

# ANALYSIS AND DESIGN OF A PLATFORM FOR REAL-TIME SPEECH TRANSLATION

JON FRIIS JAKOBSEN    123456789

Masters Project in Software Engineering

June 1887



The Maersk Mc-Kinney Moeller Institute  
University of Southern Denmark

## **Abstract**

Many companies are struggling to combine Java and Machine Learning...

# Contents

<b>Contents</b>	<b>ii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background and Motivation . . . . .	1
1.2 Problem Statement . . . . .	1
1.3 Research Questions . . . . .	2
1.4 Success Criteria . . . . .	2
<b>2 Background</b>	<b>4</b>
<b>3 Analysis and Design</b>	<b>5</b>
<b>4 Implementation</b>	<b>6</b>
<b>5 Evaluation</b>	<b>7</b>
<b>6 Conclusion</b>	<b>8</b>
<b>A Appendix</b>	<b>9</b>
<b>Bibliography</b>	<b>10</b>

# 1 Introduction

## 1.1 Background and Motivation

In a world marked by accelerating globalization and rapid digitalization, effective cross-language communication has become a critical enabler of collaborative innovation, knowledge transfer, and equitable access to information. Real-time speech-to-speech translation—the ability to convert spoken utterances from one language to another with minimal latency—holds significant promise for breaking down linguistic barriers and fostering inclusion across cultural and geographical boundaries. This potential extends not merely to large enterprises but equally to smaller organizations, public institutions, and civil society actors, for whom linguistic accessibility can profoundly influence participation and equal opportunity [11].

The market for real-time speech translation is today dominated by large technology companies including Google (Alphabet Inc.), Microsoft Corporation, and Amazon. These organizations offer integrated solutions through Google Translate [7], Microsoft Translator [9], and AWS services combining Amazon Transcribe, Amazon Translate, and Amazon Polly [2, 3, 1]. While these platforms deliver functional capabilities, they are typically embedded within larger, proprietary ecosystems that can be expensive, restrictive in customization, and difficult to integrate for smaller actors, open-source projects, or specialized applications. Moreover, independent research into optimal architectural principles for such systems remains limited, particularly regarding the critical balance between latency, horizontal scalability, and robustness in distributed environments [14, 15].

This thesis therefore investigates an alternative architectural approach grounded in event-driven microservices. The premise is that by decomposing the speech translation pipeline into loosely coupled, independently scalable services communicating asynchronously through an event backbone, we can achieve both the flexibility required for customization and the performance characteristics necessary for real-time operation.

## 1.2 Problem Statement

Existing research on microservices architecture and real-time translation systems typically concentrates on either improving individual AI components—such as automatic speech recognition (ASR) models or machine translation quality—or on end-to-end monolithic systems. While advances in component-level

technologies have been substantial (e.g., neural machine translation [4, 12]; modern ASR systems [10]; and large-scale translation frameworks [13]), there remains a significant gap in systematic, architecture-level guidance for integrating multiple AI services under strict latency constraints.

In particular, the literature offers limited empirical understanding of how event-driven communication can be optimized to handle the sequential data flow of speech through multiple processing stages while simultaneously enabling horizontal scaling and maintaining high availability under variable load. Questions around message ordering, delivery semantics (at-least-once vs exactly-once), payload constraints, failure recovery, and observability remain unspecified in the context of real-time, multi-stage AI pipelines. Furthermore, the engineering trade-offs between latency, throughput, fault tolerance, and resource consumption—critical to thesis-grade evaluation—have not been systematically measured and documented for this domain.

### 1.3 Research Questions

The above context motivates the following inquiry:

**Main Research Question (MRQ):**

How can an event-driven microservice architecture be designed and implemented to deliver robust and scalable real-time speech-to-speech translation with measurable performance optimization?

This overarching question is decomposed into five specific sub-questions:

**SQ1** Which architectural principles and design patterns support an effective real-time speech translation pipeline?

**SQ2** Which factors influence end-to-end latency and the ability to scale?

**SQ3** How can these factors be measured and optimized in practice?

**SQ4** How can the system remain robust and available when individual components fail?

**SQ5** How can system performance and output quality be evaluated and compared against existing solutions?

### 1.4 Success Criteria

To ensure rigorous and objective evaluation, this thesis establishes the following measurable success criteria across system performance and output quality dimensions:

**Latency** Measurement of end-to-end latency from audio ingress to translated output, reported using percentile statistics (P50, P90, P99) and including

per-stage latency decomposition where feasible. This addresses SQ2 and SQ3.

**Capacity and Scalability** Determination of maximum sustainable throughput (measured in requests per second or concurrent audio streams) while maintaining latency within acceptable bounds. This directly addresses SQ2 and establishes the baseline for scaling experiments.

**Robustness and Availability** Experimental validation of system behavior under adverse conditions including component failures, overload scenarios, and malformed messages. Measurements include error recovery rate, time to recovery, and graceful degradation characteristics. This addresses SQ4.

**Resource Utilization** Profiling of CPU and memory consumption as a function of throughput and latency, providing insight into cost-efficiency trade-offs. This informs SQ2 and practical deployment decisions.

**Output Quality** Evaluation of ASR output quality using standard metrics (Word Error Rate, Character Error Rate) and translation quality using both automated metrics (BLEU, COMET) and where feasible human evaluation. This addresses SQ5 and enables comparison with commercial solutions.

**Deployability and Extensibility** Demonstration of deployment on a cloud platform and extension to an additional language pair with minimal engineering effort, validating the claimed flexibility of the microservices approach. This addresses the practical utility and architectural generalizability of the design.

## 2 Background

### **State of the Art**

Gennemgang af eksisterende løsninger (Google Translate, Microsoft Translator, AWS).

### **Microservices vs. Monoliths**

Teori om hændelsesdrevet arkitektur (Newman, Fowler).

### **AI Components**

Teoretisk baggrund for ASR (f.eks. Whisper), MT og TTS.

### **Event-Driven Principles**

Hvorfor Kafka og asynkron kommunikation er valgt til realtidsdata.

## 3 Analysis and Design

### System Architecture

Beskrivelse af Kafka-broderen, Zookeeper og Confluent Schema Registry. Hvordan mikroservices kommunikerer asynkront via Kafka-emner.

### Data Modeling

Dokumentation af dine Avro-schemas (BaseEvent.avsc, VoiceDetectedEvent.avsc) for at sikre type-sikkerhed.

### End-to-End Pipeline

Indsæt dine Mermaid-sekvensdiagrammer, der viser flowet fra API Gateway gennem VAD, ASR, Translation og TTS.

### Orchestration Logic

Hvordan din Pipeline Orchestrator styrer tilstanden og bruger Correlation IDs til tracking.

## 4 Implementation

### **Container Orchestration**

Opsætning af Kubernetes (GKE/EKS) med GPU-optimerede node-pools til ML-modeller.

### **Scaling Strategies**

Implementering af Horizontal Pod Autoscaler (HPA) og din prædiktive skaleringsmodel.

### **CI/CD & GitOps**

Brug af ArgoCD til Blue-Green og Canary deployments.

### **Metrics Collection Setup**

Implementering af Prometheus, Grafana og din "Metrics Collector Service" til opsamling af forskningsdata.

## 5 Evaluation

### Performance Results

Præsentation af latens-målinger (mål:  $< 3$  sekunder).

### Scalability Testing

Grafer der viser systemets adfærd under stigende belastning (op til 1000+ events/sek).

### Cross-Linguistic Analysis

Evaluering af hvordan forskellige sprogfamilier påvirker systemets præcision og hastighed.

### Robustness Evaluation

Test af fejlscenarier (f.eks. service-nedbrud) og systemets evne til auto-recovery.

## 6 Conclusion

### Summary

Sammenfatning af hvordan din event-drevne arkitektur løser problemet med skalering og latens.

### Future Work

Forslag til forbedringer, såsom integration af flere sprog, forbedret fejlhåndtering og avancerede skaleringsalgoritmer.

# A Appendix

We include here the API documentation for our library.

# Bibliography

- [1] Amazon Web Services. Amazon polly. <https://aws.amazon.com/polly/>, 2024. Accessed: 2026-01-26.
- [2] Amazon Web Services. Amazon transcribe. <https://aws.amazon.com/transcribe/>, 2024. Accessed: 2026-01-26.
- [3] Amazon Web Services. Amazon translate. <https://aws.amazon.com/translate/>, 2024. Accessed: 2026-01-26.
- [4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [5] Len Bass, Paul Clements, and Rick Kazman. *Software Architecture in Practice*. Addison-Wesley Professional, 4 edition, 2021.
- [6] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, Cambridge, Massachusetts, second edition, 2001.
- [7] Google Cloud. Cloud translation. <https://cloud.google.com/translate>, 2024. Accessed: 2026-01-26.
- [8] Tim Lindholm and Frank Yellin. *The Java<sup>TM</sup> Virtual Machine Specification*. Addison-Wesley, 2nd edition, 1999.
- [9] Microsoft. Microsoft translator. <https://www.microsoft.com/translator/>, 2024. Accessed: 2026-01-26.
- [10] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *arXiv preprint arXiv:2212.04356*, 2022.
- [11] UNESCO. Inclusion and education: All means all. <https://www.unesco.org/>, 2022. Accessed: 2026-01-26.
- [12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

- [13] Changhan Wang, Anne Wu, Juan Pino, Alexei Baevski, Michael Auli, and Alexis Conneau. Fairseq s2t: Fast speech-to-text modeling with fairseq. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.
- [14] J. Xu et al. Scalability challenges in distributed speech translation systems, 2025. Preprint.
- [15] Y. Zhang et al. Direct speech-to-speech translation: Architecture and optimization, 2024. Preprint.