Time kya hua

Rethinking Technical Translation for India

3 min read · Just now

Yogesh Haribhau Kulkarni (PhD)

(Source: Pixabay)

Technical knowledge grows globally in English, but its adoption in India depends on how naturally it can be consumed. For decades, we have tried to solve this gap using direct translation into Indian languages. The intent was noble, but the outcome often missed the mark.

Literal translations of technical content sound grammatically correct, yet feel alien. Learners struggle not because the concept is hard, but because the language feels unnatural.

Take a simple programming sentence: “This class has 4 member variables.”

Google translation to Hindi: इस क्लास में 4 सदस्य चर हैं।

As you can see, no one can understand this. What we want (of course with respect to domain as Programming):

- Hindi-Roman: iis class mein chaar member variables hai
- Hindi-Devanagari: इस क्लास में चार मेंबर व्हेरिएबल्स है ।

A programming student does not think in “सदस्य चर”. They think in “member variable”. The brain switches context, and comprehension slows down.

In daily life, Indians already solved this problem organically. We speak in hybrids. We say “server down hai”, “code debug karna hai”, and “traffic bahut zyada hai”. These sentences are not broken. They are optimized for communication.

Some examples

Programming

- English: In the for loop, value of i is assigned first then incremented.
- Hinglish-Roman: for loop mein pahile I keep value assign hoti hai baad mein badhati hai.
- Hinglish-Devnagari: फॉर लूप में पहिले आय की व्हॅल्यू असाईन होती हैं बाद में बढती हैं.

DevOps

- English: The server is down due to high traffic.
- Hinglish-Roman: Server down hai, traffic bahut zyada hai.
- Hinglish-Devnagari: सर्वर डाउन है, ट्रैफिक बहुत ज्यादा है।

Programming

- English: We need to debug the code.
- Hinglish-Roman: Code ko debug karna hai.
- Hinglish-Devnagari: कोड को डीबग करना है।

Apps

- English: This app has a user-friendly interface.
- Hinglish-Roman: Is app ka interface user-friendly hai.
- Hinglish-Devnagari: इस ऐप का इंटरफेस यूजर-फ्रेंडली है।

This natural solution can be formalized into a translation methodology. The idea is simple: do not translate domain-specific terminology. Transliterate it. Translate everything else.

This creates a structured hybrid language:

- Hinglish for Hindi
- Minglish for Marathi
- And more broadly, Inglish for Indian languages

The approach is domain-agnostic. Programming today, physics tomorrow, mathematics next. Every domain has its own vocabulary that should remain intact across languages. Words like “for loop”, “derivative”, “vector”, or “algorithm” carry precise meanings and global consistency. Translating them often creates confusion rather than clarity.

A simple AI-assisted pipeline can achieve this:

- Start with an English sentence.
- Identify technical terms using a domain glossary.
- Mark these as protected or guarded.
- Translate the remaining text into the target language.
- Transliterate the guarded terms into native script or keep them in Roman form.
- Ensure grammatical flow and readability.

The output should always be available in both Roman and native scripts. Roman helps digital natives. Native script supports inclusivity and formal publishing.

This is not an argument against Indian languages. It is an argument for making them usable in modern technical contexts. Language survives by adapting, not by freezing.

Future work can include:

- Standardized domain glossaries per language.
- Evaluation metrics focused on comprehension, not purity.
- AI models trained specifically for Inglish generation.
- Adoption in textbooks, MOOCs, and government initiatives.

If India wants mass technical literacy, the question is simple: should we teach people how language ought to be used, or how people actually use language?

Technical Translation | Artificial Intelligence | Hinglish | India | Open Source