan Wang

Date

29 / 01 / 2021 .

Signature Lead Researcher or Supervisor

M J Nederhof

Print Name

M J Nederhof

Date

29 / 01 / 2021 .