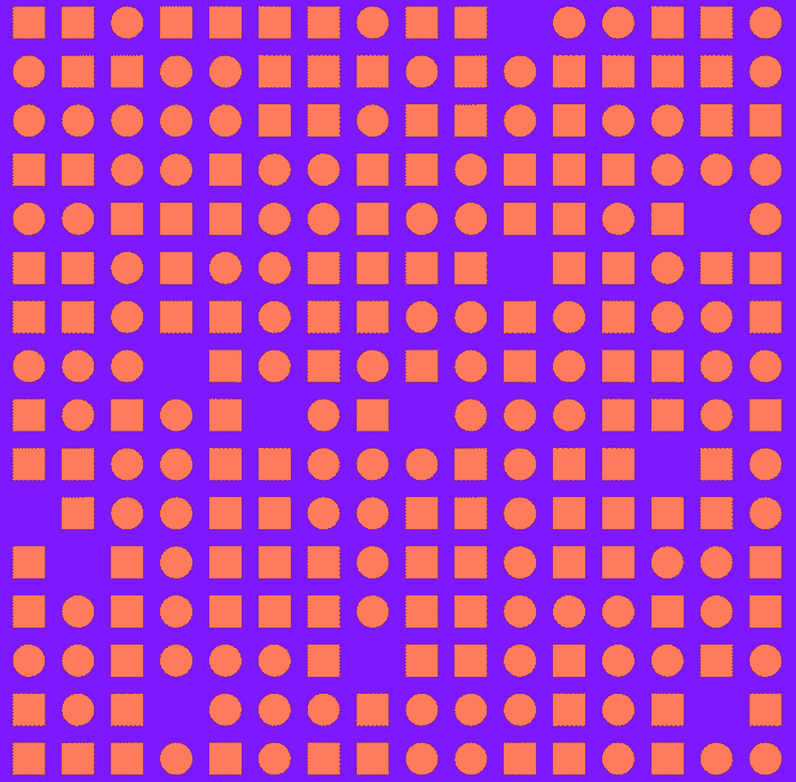
TypeFox

Langium + AI

Building AI Applications for Langium DSLs



LangDev CON 2024

Overview

1. Introduction
2. Background
3. Problem
4. Solution
5. Implementation
6. Demo
7. Wrap-up

Introduction

Introduction

Myself

- Benjamin Friedman Wilson
- Language Engineer @ TypeFox s. 2022
- Working on DSLs (Langium)
- Full Stack Applications

Introduction

And TypeFox

- Benjamin Friedman Wilson
- Language Engineer @ TypeFox s. 2022
- Working on DSLs (Langium)
- Full Stack Applications

- Topics at TypeFox
  - Language Engineering services
  - Cloud & Desktop IDEs
  - Modeling & Diagramming

Background

Background

Motivation for Digging into this

- Langium has been doing *great*
- AI Assistants have been doing *great*
  - CoPilot, Tab9, Anthropic, ChatGPT,…

Background

Motivation for Digging into this

- Langium has been doing *great*
- AI Assistants have been doing *great*
  - CoPilot, Tab9, Anthropic, ChatGPT,…

Which leads to the following:

How can I get AI to work with my DSL?

Background

Motivation for Digging into this

- Langium has been doing *great*
- AI Assistants have been doing *great*
  - CoPilot, Tab9, Anthropic, ChatGPT,…

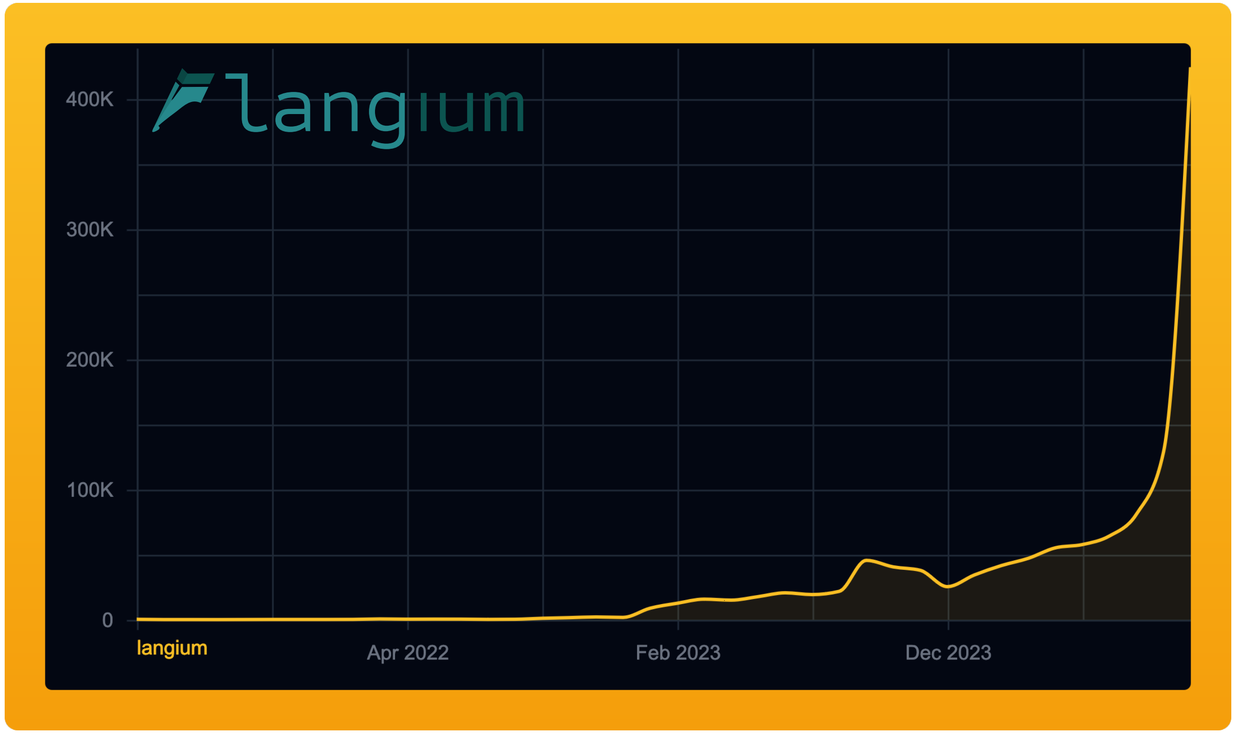
Which leads to the following:

How can I get AI to work with my DSL?

- But for *novel* DSLs:
  - Not much training data
  - Not many examples

Background

And a Quick Aside on Eclipse Langium

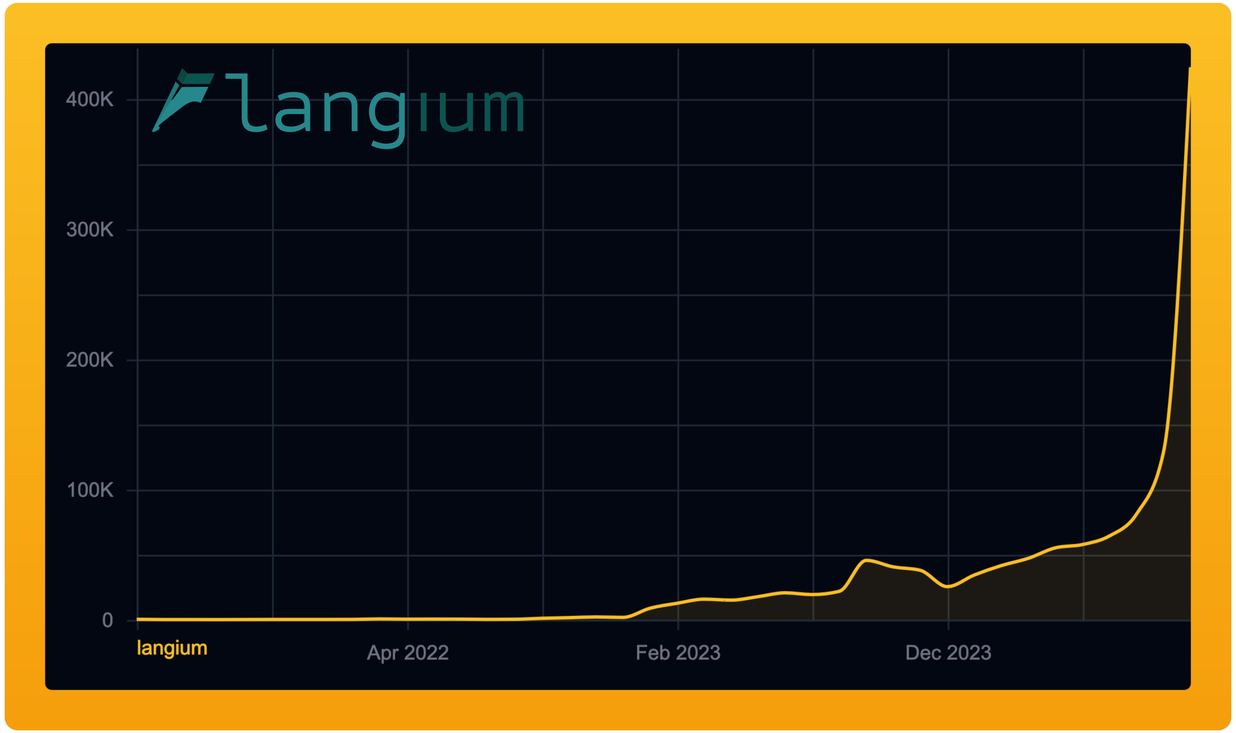


- Langium is an LE Framework written in TS
- Based on Chevrotain (parser-builder)
- Runs on Server & Browser

- LSP support out of the box for your DSLs
- Highly customizable w/ suitable defaults
- Now an *Eclipse Foundation* project

Background

And a Quick Aside on Eclipse Langium



- Langium is an LE Framework written in TS
- Based on Chevrotain (parser-builder)
- Runs on Server & Browser

- LSP support out of the box for your DSLs
- Highly customizable w/ suitable defaults
- Now an *Eclipse Foundation* project

- If you haven't heard of it, it's akin to *Xtext*

Problem

Problem

Can we get AI to support language XYZ?

Problem

Can we get AI to support language XYZ?

Yes, but why?

Problem

Can we get AI to support language XYZ?

Yes, but why?
No, and why?

Problem

Can we get AI to support language XYZ?

- Generally speaking
  - Yes.

Problem

Can we get AI to support language XYZ?

- Generally speaking
  - Yes.
- With a Langium DSL?
  - *Kinda.*

Problem

Can we get AI to support language XYZ?

- Generally speaking
  - Yes.
- With a Langium DSL?
  - *Kinda.*
- Are the results good?
  - No.

Problem

Can we get AI to support language XYZ?

- Generally speaking
  - Yes.
- With a Langium DSL?
  - *Kinda.*
- Are the results good?
  - No.
- Are they getting better?
  - Yes, but it depends…

Problem

Can we get AI to support Langium DSLs?

- Langium itself will get support passively
  - Present in data for GPT, Claude, etc.
- But *not* so much for Langium-based DSLS

Problem

Can we get AI to support Langium DSLs?

- Langium itself will get support passively
  - Present in data for GPT, Claude, etc.
- But *not* so much for Langium-based DSLS

- **Crux of the problem**
  - *How to get AI support for novel Langium DSLs?*

Problem

Issues with Supporting Langium DSLs

- No pre-existing training data
  - i.e. it’s not in the model

Problem

Issues with Supporting Langium DSLs

- No pre-existing training data
  - i.e. it’s not in the model
- Often few or limited examples

Problem

Issues with Supporting Langium DSLs

- No pre-existing training data
  - i.e. it’s not in the model
- Often few or limited examples
- Documentation is often limited

Problem

Issues with Supporting Langium DSLs

- No pre-existing training data
  - i.e. it’s not in the model
- Often few or limited examples
- Documentation is often limited

- Language is often *evolving*
  - Creates a moving target

Problem

Overview

- We need:
  - a quick way to build AI agents w/ DSL knowledge
  - to be able to iterate on the agent and the DSL
  - to be able to reasonably trust the agent’s results

Solution

Solution

Langium AI

- Langium AI
  - Taps into an existing Langium DSL
  - Leverages existing DSL services
  - Provides:
    - DSL Doc splitting
    - AI App evaluation
- Naturally, also written in TypeScript

Solution

Langium AI

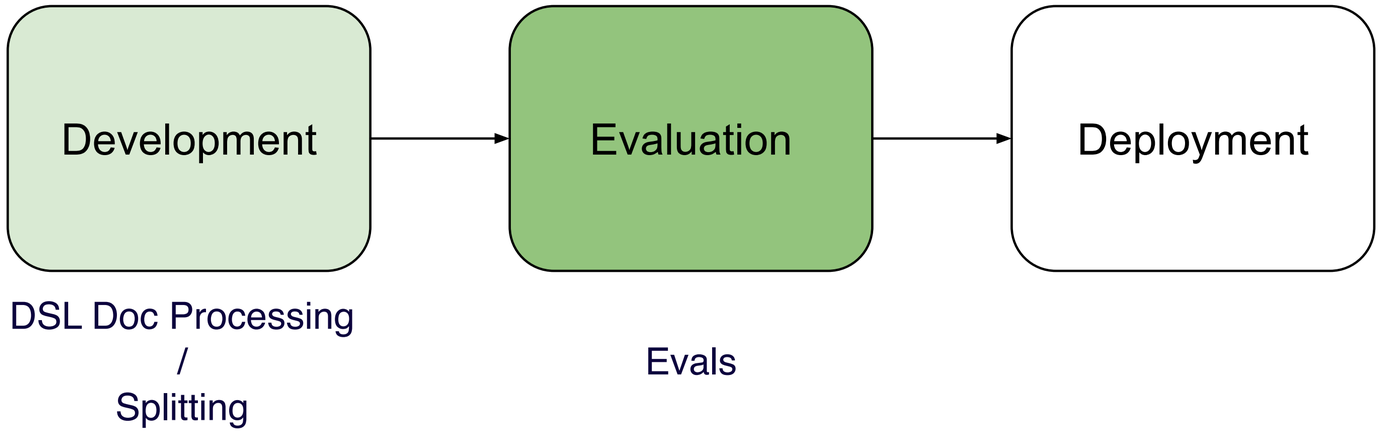
- Langium AI is for:
  - Langium developers
  - Those who need an agent for an LDSL
  - Not *necessarily* ML experts
- But it's *not* for:
  - Replacing existing models & stacks
  - Fine-tuning models
  - Training models
- *This complements existing AI apps that work with or support Langium DSLs*

Solution

Where does Langium AI Fit in?

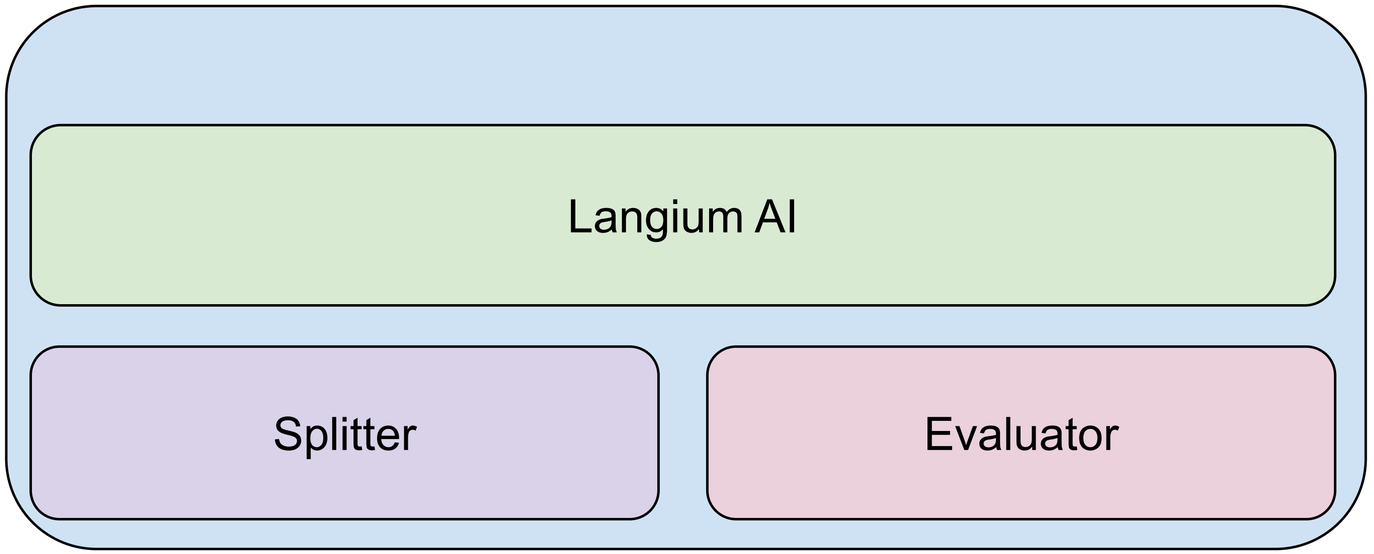
Solution

Where does Langium AI Fit in?



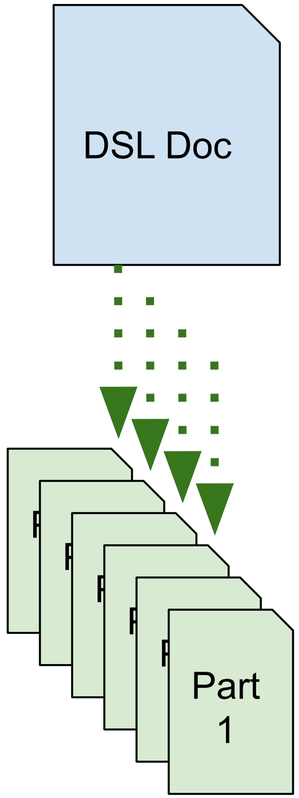
Solution

Splitter & Evaluator



Solution

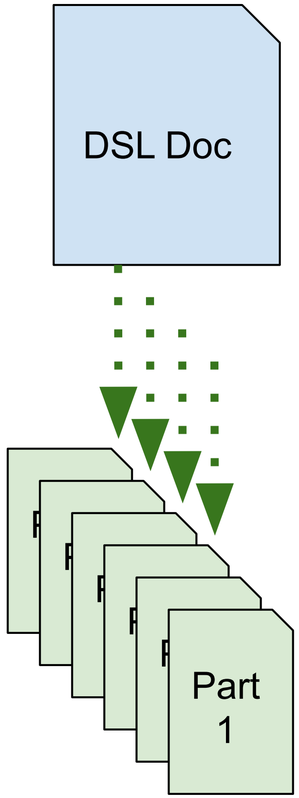
Splitter



- Assists in Splitting DSL Documents
  - Intended for RAG-based apps
    - Splitting by functions, blocks, etc.
  - Can help *reduce* larger documents
  - Can help *remove* redundant chunks

Solution

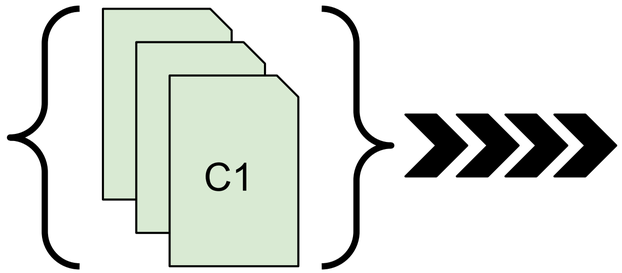
Splitter

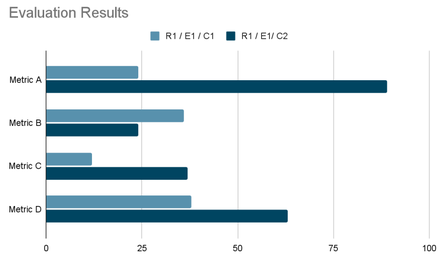


- Assists in Splitting DSL Documents
  - Intended for RAG-based apps
    - Splitting by functions, blocks, etc.
  - Can help *reduce* larger documents
  - Can help *remove* redundant chunks
- Idea is to split using the syntax
  - ex. Rules of your grammar / AST nodes

Solution

Evaluator

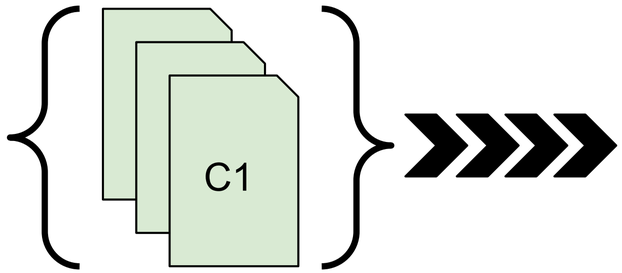


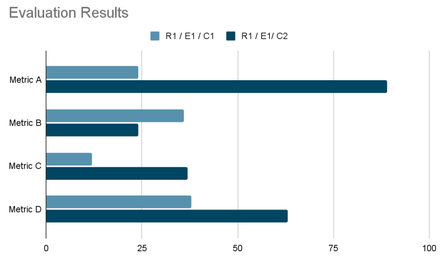


- Produce metrics from input/output pairs
  - Prompt & Response
- Provide context history as needed
- Collect metrics into a reportable form

Solution

Evaluator



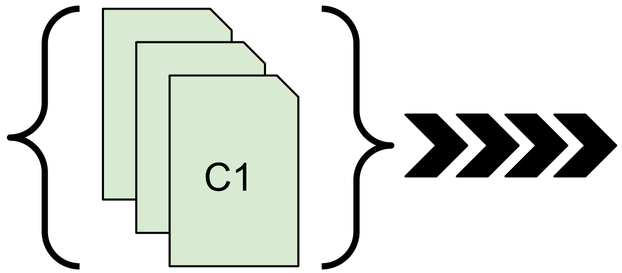


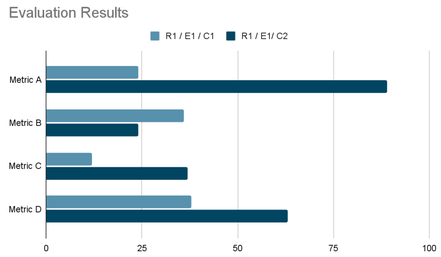
- Produce metrics from input/output pairs
  - Prompt & Response
- Provide context history as needed
- Collect metrics into a reportable form

- Independent of model or stack
- Helps determine *quality* of output
- Leverages your DSL's existing validations

Solution

Evaluator





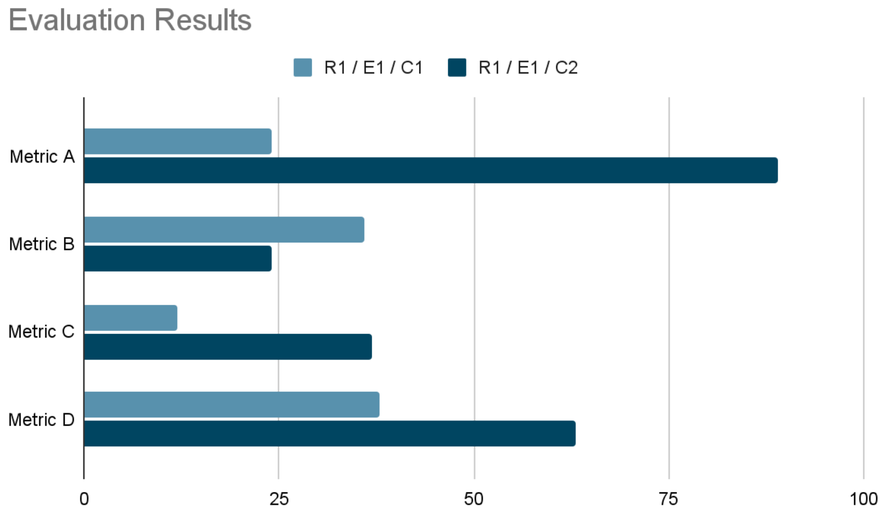
- Produce metrics from input/output pairs
  - Prompt & Response
- Provide context history as needed
- Collect metrics into a reportable form

- Independent of model or stack
- Helps determine *quality* of output
- Leverages your DSL's existing validations

- Lets us *iterate* quickly and maintain *trust*

Solution

And Evaluator Results

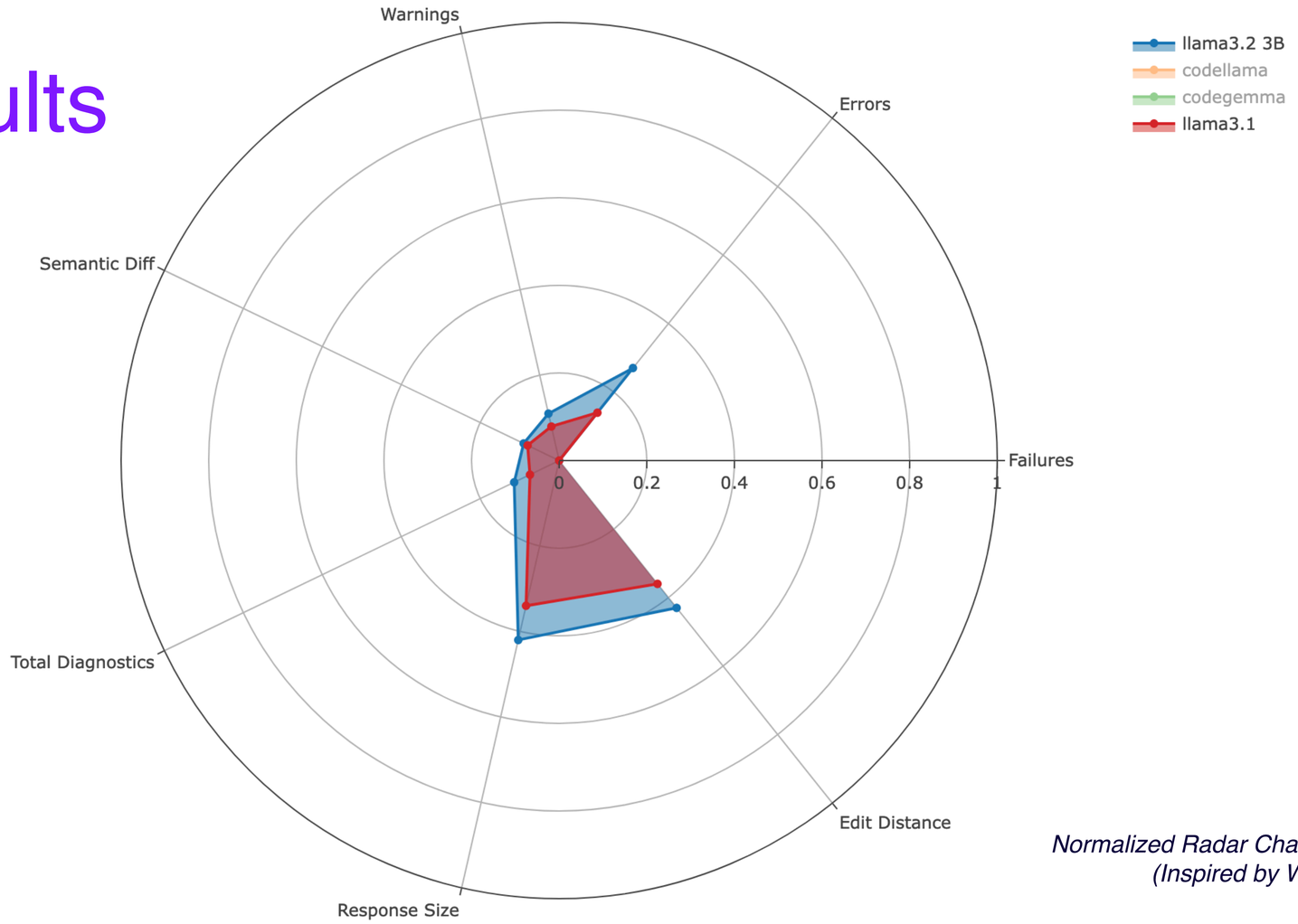


- Evaluation results capture
  - Input/Output
  - Metrics
  - Metadata

- Used for comparative analysis
- I.e. apples to apples analysis

Solution

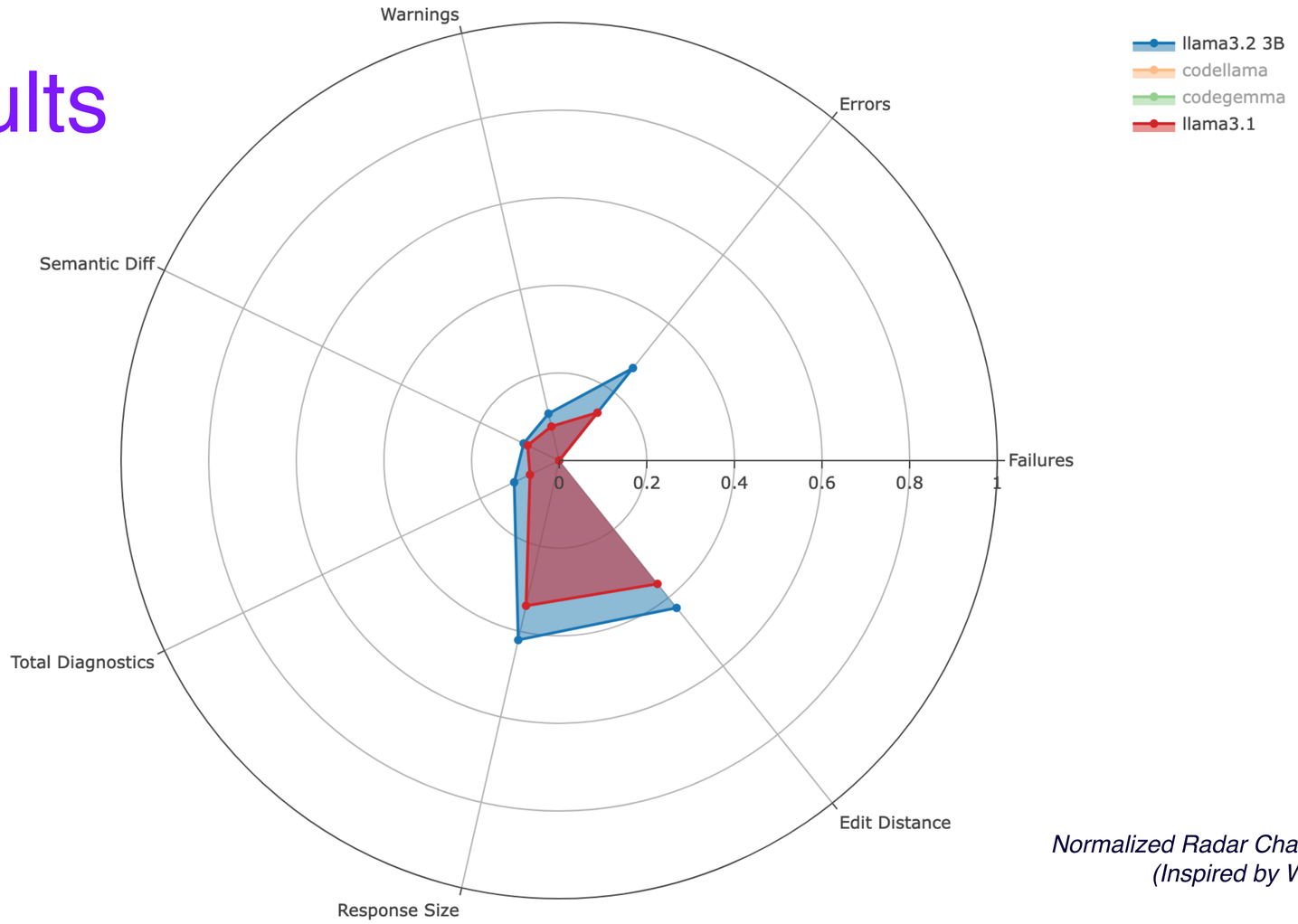
Evaluator Results



Normalized Radar Chart using PlotlyJS
(Inspired by WandB)

Solution

Evaluator Results



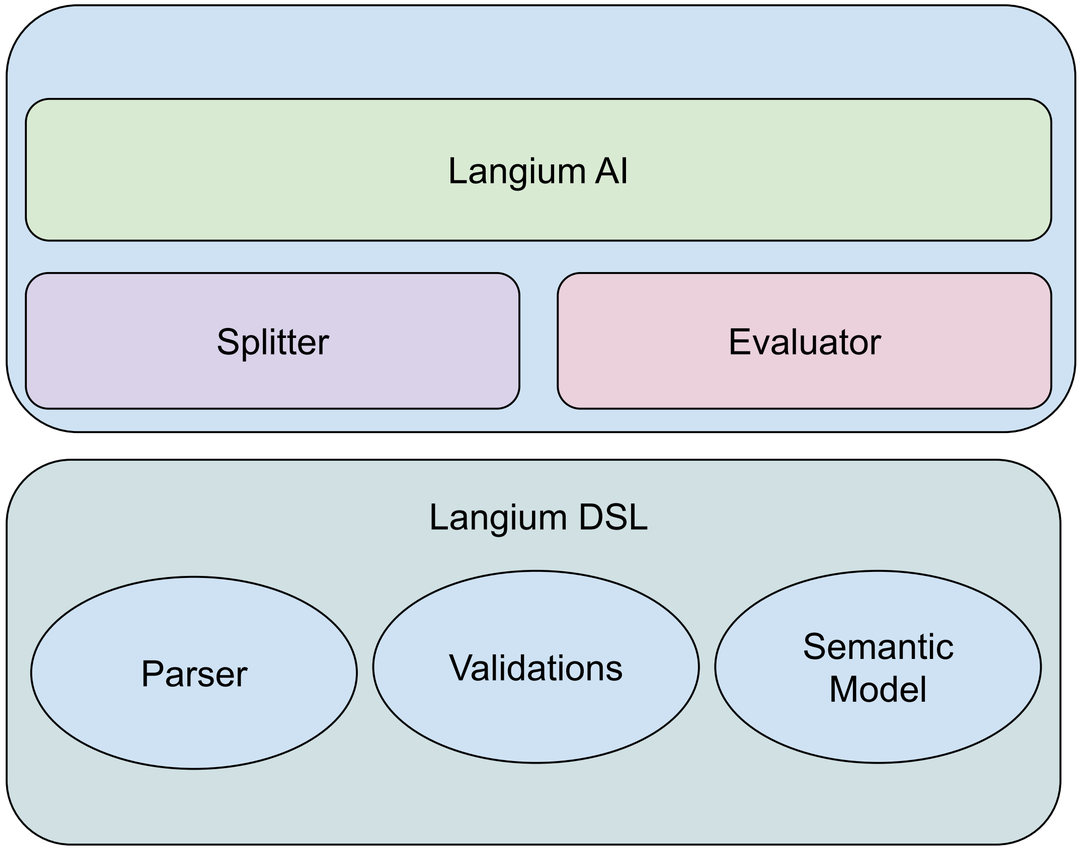
Normalized Radar Chart using PlotlyJS
(Inspired by WandB)

Implementation

Overview of Splitter & Evaluator

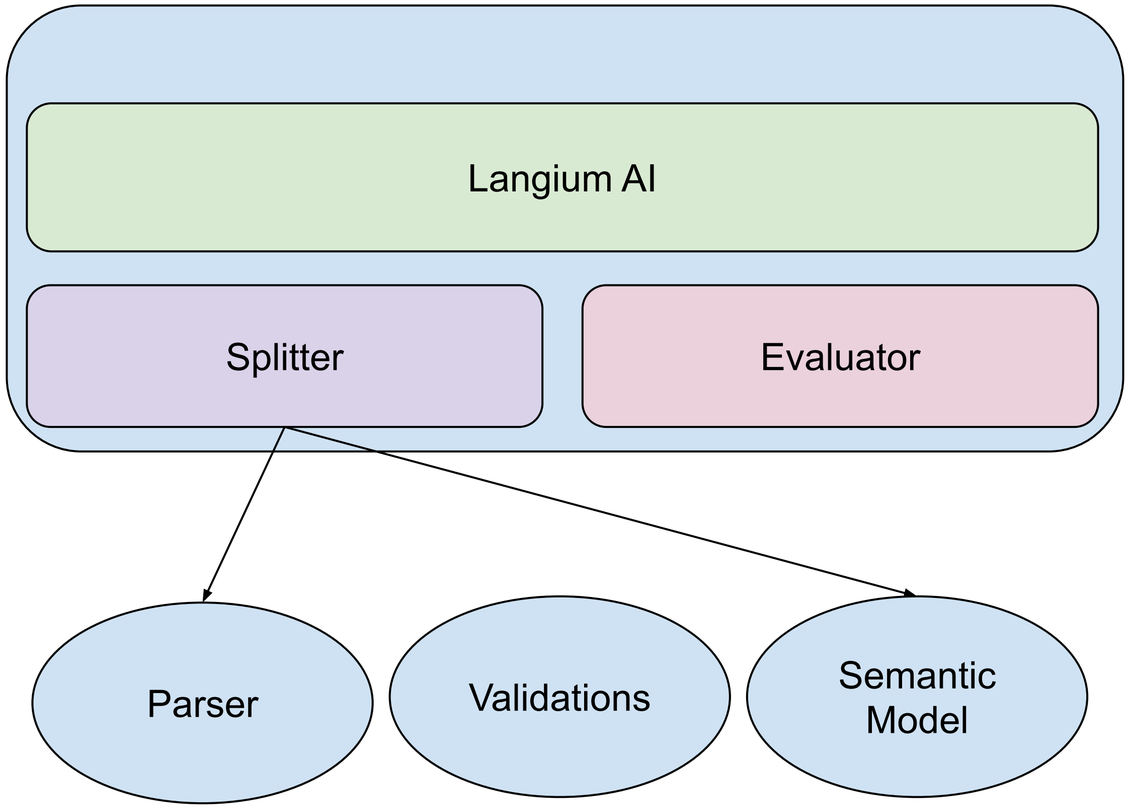
Implementation

Langium AI + Langium DSL



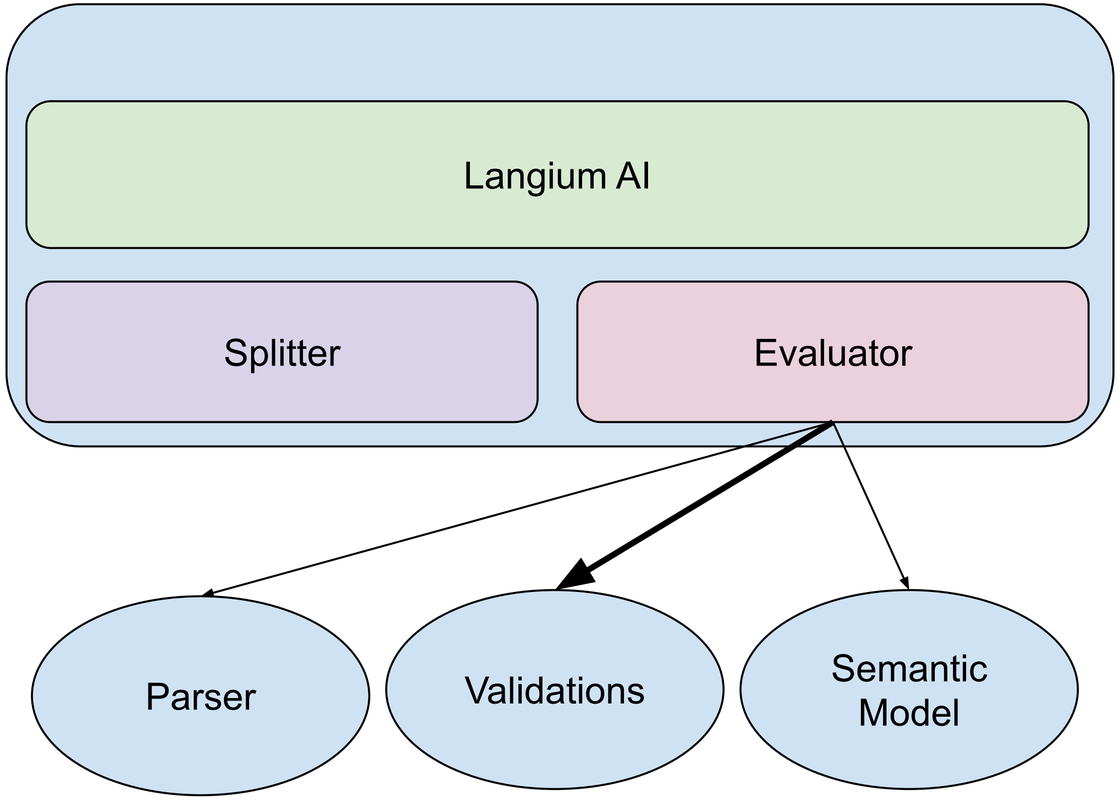
Implementation

Splitter



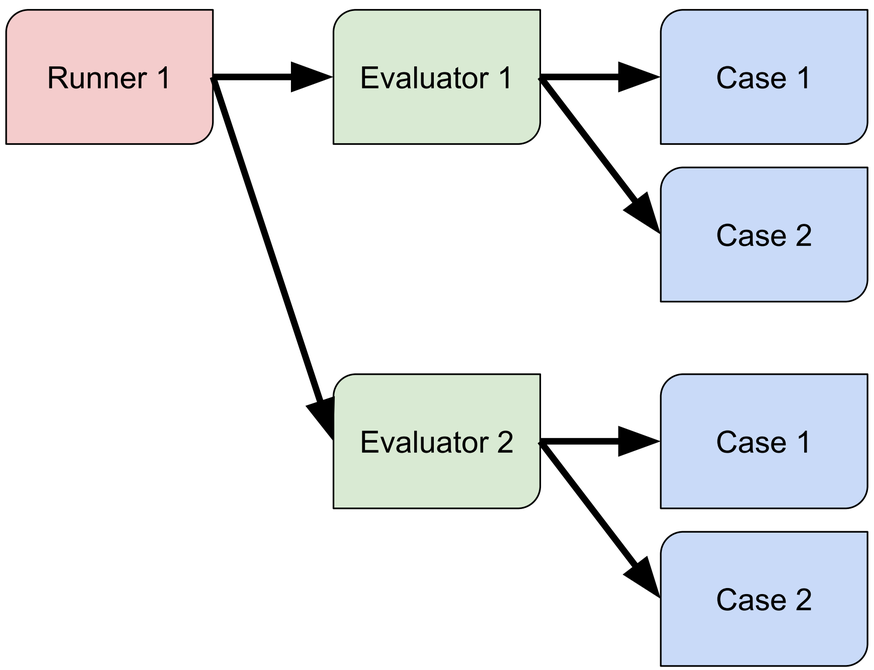
Implementation Overview

Evaluator



Implementation Overview

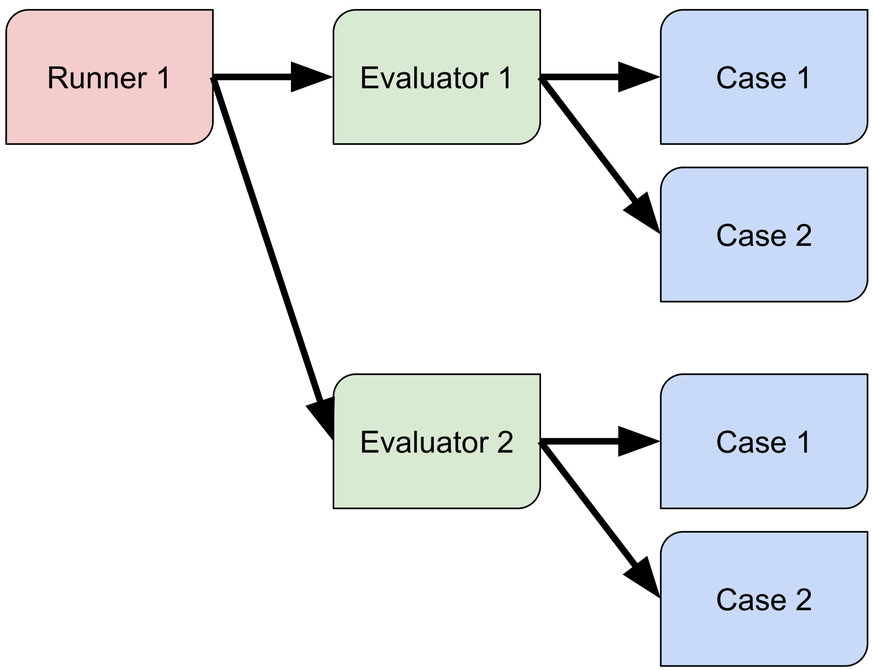
Evaluator



- Evaluations are defined as a matrix
  - Runners x Evaluators x Cases
  - **Runners** are invocable, take an input, run it on a model, and return the result
  - **Evaluators** define a strategy to assess a given response, provided some case
  - **Cases** define an input/output pair you want to assess

Implementation Overview

Evaluator



- Evaluations are defined as a matrix
  - Runners x Evaluators x Cases
  - **Runners** are invocable, take an input, run it on a model, and return the result
  - **Evaluators** define a strategy to assess a given response, provided some case
  - **Cases** define an input/output pair you want to assess
- LangiumEvaluator is provided
  - Uses validations as *metrics*

Demo

Evaluator Code for the Langium DSL

Summary

Langium AI

- Supports developing AI applications for Langium DSLs
  - Provides doc splitting utilities
  - Provides tools for evaluating AI app performance
- Facilitates iterative development of AI applications
- Agnostic of your AI tooling

Summary

Langium AI

- Supports developing AI applications for Langium DSLs
  - Provides doc splitting utilities
  - Provides tools for evaluating AI app performance
- Facilitates iterative development of AI applications
- Agnostic of your AI tooling

Ideally, this helps to assess what you're already working on

Final Remarks

- Currently a WIP (repo will come soon)
- Looking to push out an 0.2.0 later this year
- We'll notify on www.typefox.io/blog & socials
- Plan is to:
  - Expand on splitter
  - Improve evaluator
  - Iterate based on practical needs (while staying agnostic)

TypeFox

That’s all.
Thanks for your attention!