Automatic Performance Optimization with PerfExpert

Leo Fialho, Ashay Rane and Jim Browne

Agenda

TACC THE UNIVERSITY OF TEXAS AT AUSTIN

Introduction

Introduction

Steps for Performance Optimization

Introduction

Steps for Performance Optimization

- Metrics definition and performance measurement

Introduction

Steps for Performance Optimization

- Metrics definition and performance measurement
- Analysis, diagnosis and identification of bottlenecks

Introduction

Steps for Performance Optimization

- Metrics definition and performance measurement
- Analysis, diagnosis and identification of bottlenecks
- Recommendation of optimizations

Introduction

Steps for Performance Optimization

- Metrics definition and performance measurement
- Analysis, diagnosis and identification of bottlenecks
- Recommendation of optimizations
- Implementation of the recommended optimizations

Introduction

Steps for Performance Optimization

- Metrics definition and performance measurement
- Analysis, diagnosis and identification of bottlenecks
- Recommendation of optimizations
- Implementation of the recommended optimizations

Challenges of Automatic Performance Optimization

Introduction

Steps for Performance Optimization

- Metrics definition and performance measurement
- Analysis, diagnosis and identification of bottlenecks
- Recommendation of optimizations
- Implementation of the recommended optimizations

Challenges of Automatic Performance Optimization

- Programmers are really inventive

Introduction

Steps for Performance Optimization

- Metrics definition and performance measurement
- Analysis, diagnosis and identification of bottlenecks
- Recommendation of optimizations
- Implementation of the recommended optimizations

Challenges of Automatic Performance Optimization

- Programmers are really inventive
- The execution environment is quite complex

Introduction

Steps for Performance Optimization

- Metrics definition and performance measurement
- Analysis, diagnosis and identification of bottlenecks
- Recommendation of optimizations
- Implementation of the recommended optimizations

Challenges of Automatic Performance Optimization

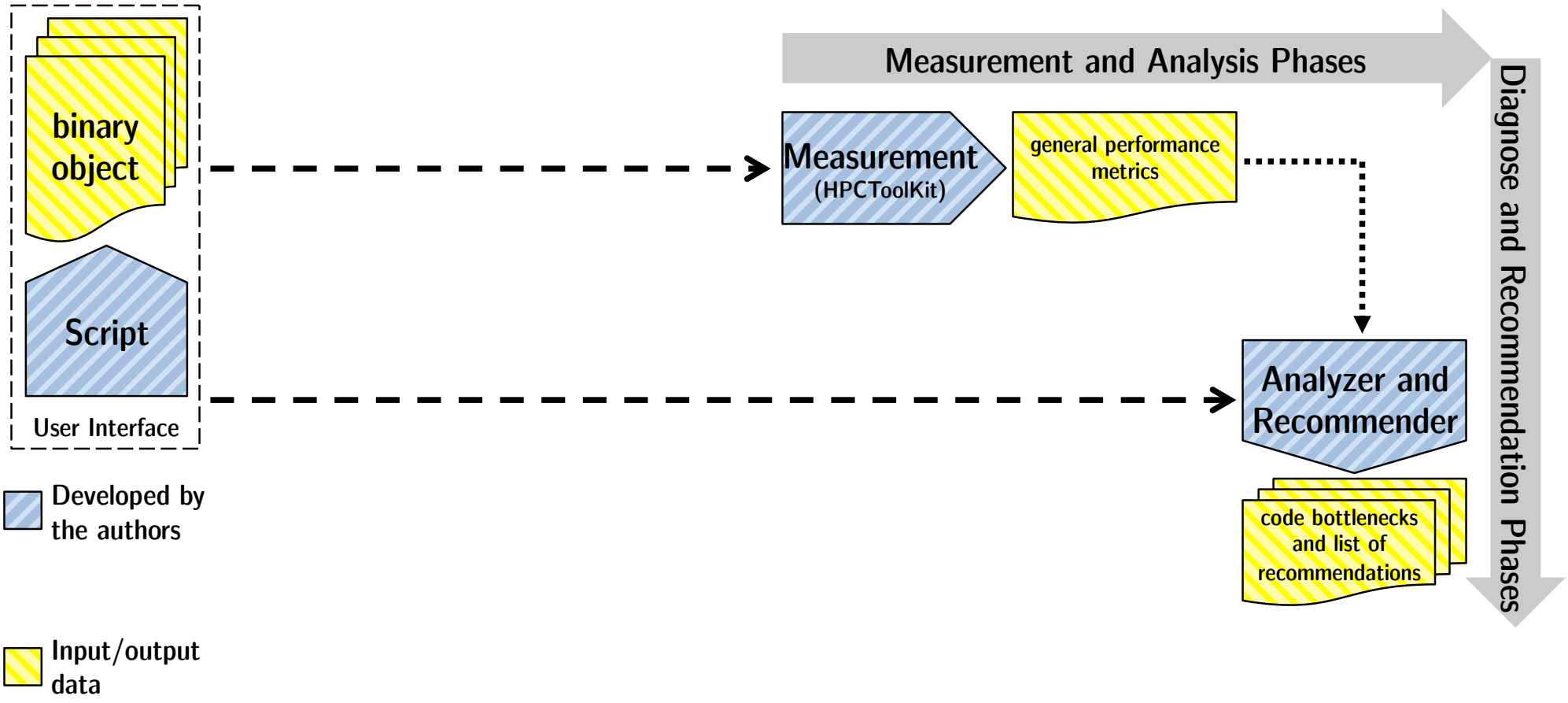
- Programmers are really inventive
- The execution environment is quite complex
- Compilers usually do only static source code analysis

TACC THE UNIVERSITY OF TEXAS AT AUSTIN

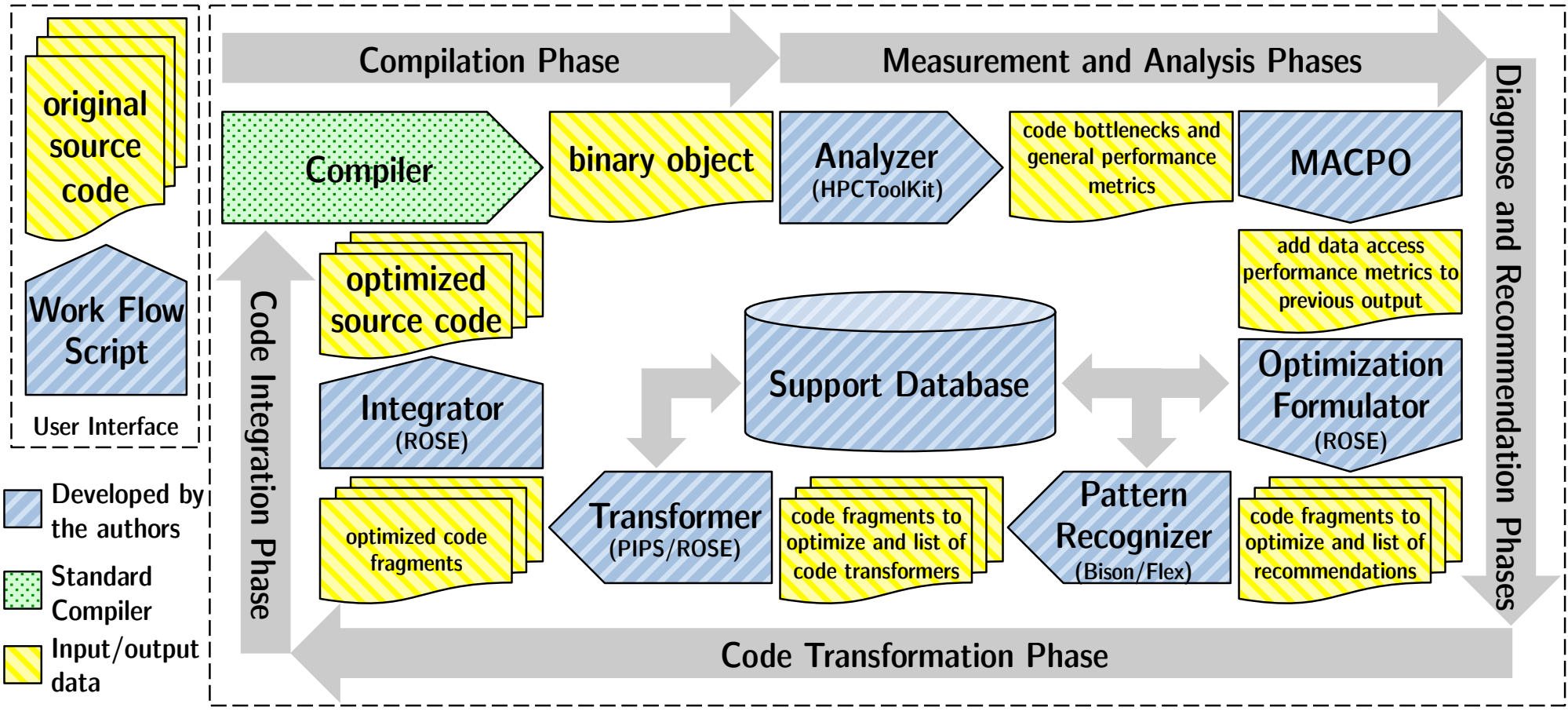
Agenda

TACC THE UNIVERSITY OF TEXAS AT AUSTIN

Current Version: The Big Picture



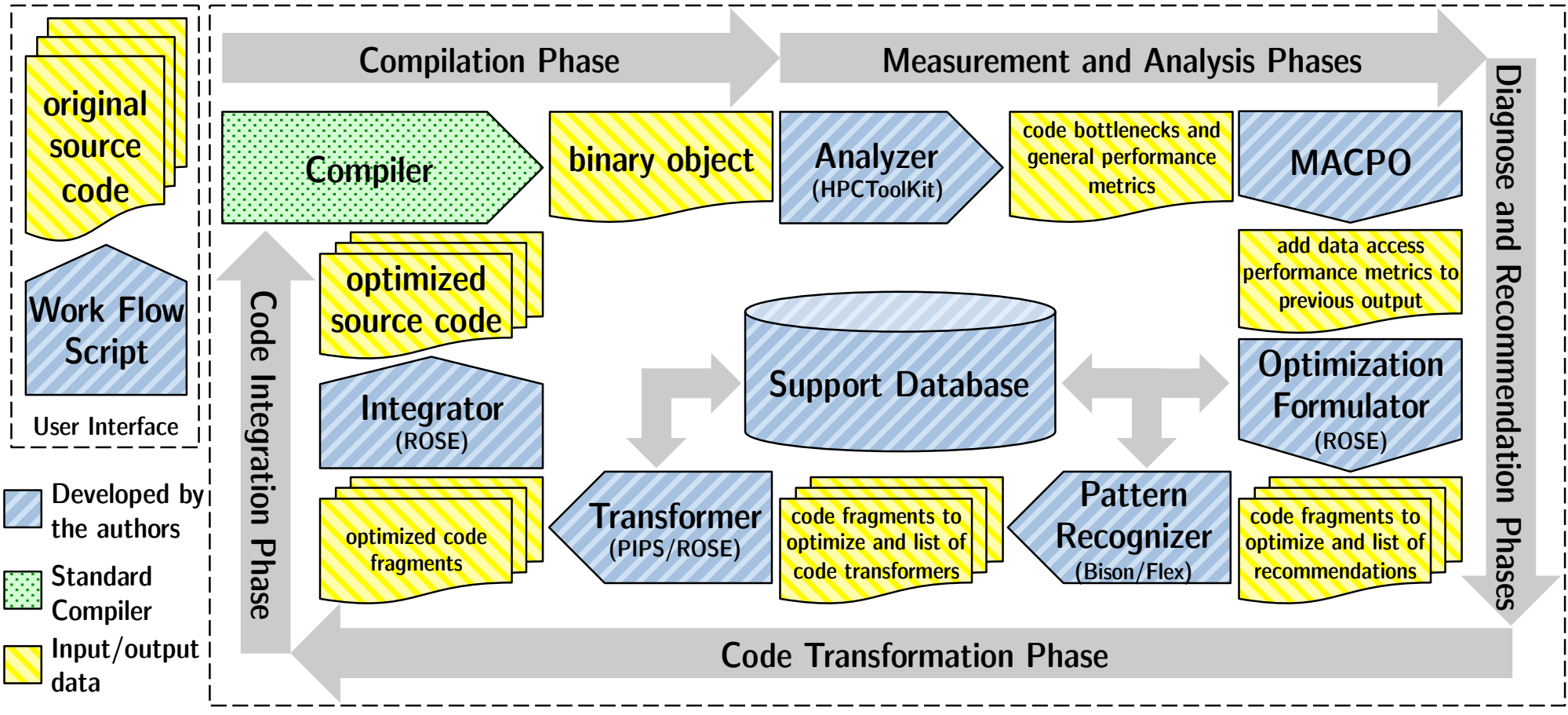
New Version: The Big Picture



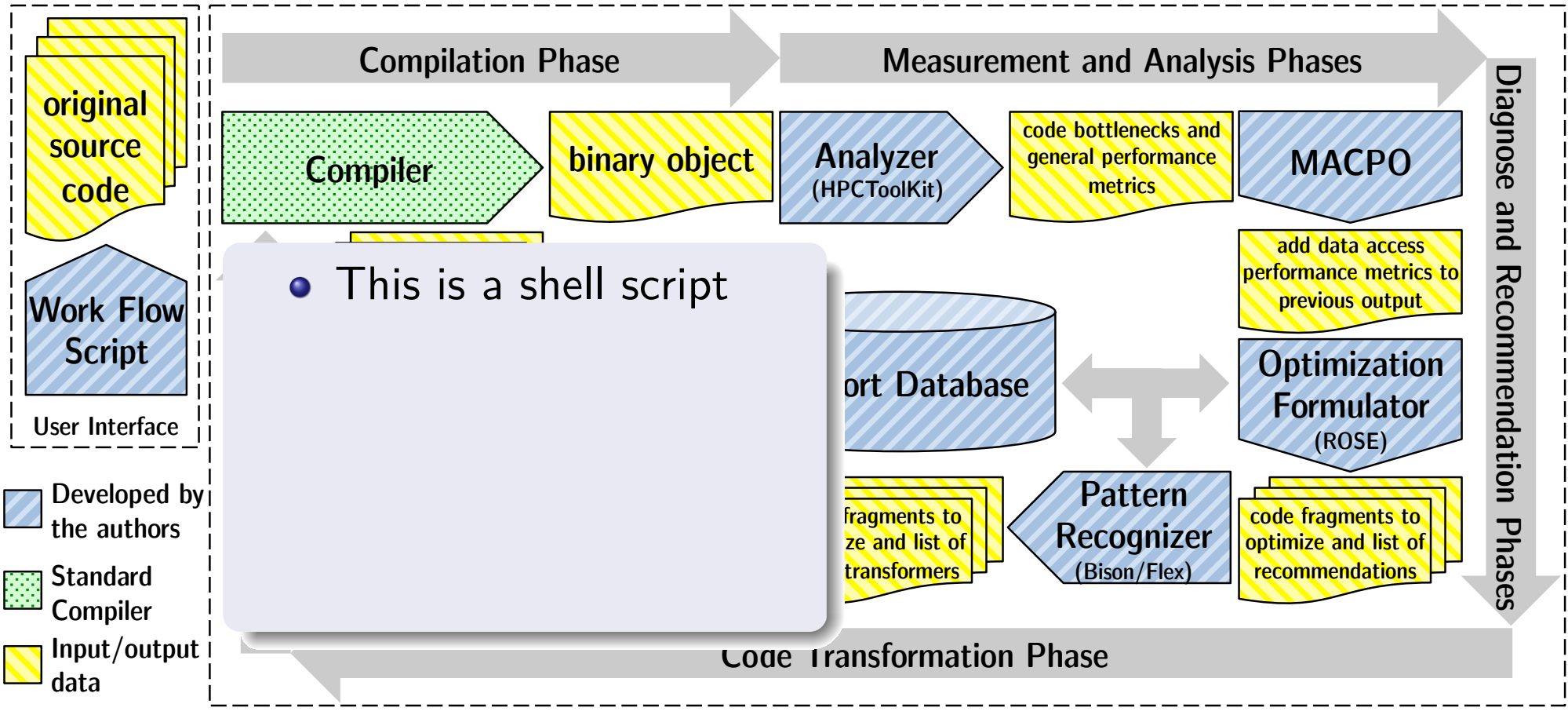
New Version: Short Demo

Short demo

New Version: Work Flow Script

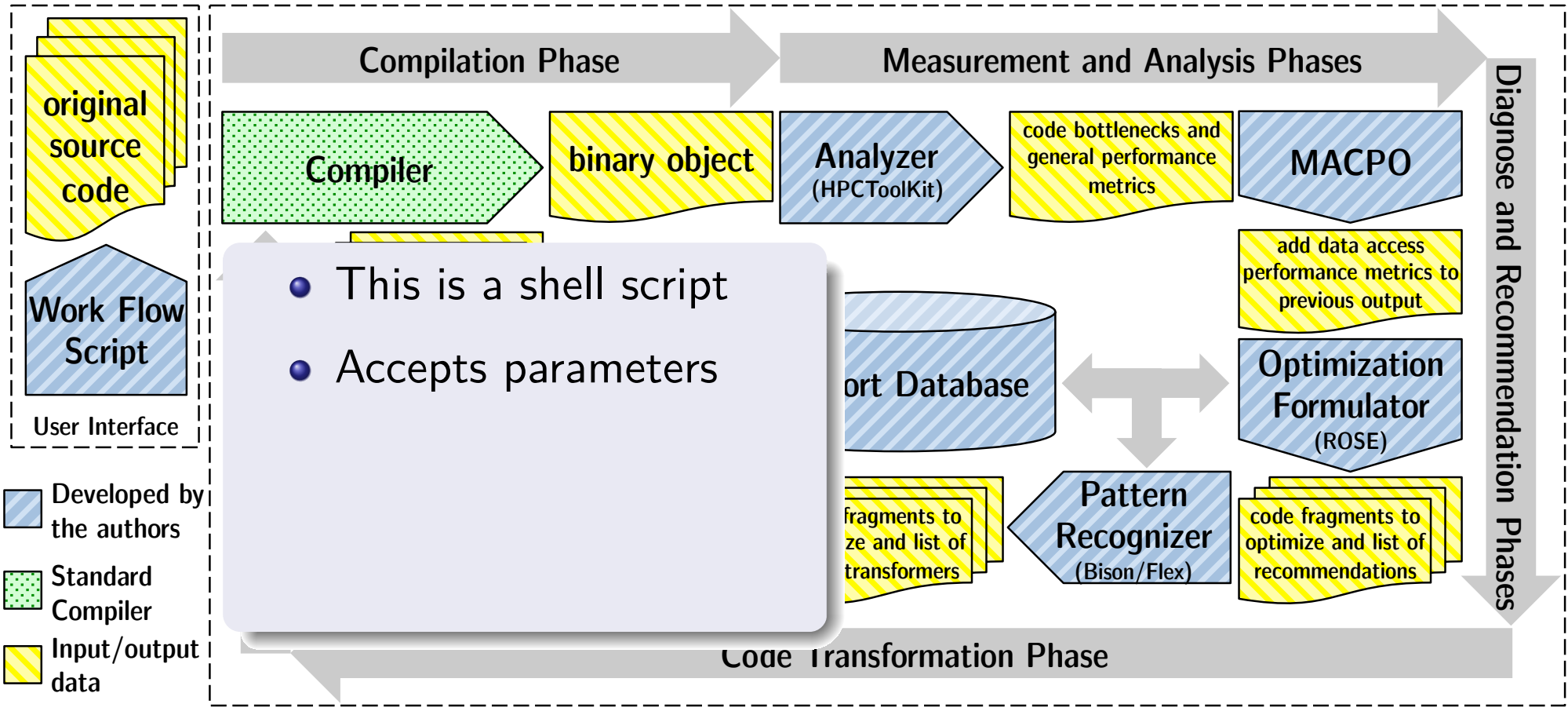


New Version: Work Flow Script

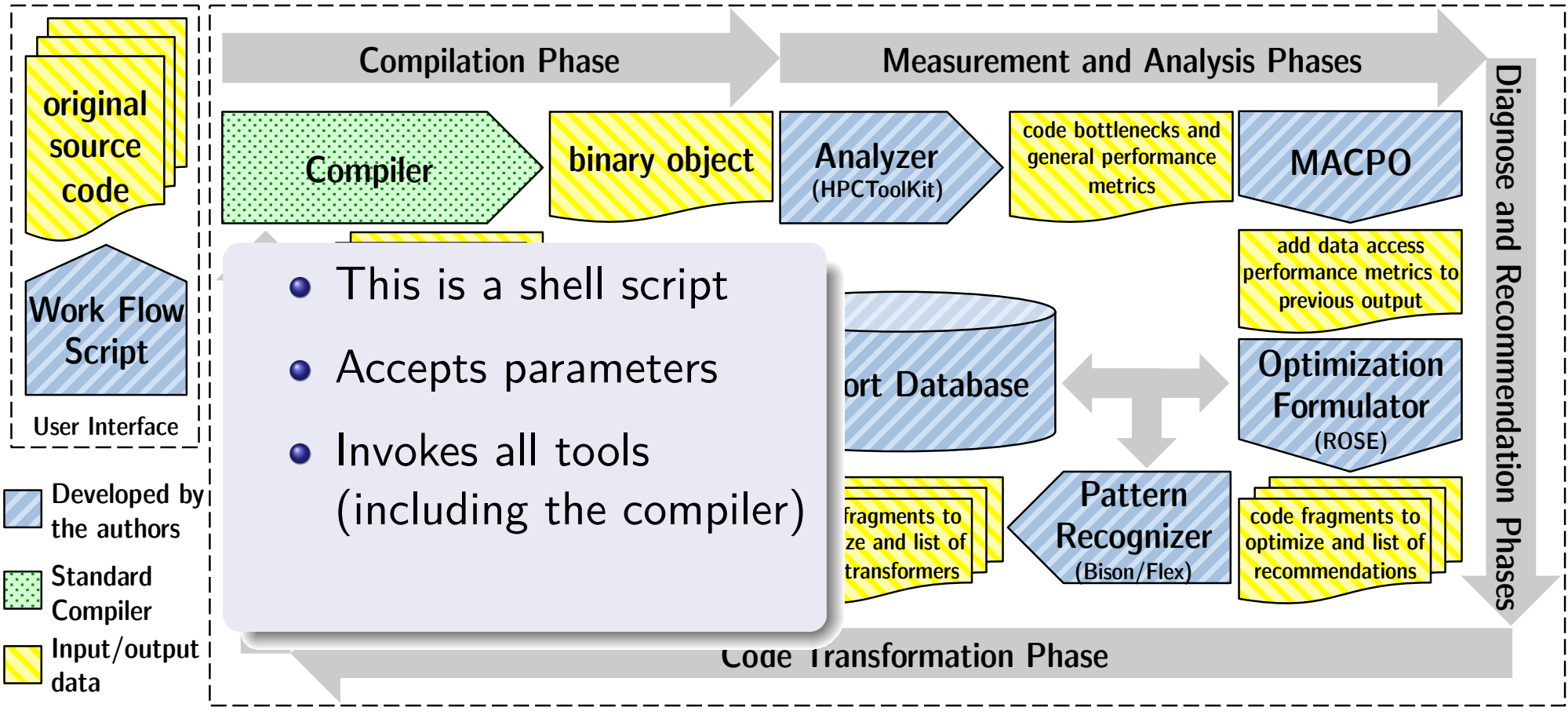


- This is a shell script

New Version: Work Flow Script

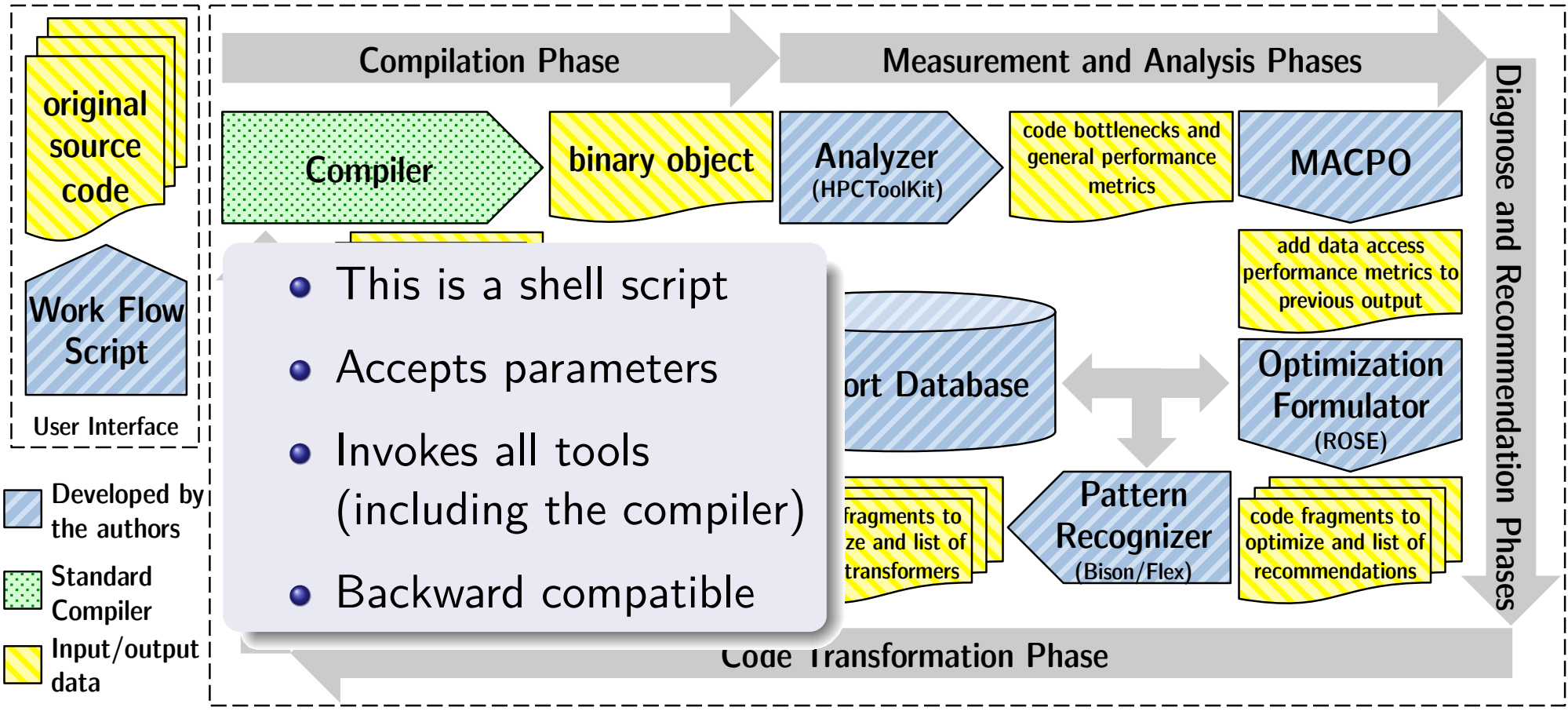


New Version: Work Flow Script



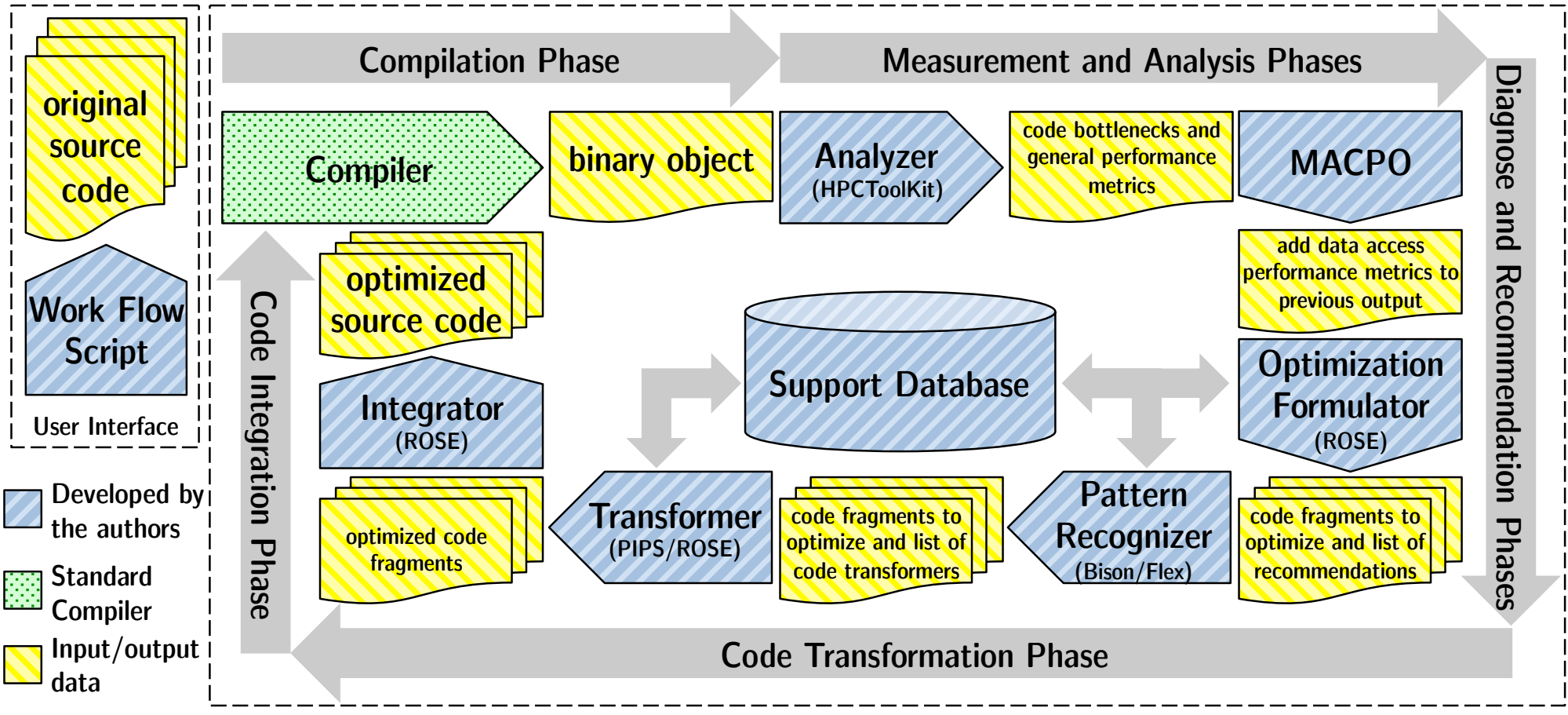
- This is a shell script
- Accepts parameters
- Invokes all tools (including the compiler)

New Version: Work Flow Script

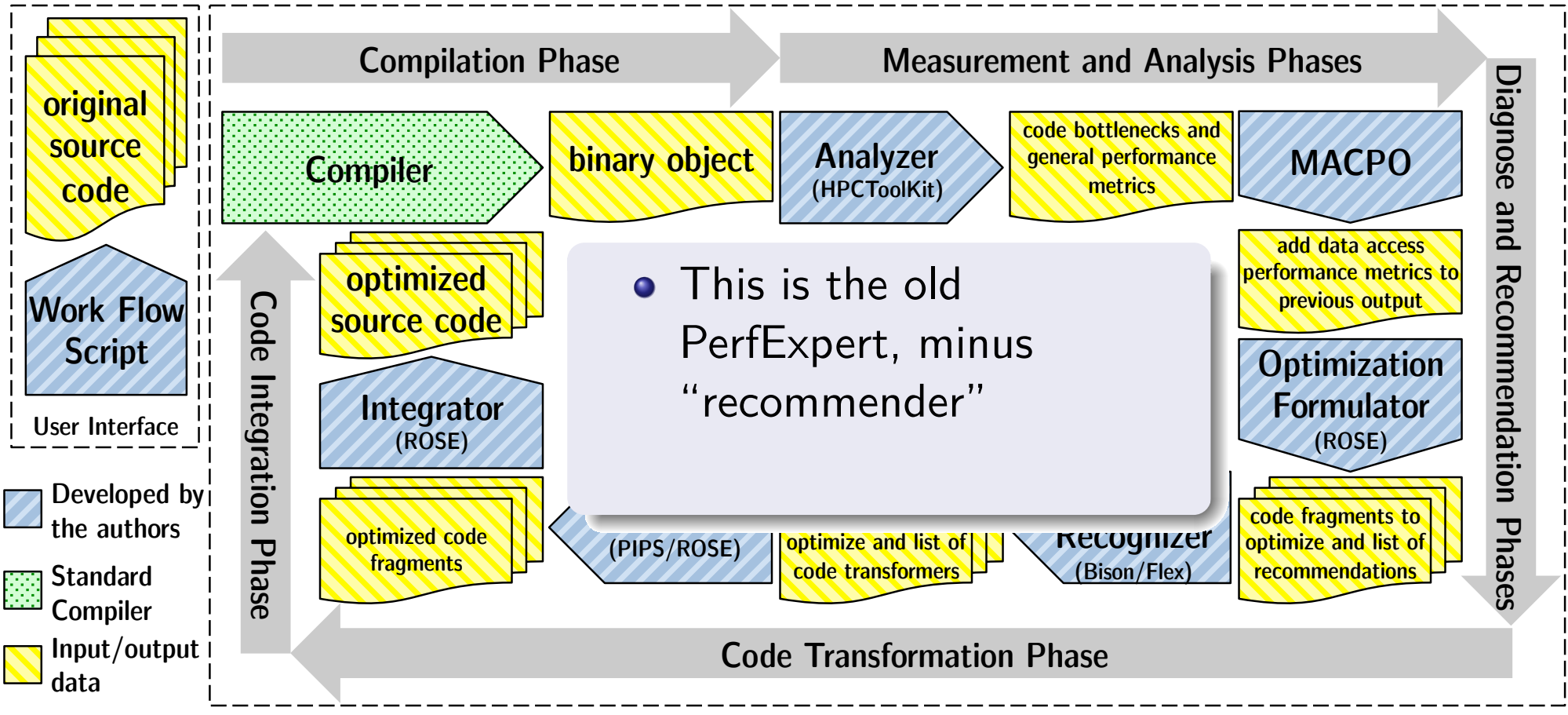


- This is a shell script
- Accepts parameters
- Invokes all tools (including the compiler)
- Backward compatible

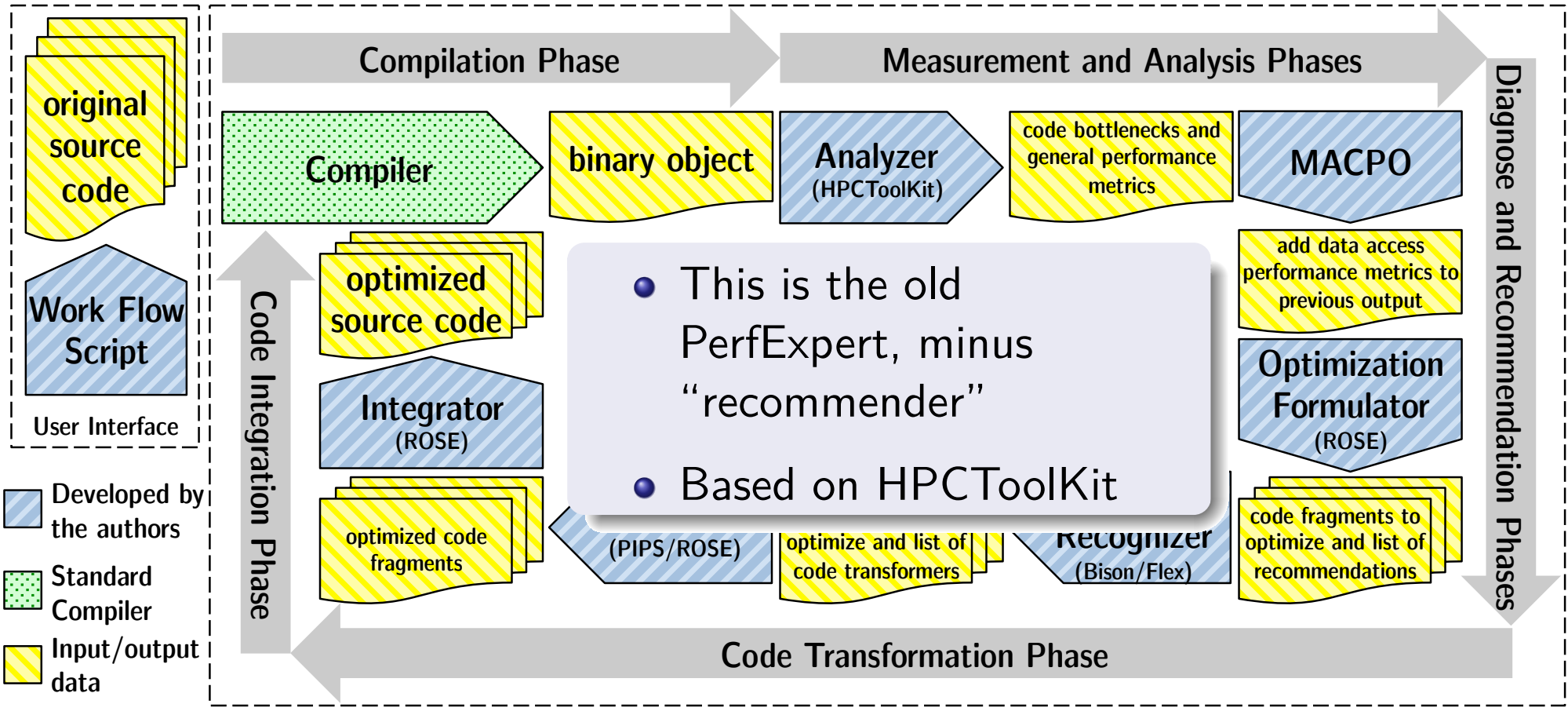
New Version: Analyzer



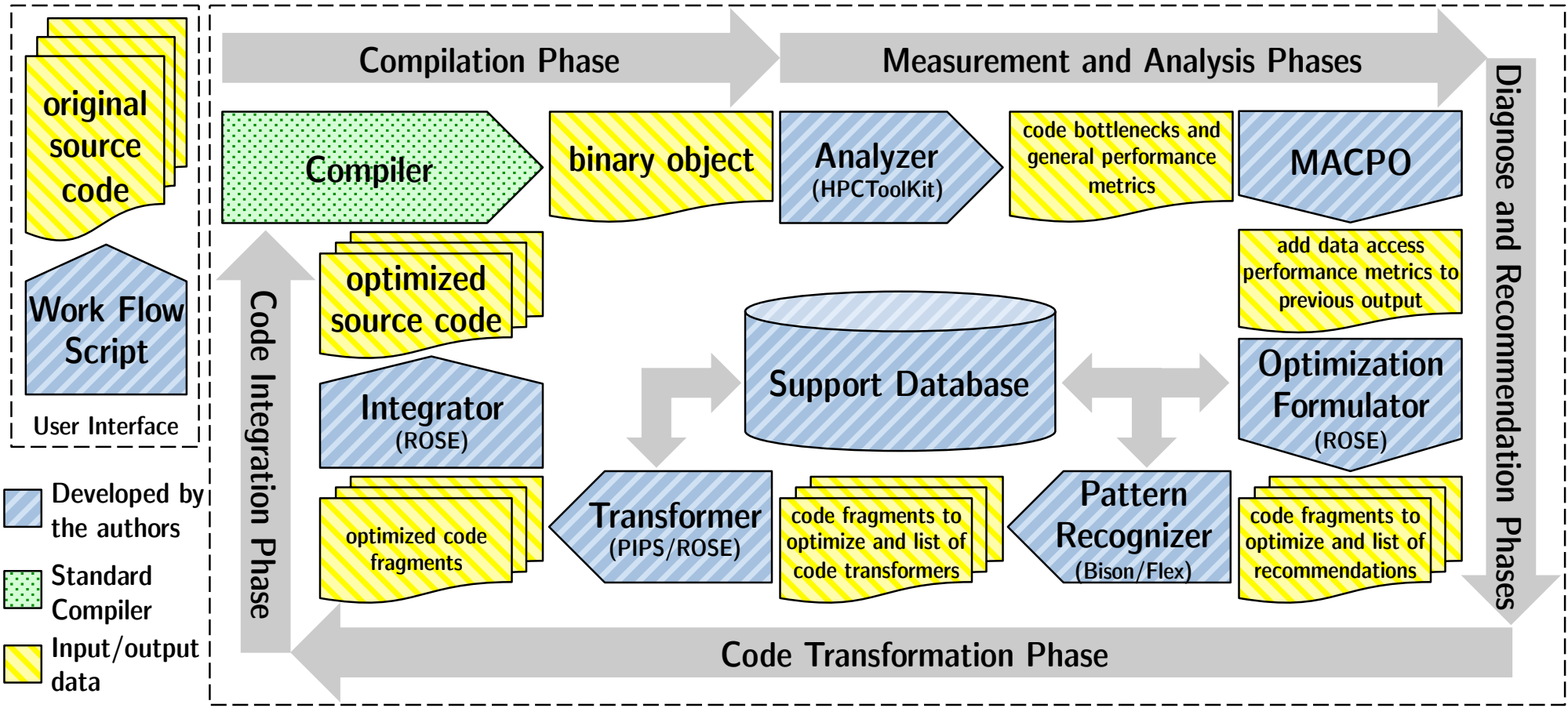
New Version: Analyzer



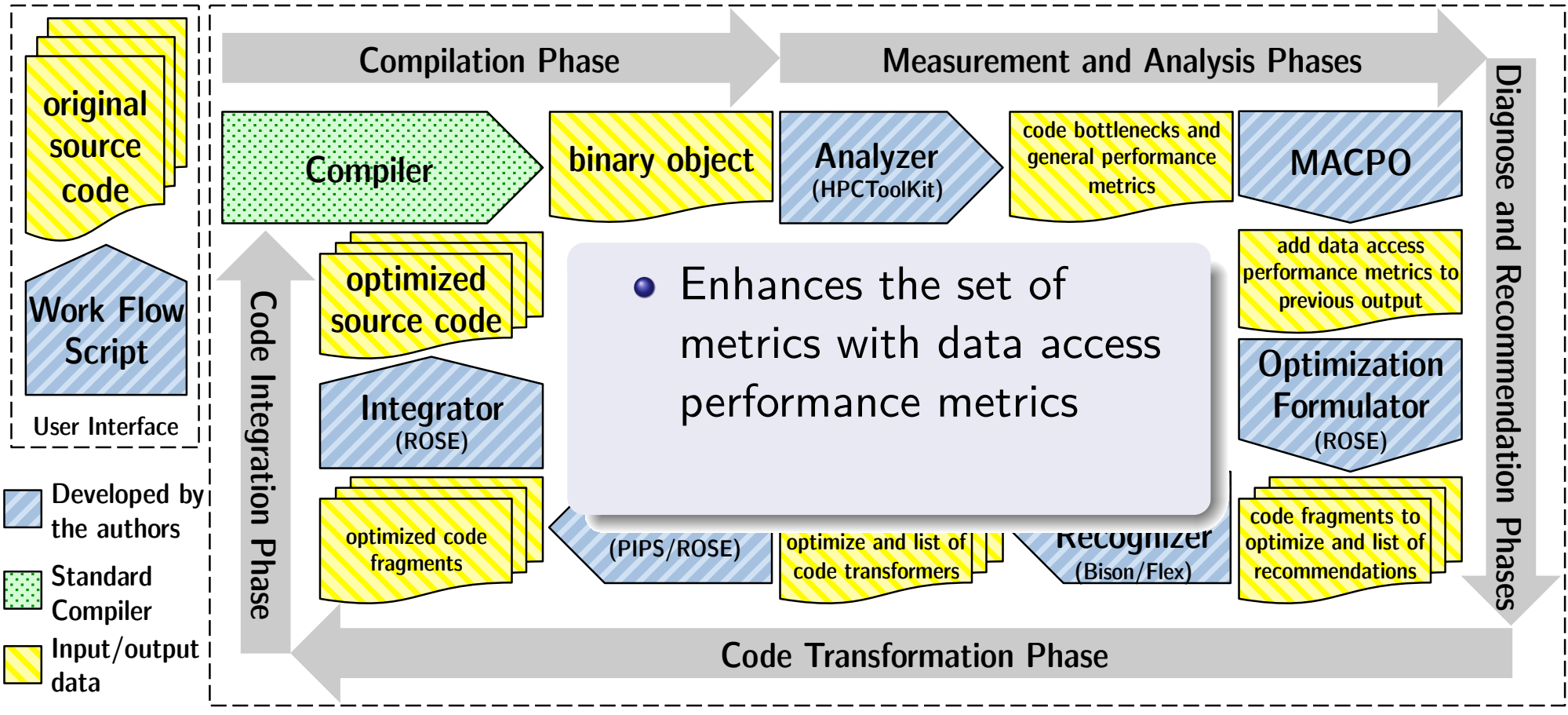
New Version: Analyzer



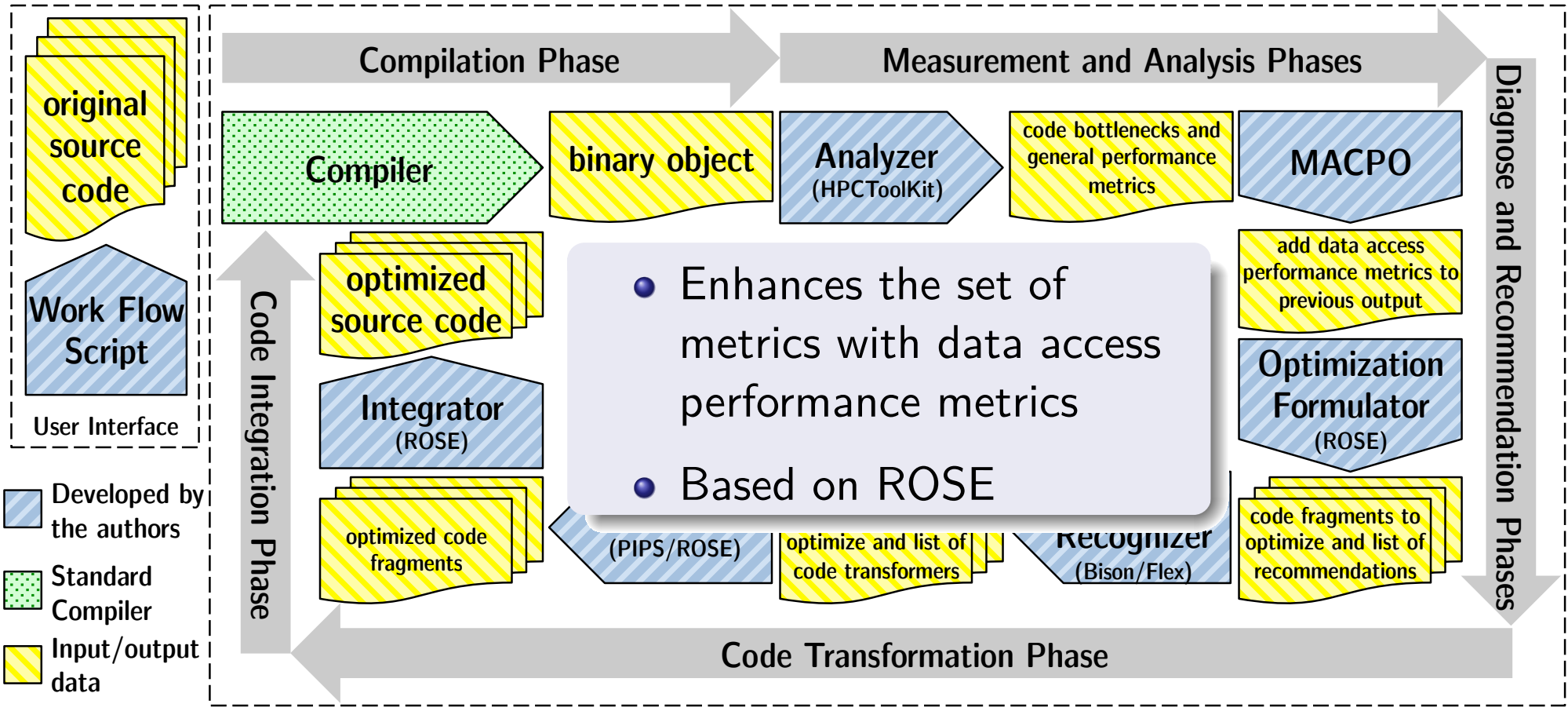
New Version: MACPO



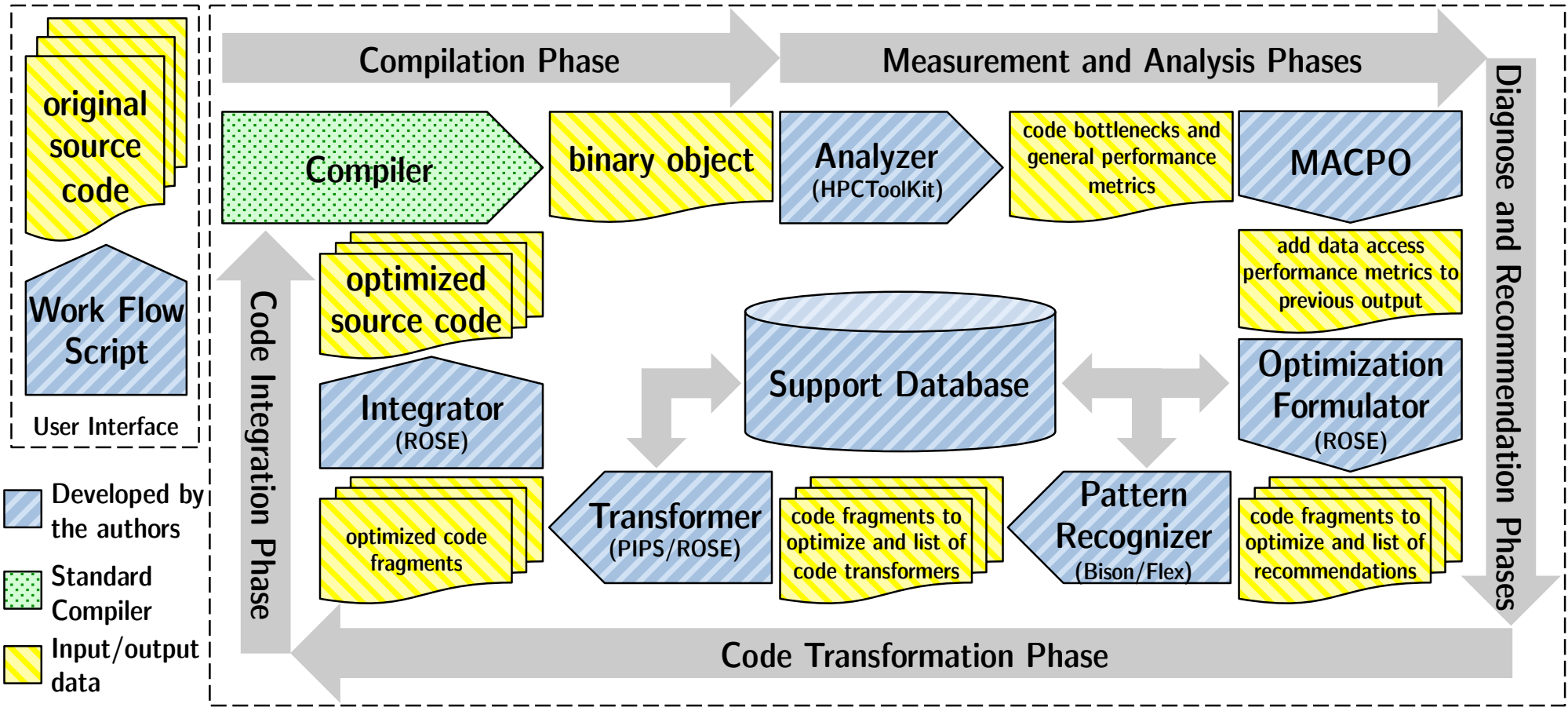
New Version: MACPO



New Version: MACPO

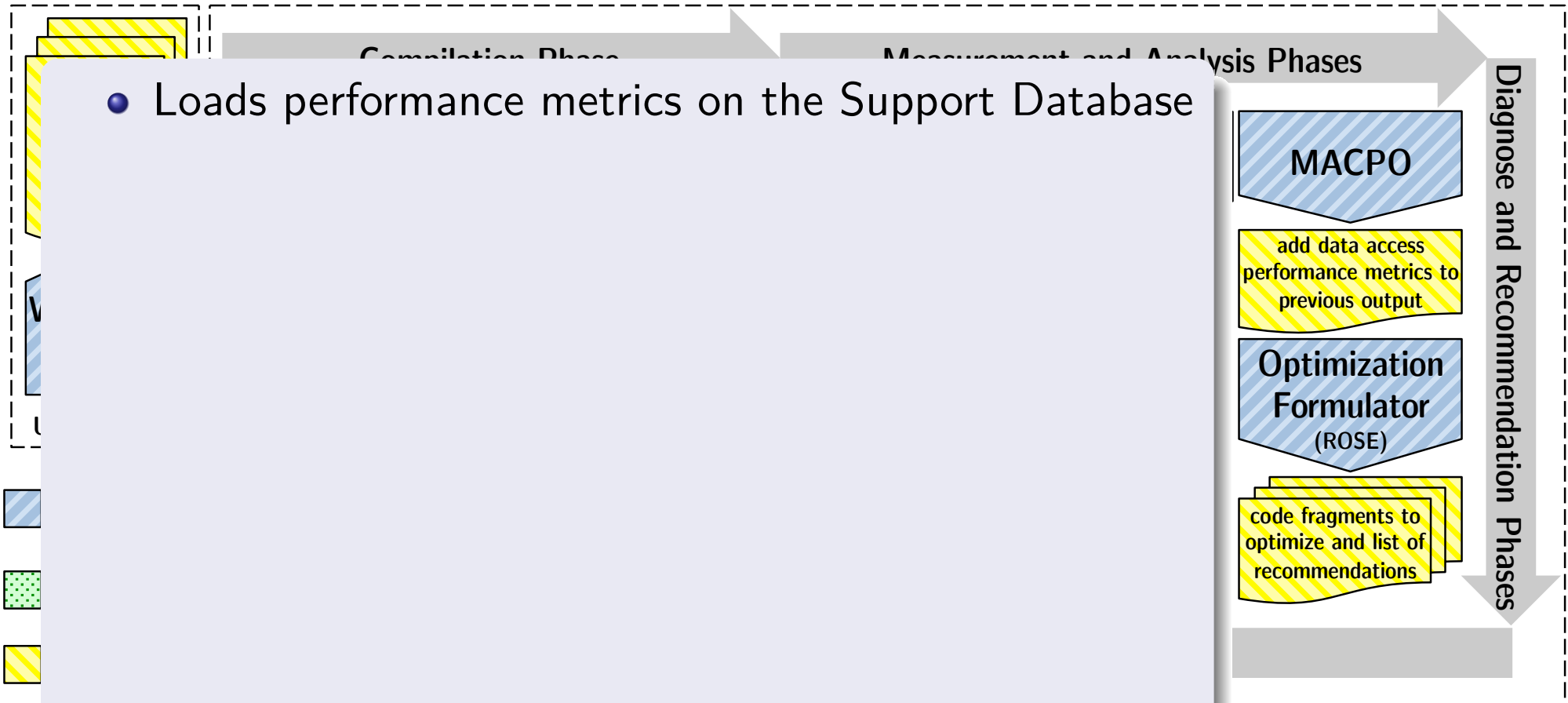


New Version: Optimization Formulator



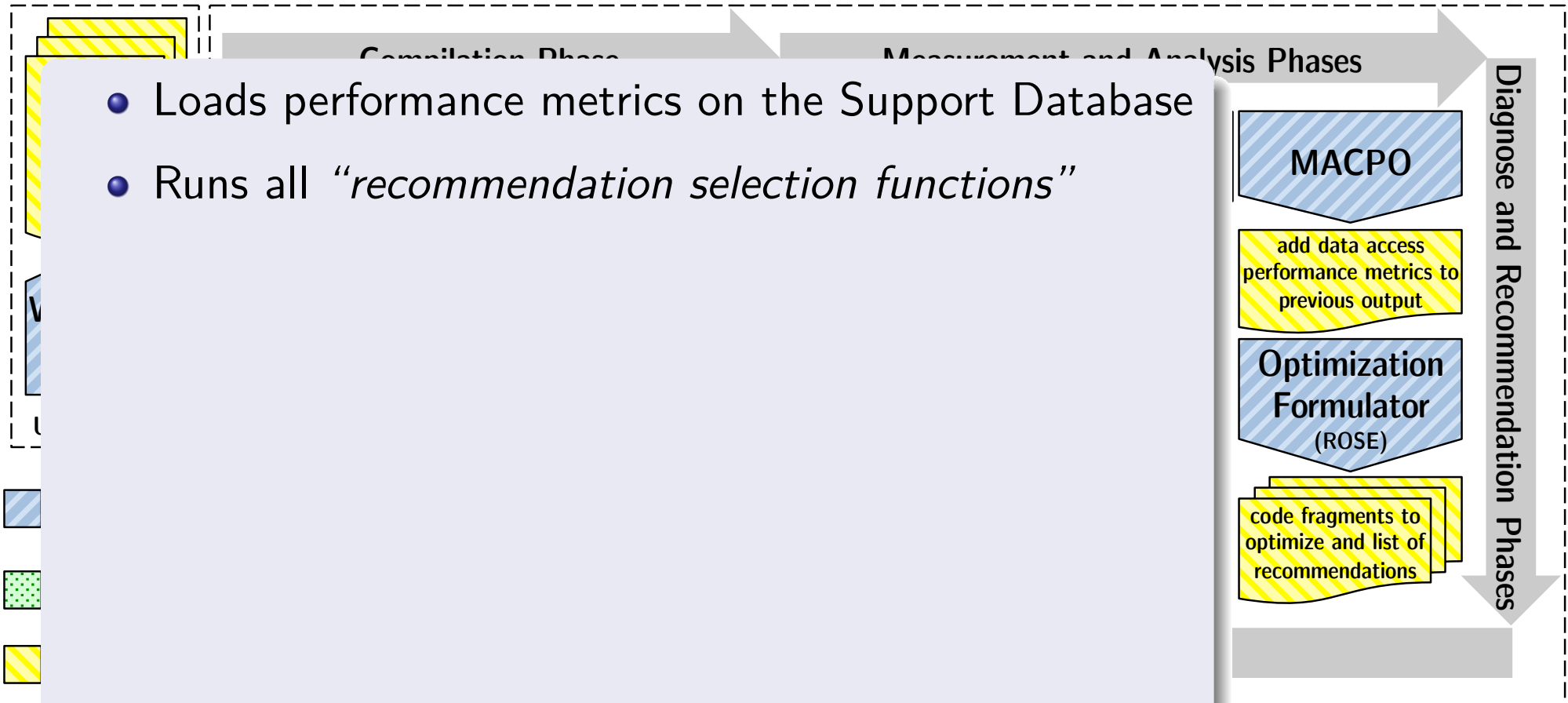
New Version: Optimization Formulator

- Loads performance metrics on the Support Database



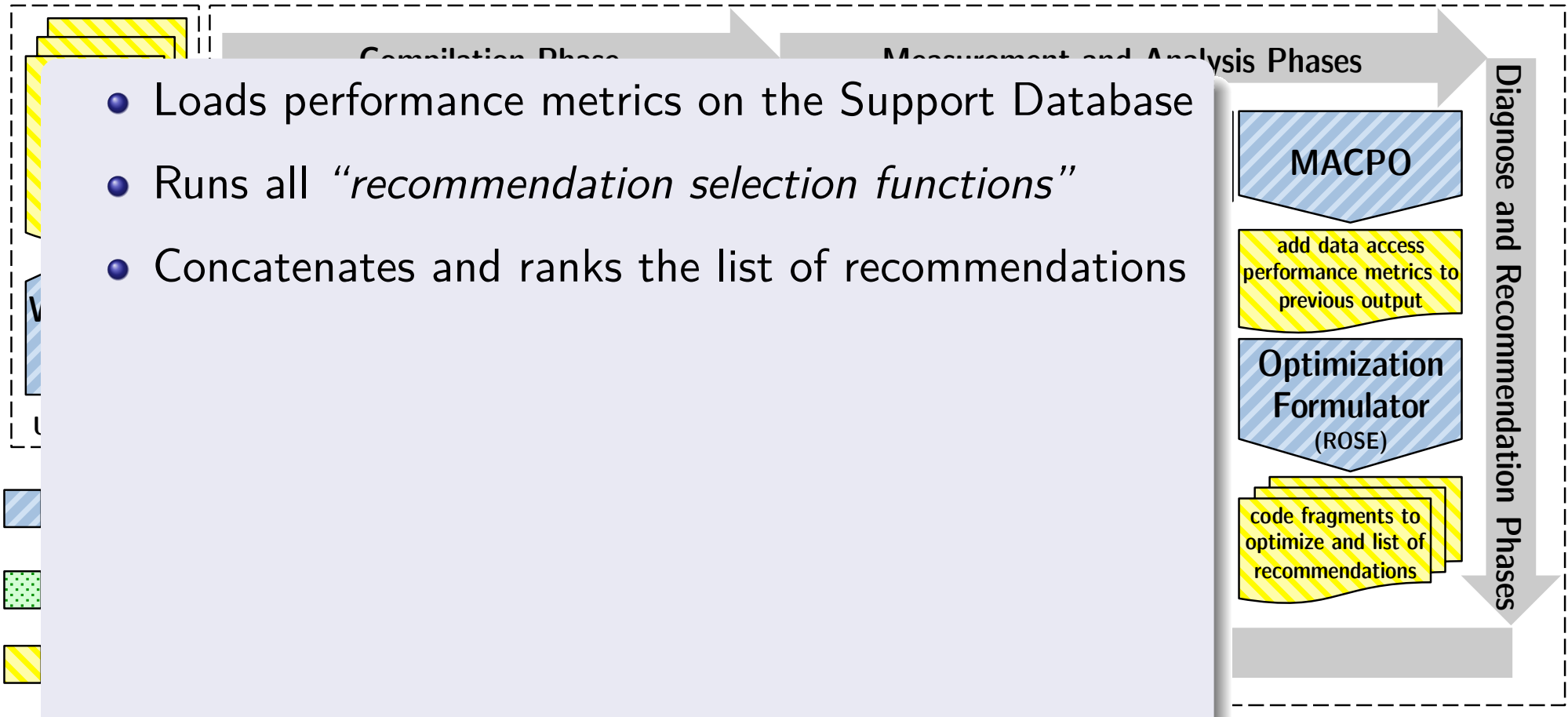
New Version: Optimization Formulator

- Loads performance metrics on the Support Database
- Runs all *"recommendation selection functions"*



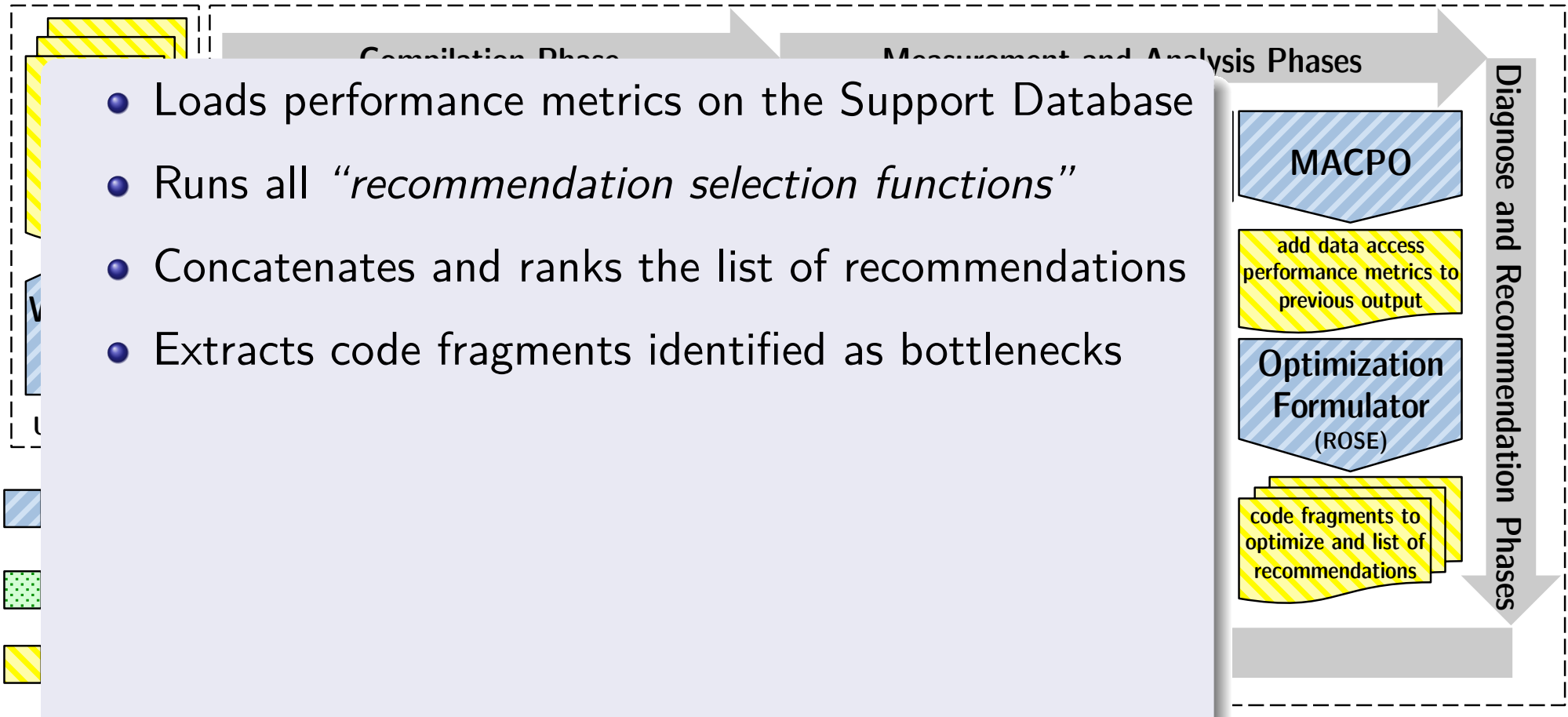
New Version: Optimization Formulator

- Loads performance metrics on the Support Database
- Runs all *"recommendation selection functions"*
- Concatenates and ranks the list of recommendations



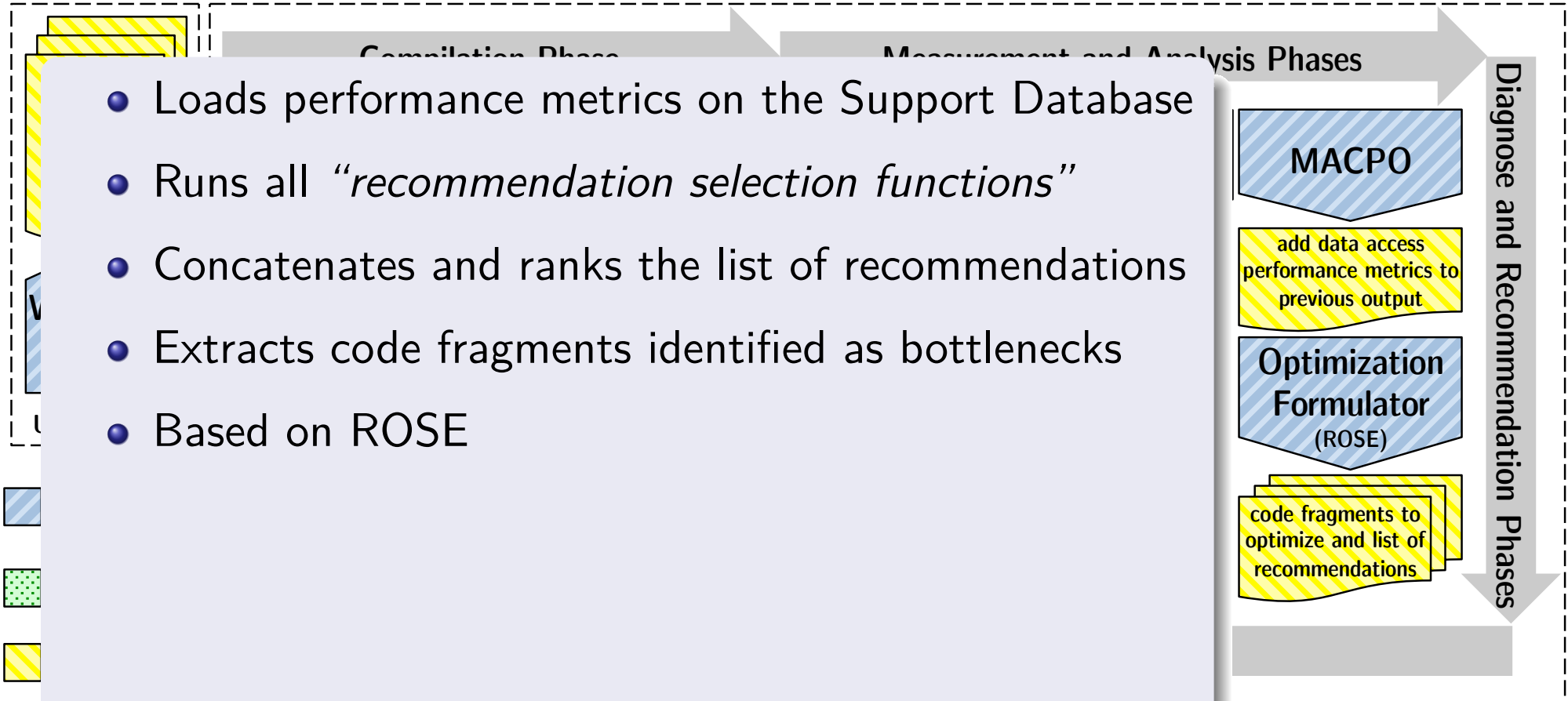
New Version: Optimization Formulator

- Loads performance metrics on the Support Database
- Runs all *"recommendation selection functions"*
- Concatenates and ranks the list of recommendations
- Extracts code fragments identified as bottlenecks



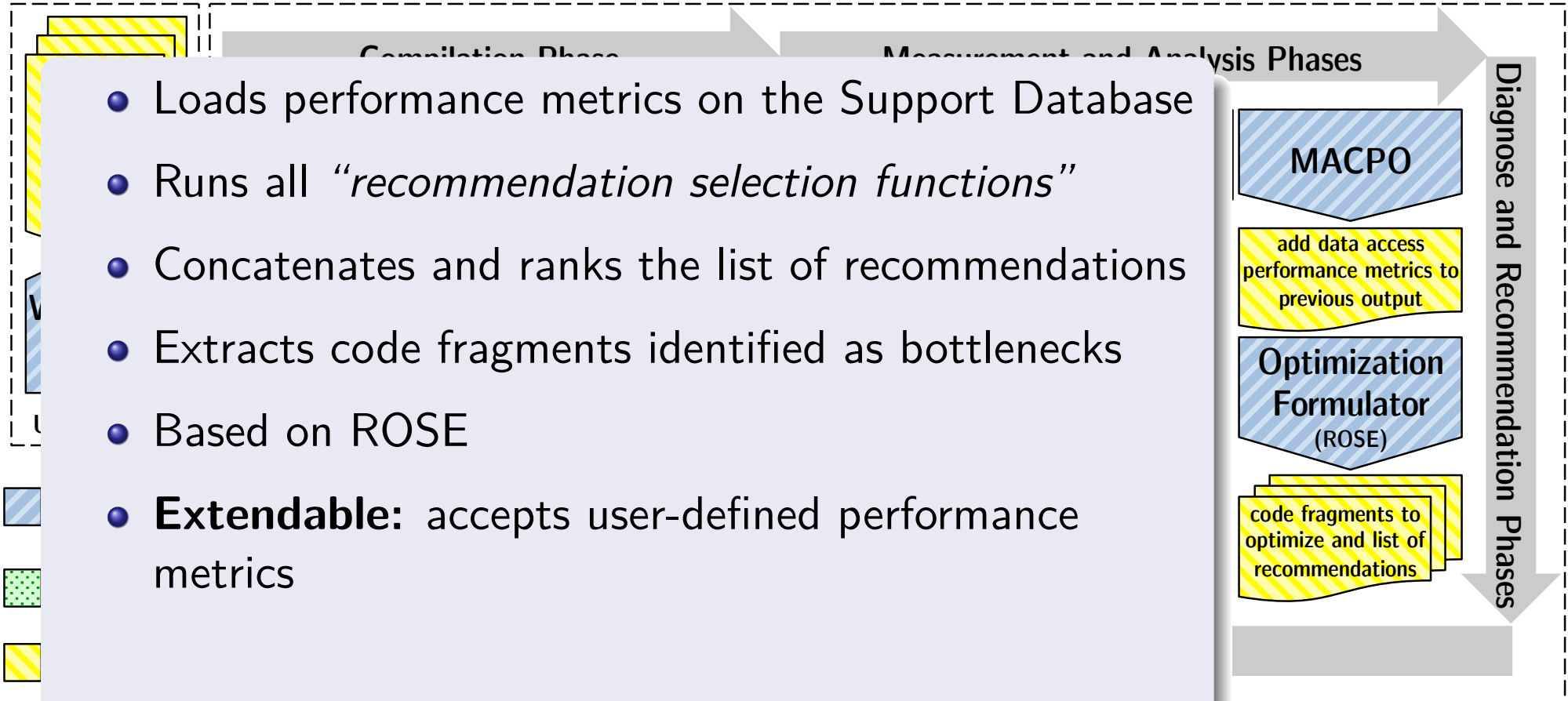
New Version: Optimization Formulator

- Loads performance metrics on the Support Database
- Runs all *"recommendation selection functions"*
- Concatenates and ranks the list of recommendations
- Extracts code fragments identified as bottlenecks
- Based on ROSE

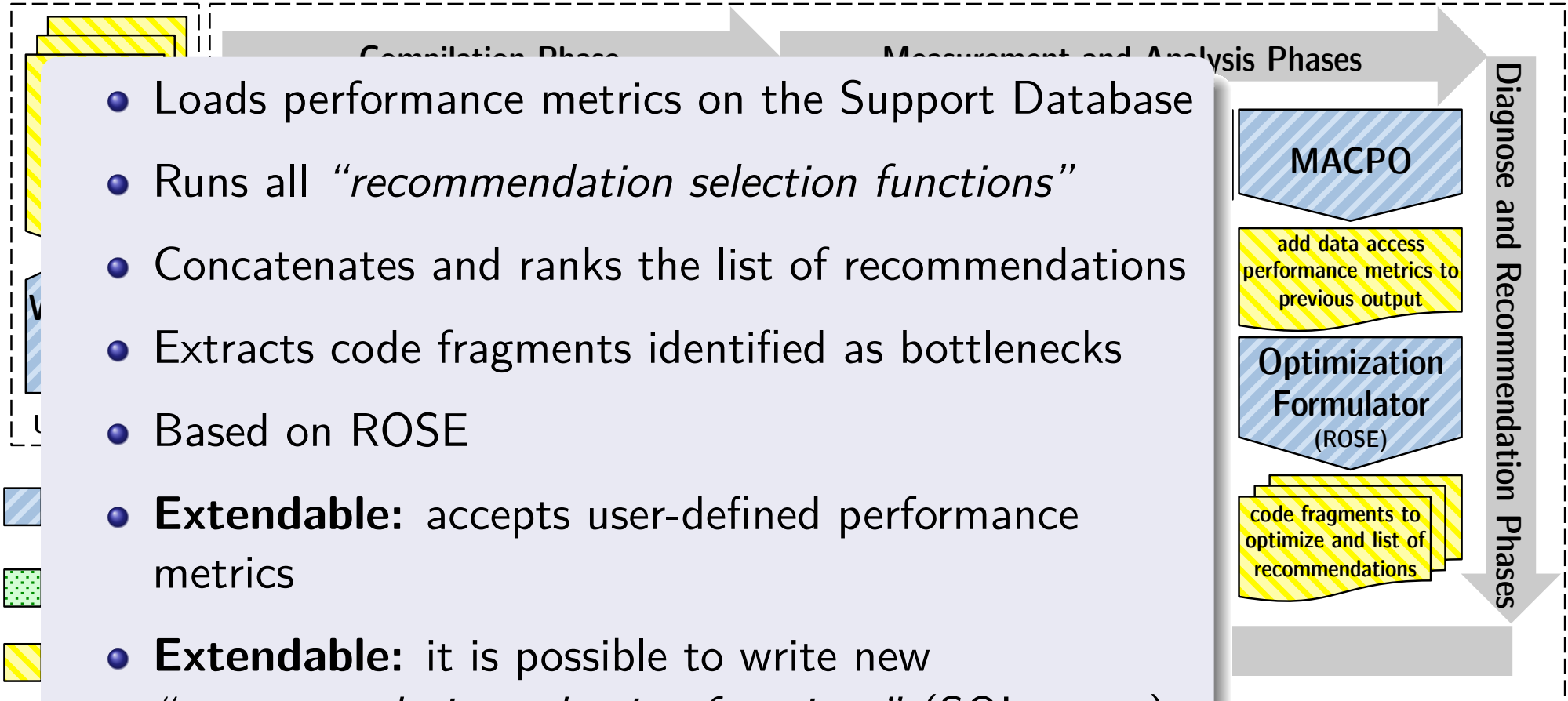


New Version: Optimization Formulator

- Loads performance metrics on the Support Database
- Runs all *"recommendation selection functions"*
- Concatenates and ranks the list of recommendations
- Extracts code fragments identified as bottlenecks
- Based on ROSE
- **Extendable:** accepts user-defined performance metrics

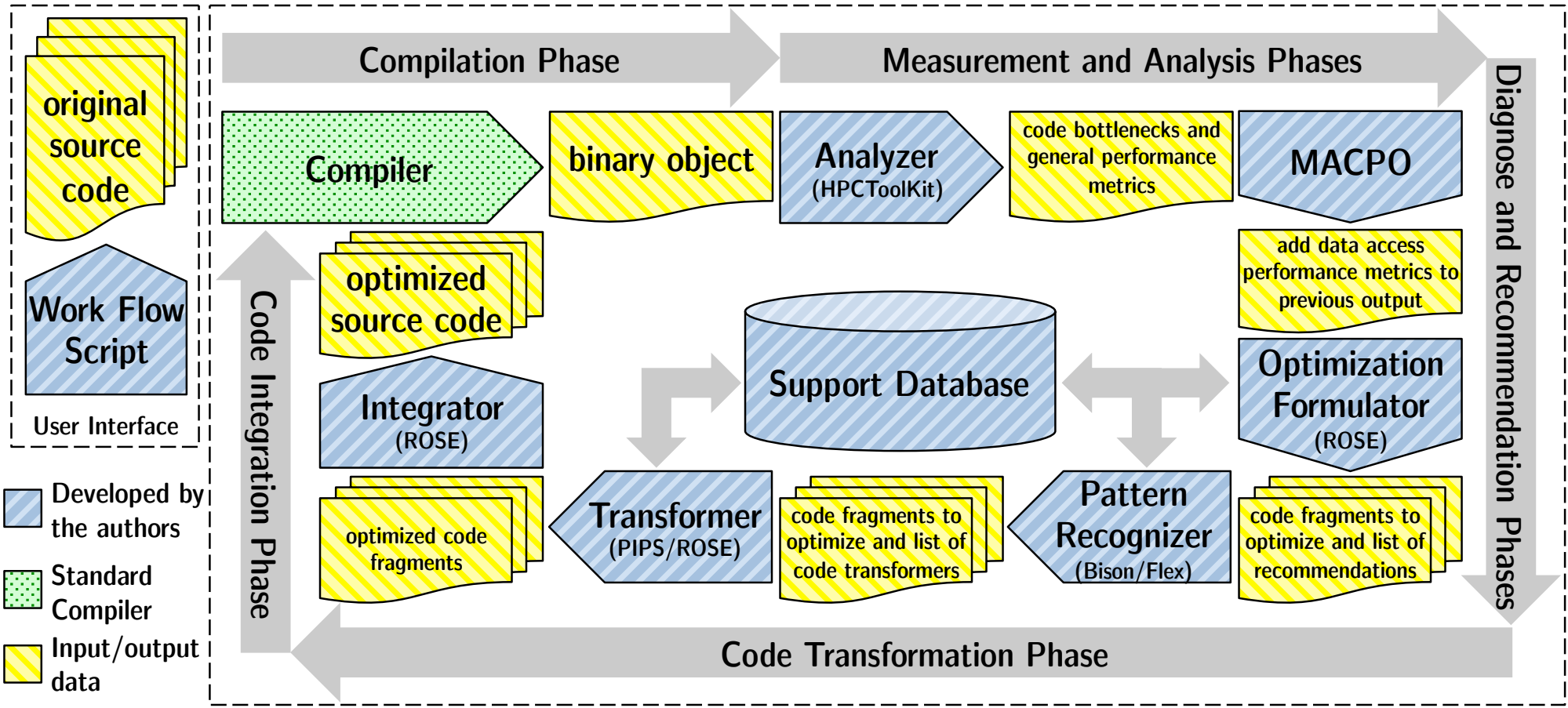


New Version: Optimization Formulator



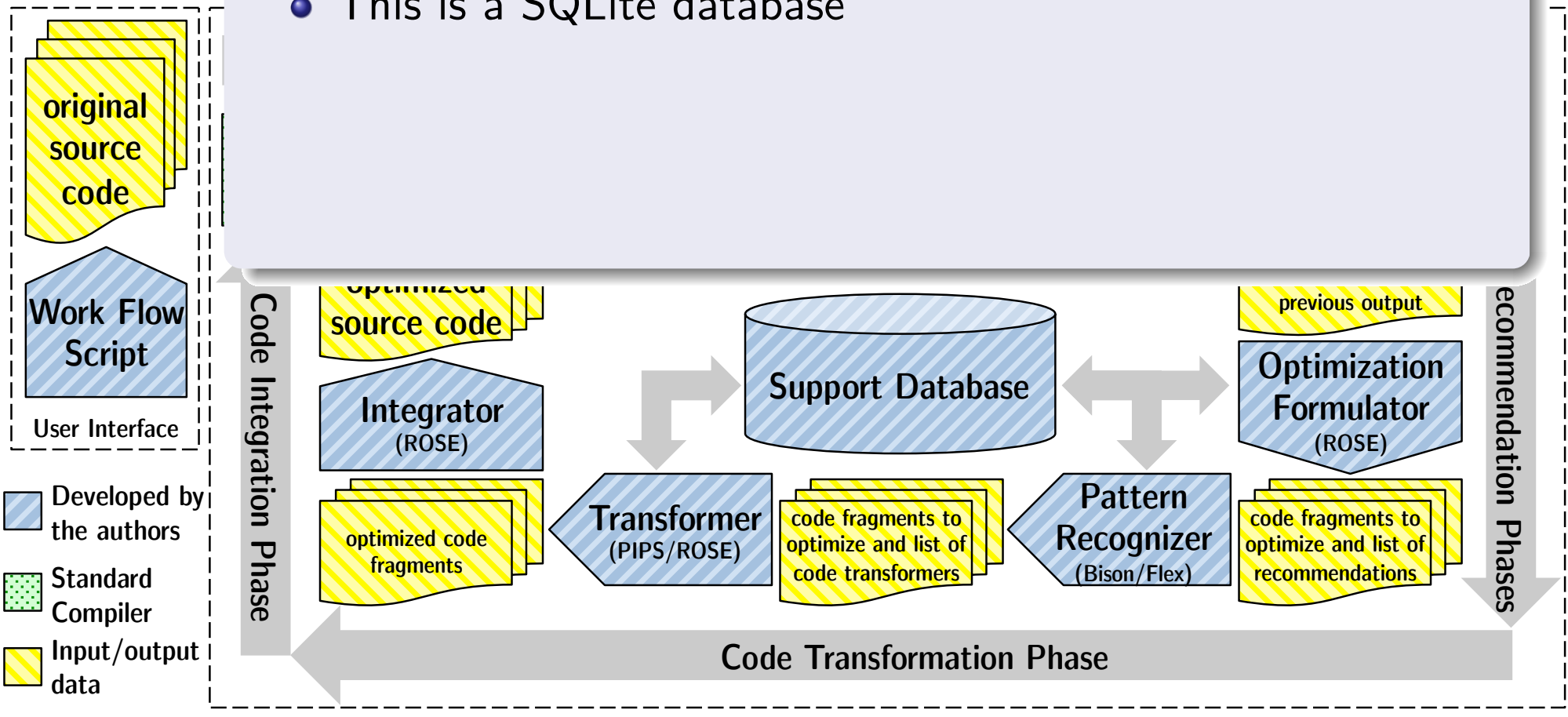
- Loads performance metrics on the Support Database
- Runs all *"recommendation selection functions"*
- Concatenates and ranks the list of recommendations
- Extracts code fragments identified as bottlenecks
- Based on ROSE
- **Extendable:** accepts user-defined performance metrics
- **Extendable:** it is possible to write new *"recommendation selection functions"* (SQL query)

New Version: Support Database



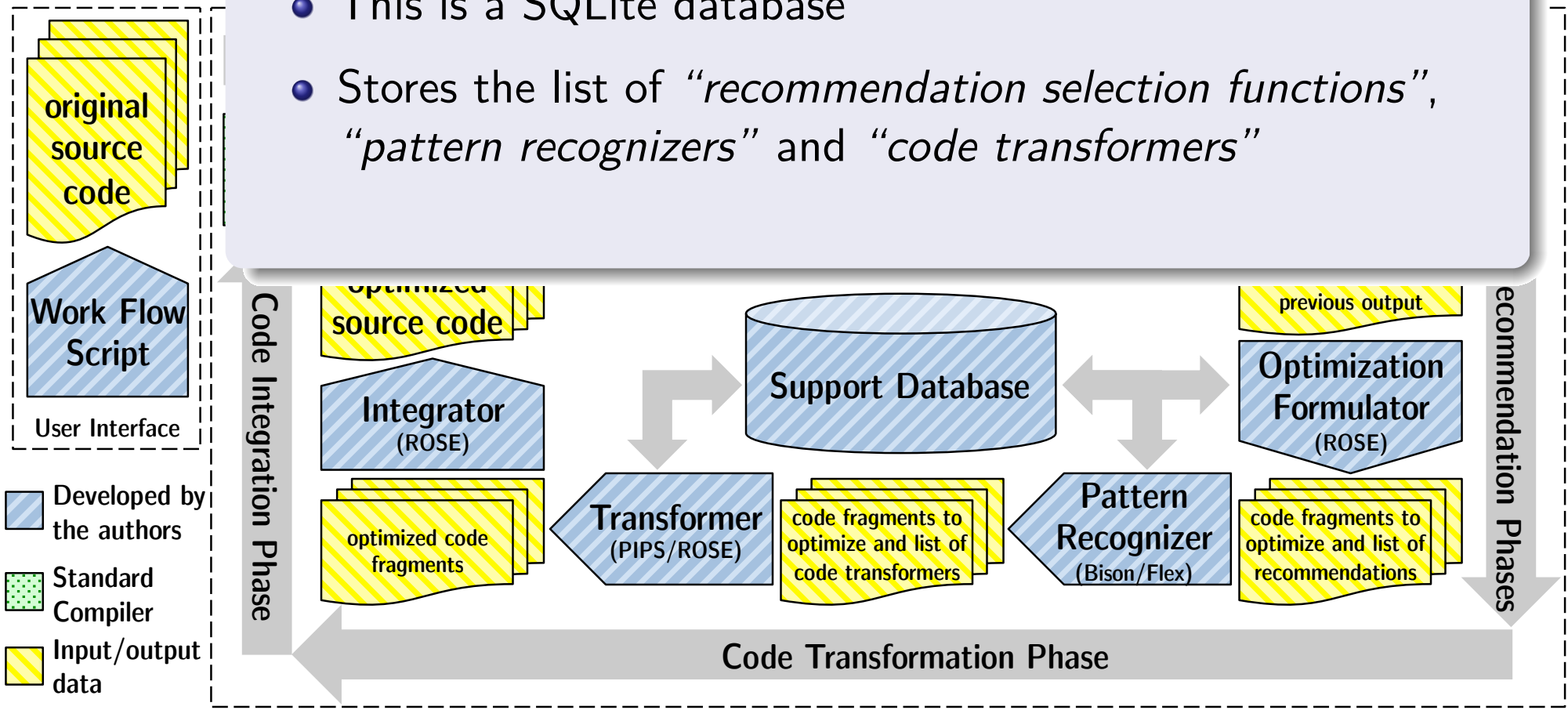
New Version: Support Database

- This is a SQLite database



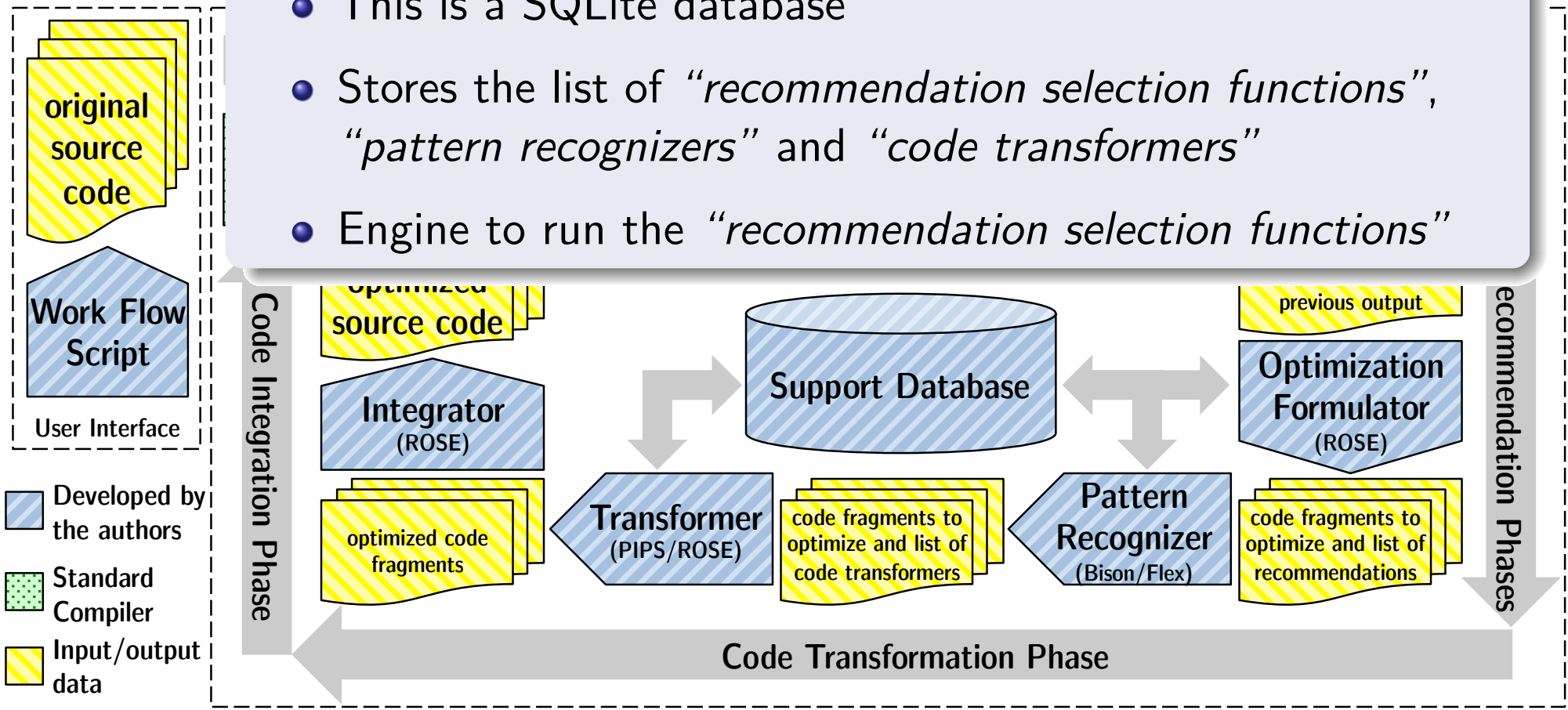
New Version: Support Database

- This is a SQLite database
- Stores the list of *"recommendation selection functions"*, *"pattern recognizers"* and *"code transformers"*

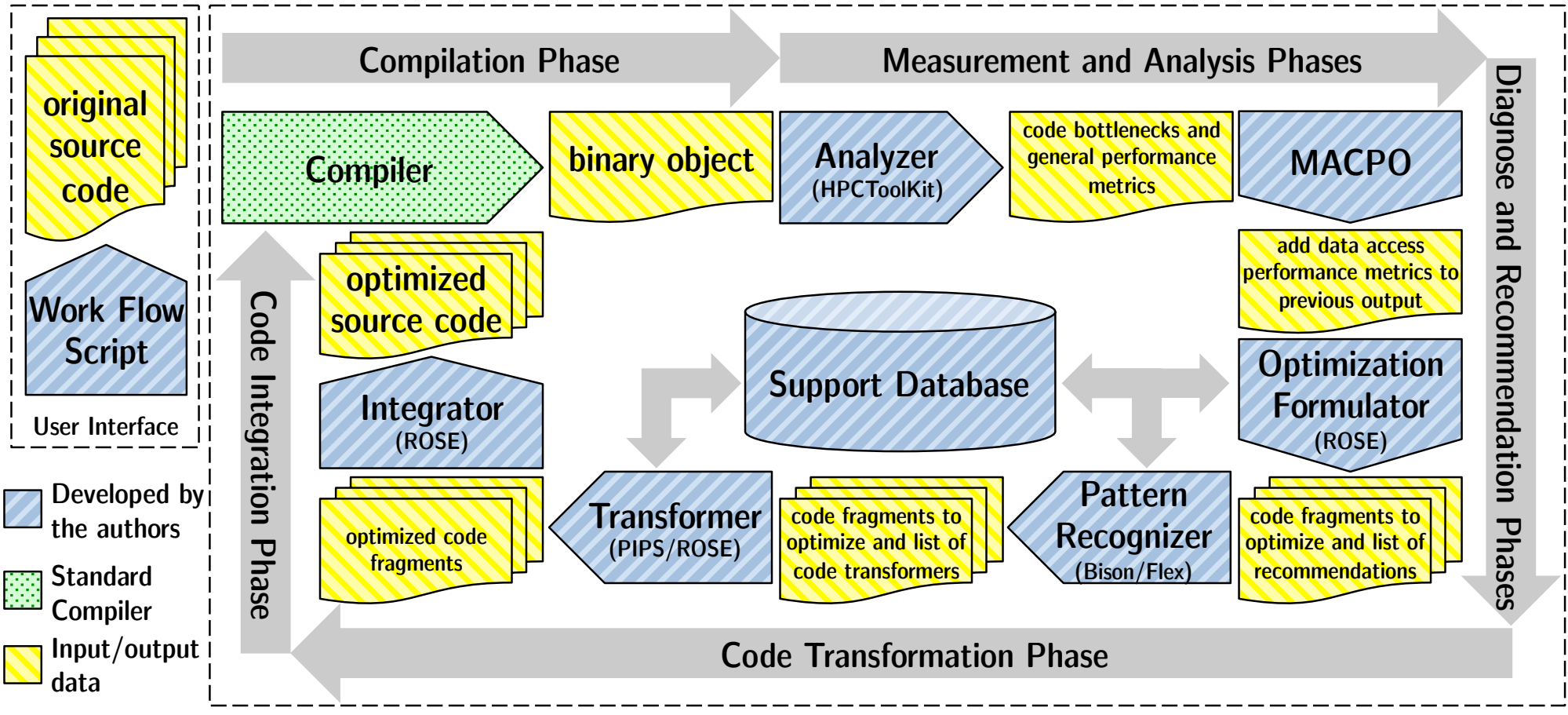


New Version: Support Database

- This is a SQLite database
- Stores the list of *"recommendation selection functions"*, *"pattern recognizers"* and *"code transformers"*
- Engine to run the *"recommendation selection functions"*

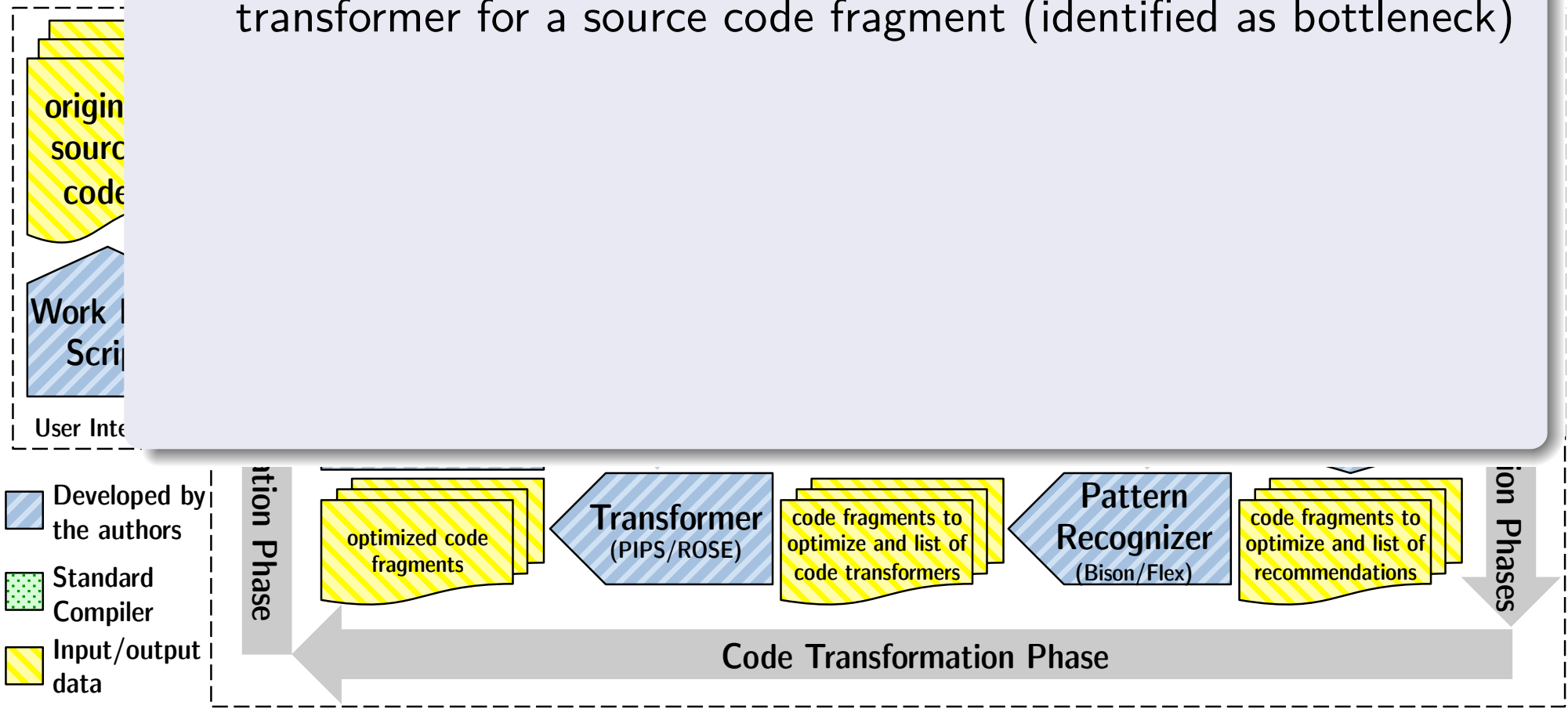


New Version: Pattern Recognizer



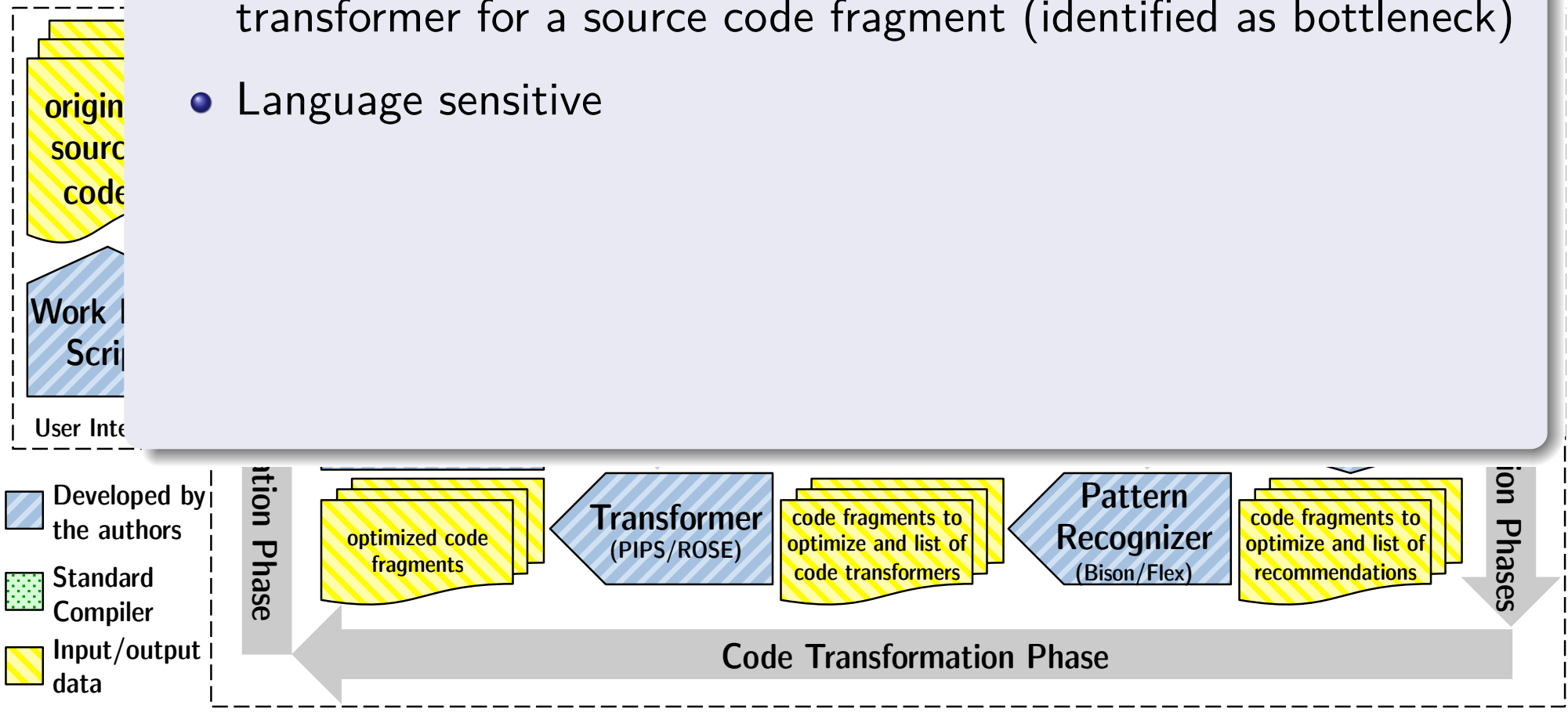
New Version: Pattern Recognizer

- Acts as a "filter" trying to find (match) the right code transformer for a source code fragment (identified as bottleneck)



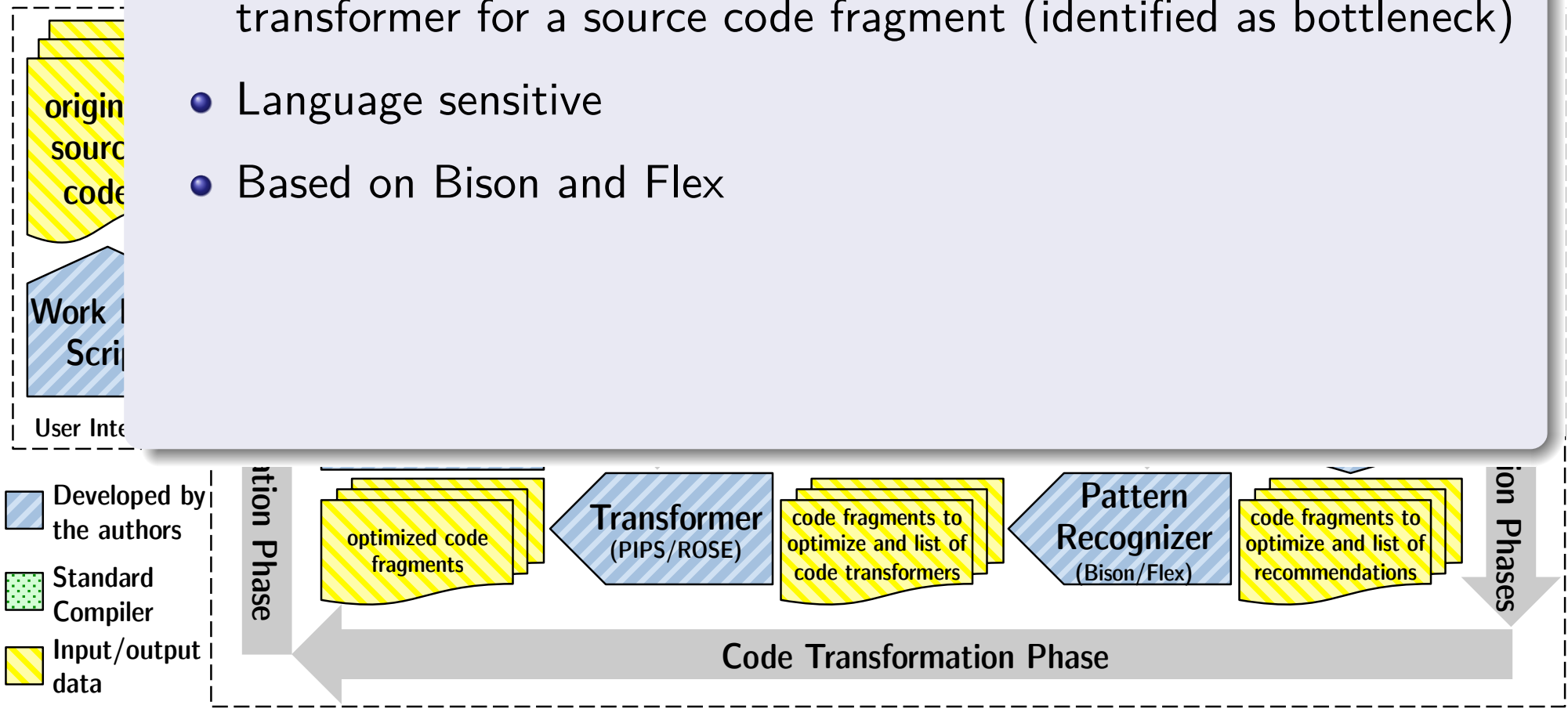
New Version: Pattern Recognizer

- Acts as a "filter" trying to find (match) the right code transformer for a source code fragment (identified as bottleneck)
- Language sensitive



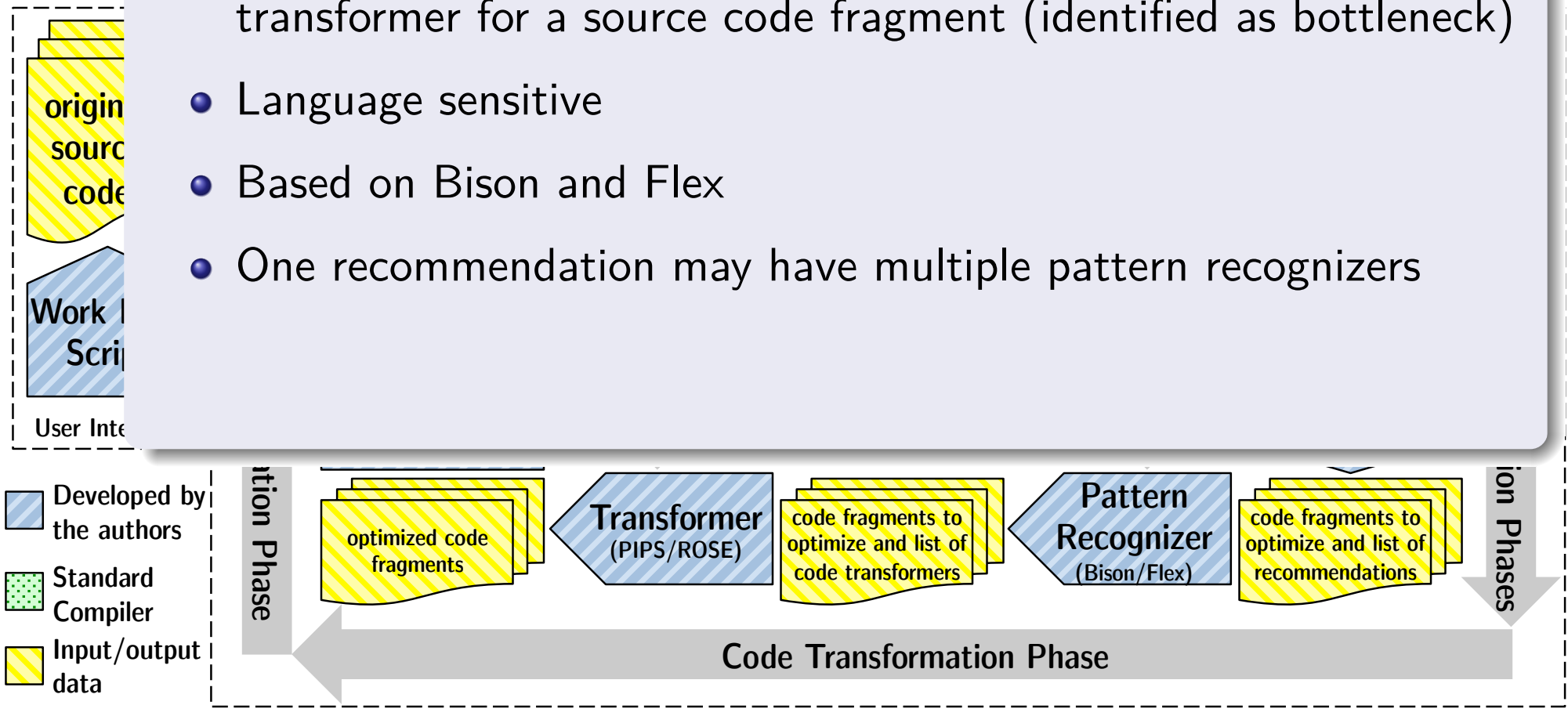
New Version: Pattern Recognizer

- Acts as a "filter" trying to find (match) the right code transformer for a source code fragment (identified as bottleneck)
- Language sensitive
- Based on Bison and Flex



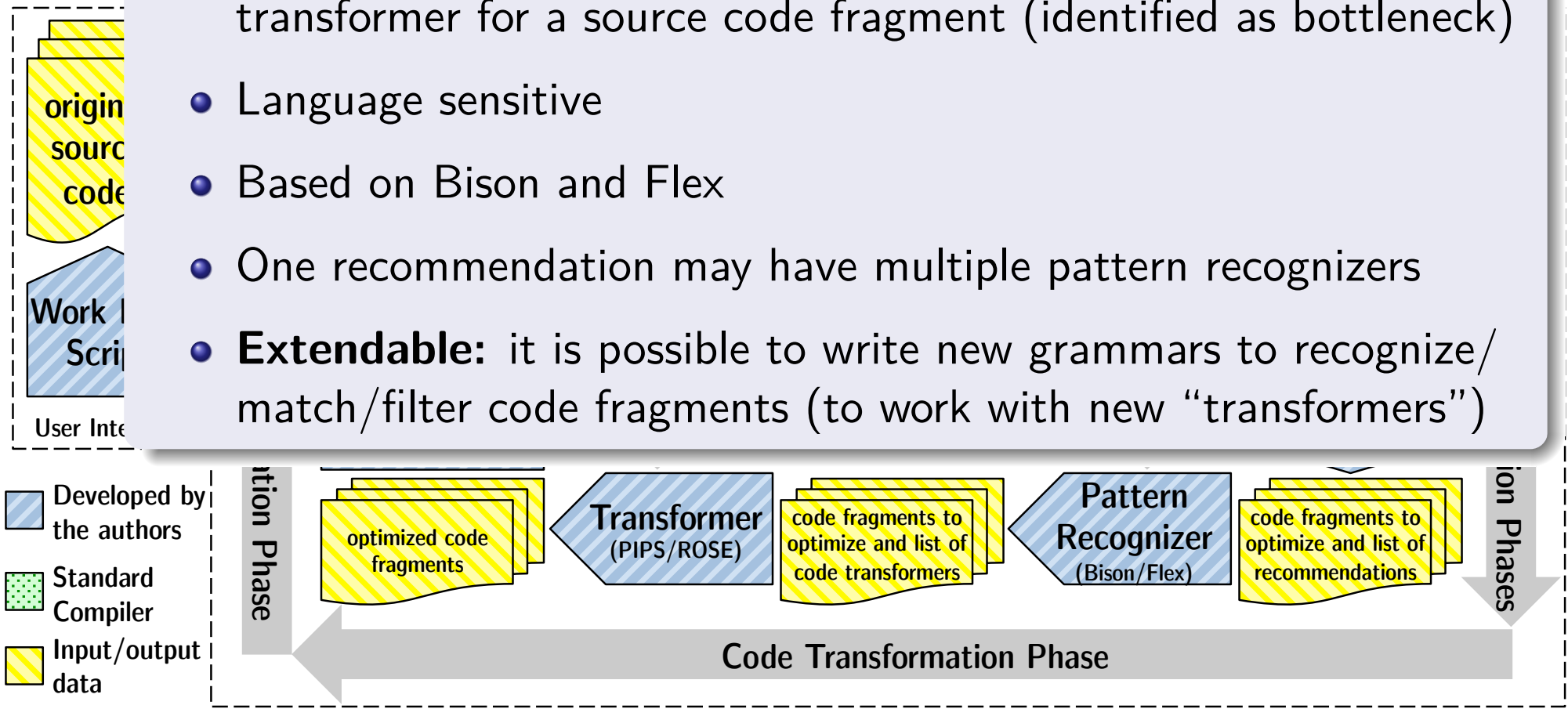
New Version: Pattern Recognizer

- Acts as a "filter" trying to find (match) the right code transformer for a source code fragment (identified as bottleneck)
- Language sensitive
- Based on Bison and Flex
- One recommendation may have multiple pattern recognizers

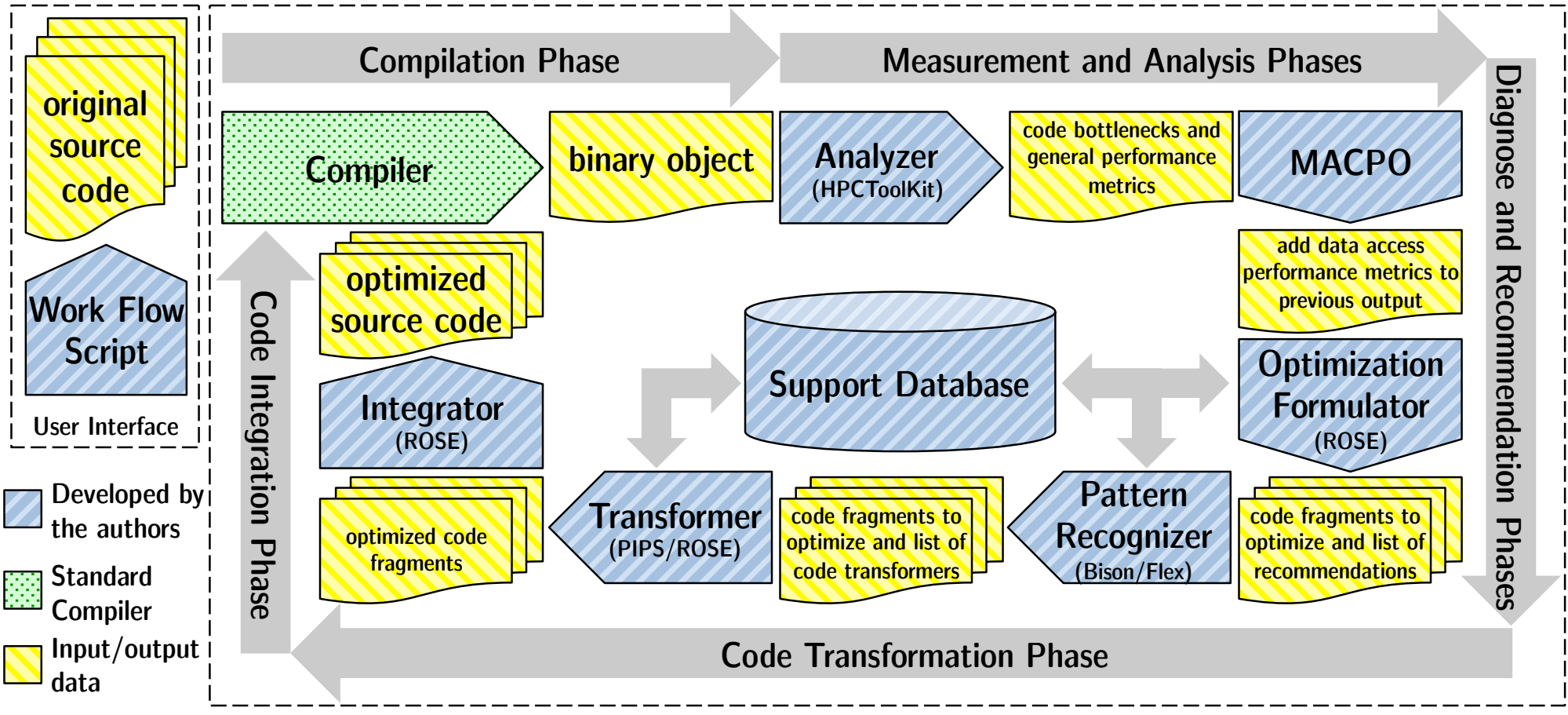


New Version: Pattern Recognizer

- Acts as a "filter" trying to find (match) the right code transformer for a source code fragment (identified as bottleneck)
- Language sensitive
- Based on Bison and Flex
- One recommendation may have multiple pattern recognizers
- **Extendable:** it is possible to write new grammars to recognize/match/filter code fragments (to work with new "transformers")

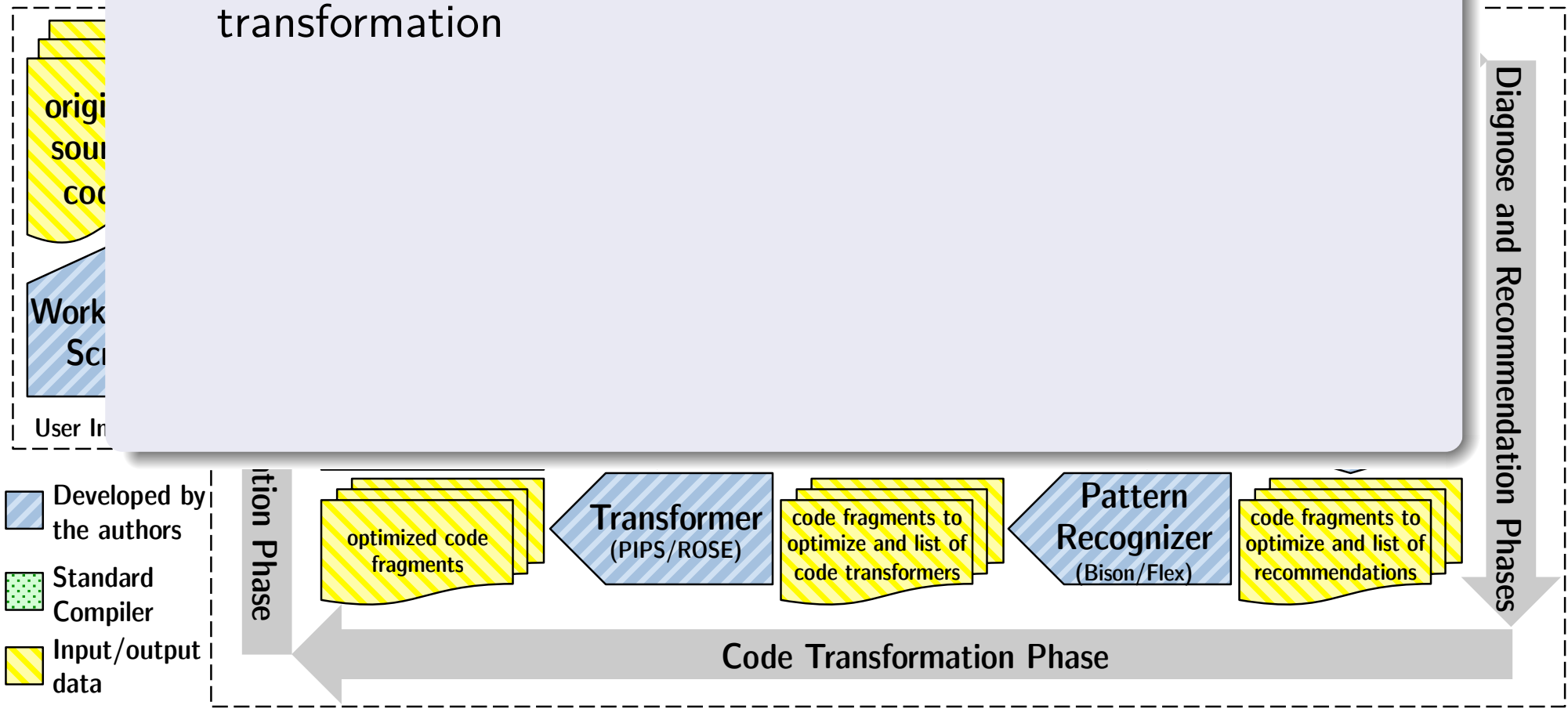


New Version: Transformer



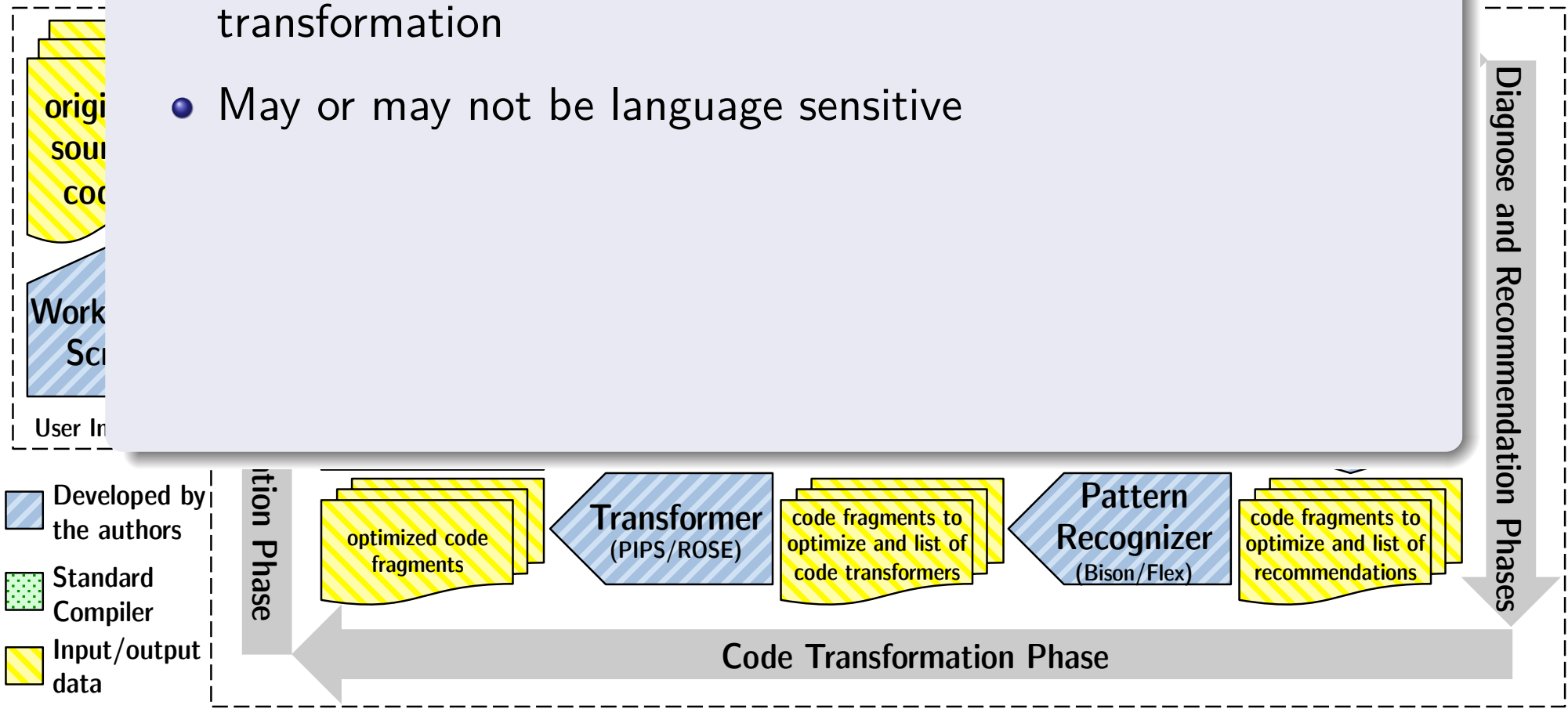
New Version: Transformer

- Implements the recommendation by applying source code transformation



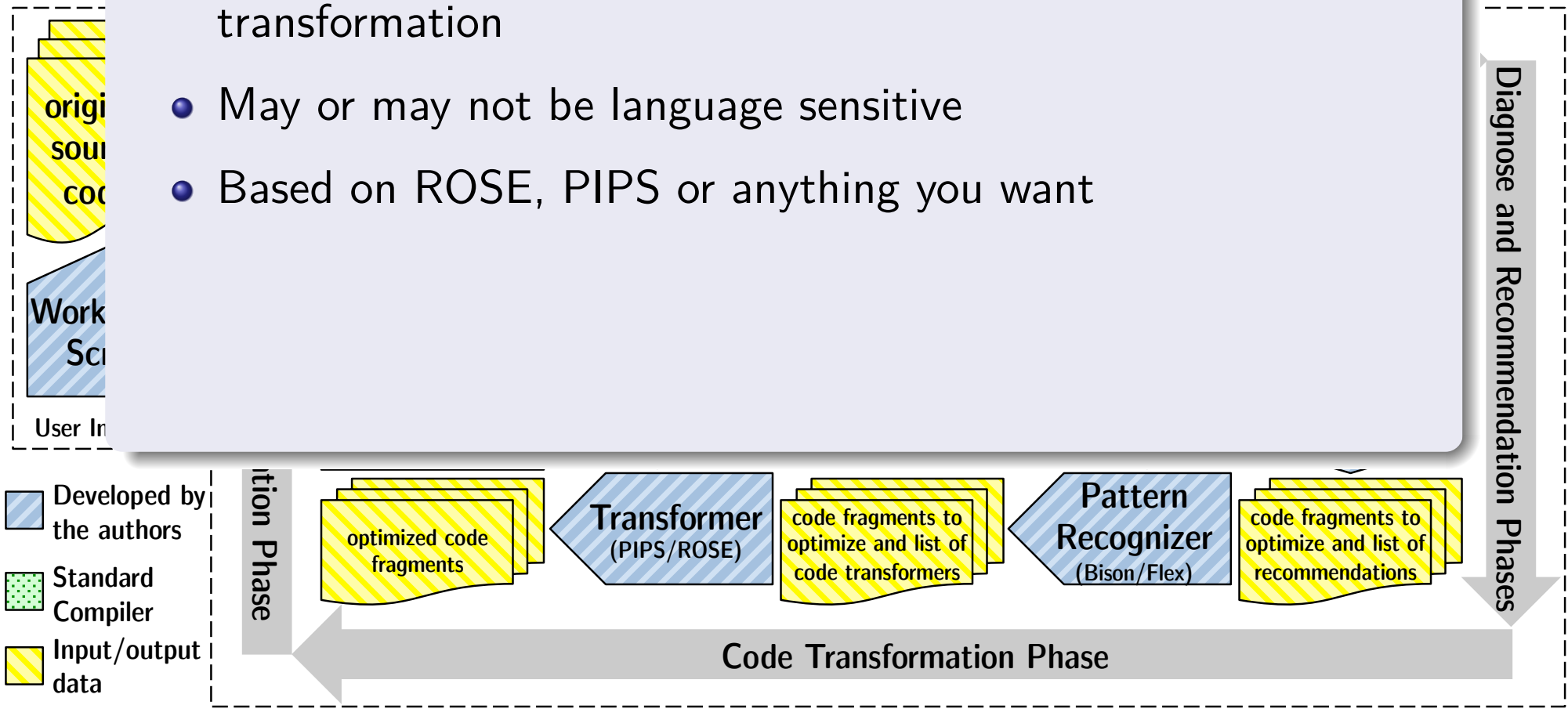
New Version: Transformer

- Implements the recommendation by applying source code transformation
- May or may not be language sensitive



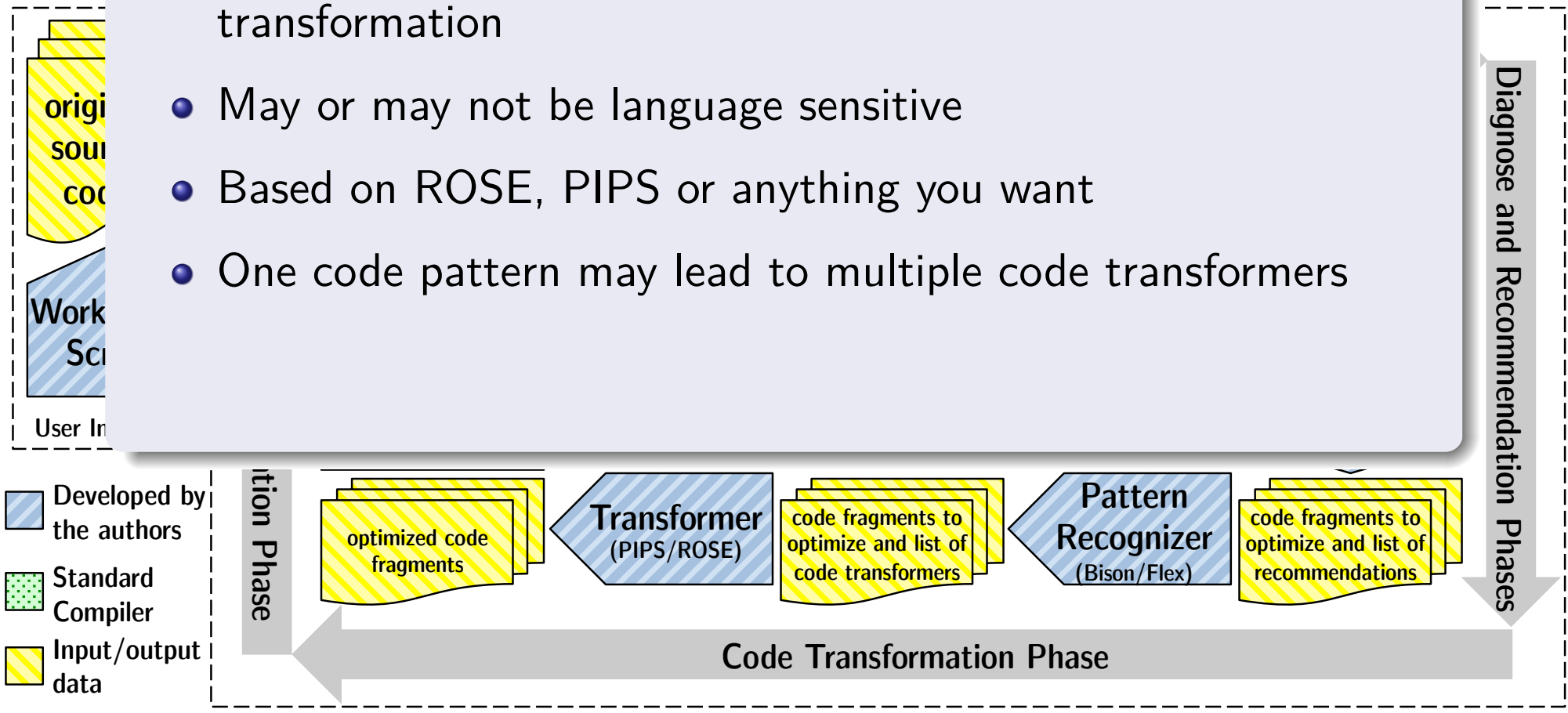
New Version: Transformer

- Implements the recommendation by applying source code transformation
- May or may not be language sensitive
- Based on ROSE, PIPS or anything you want



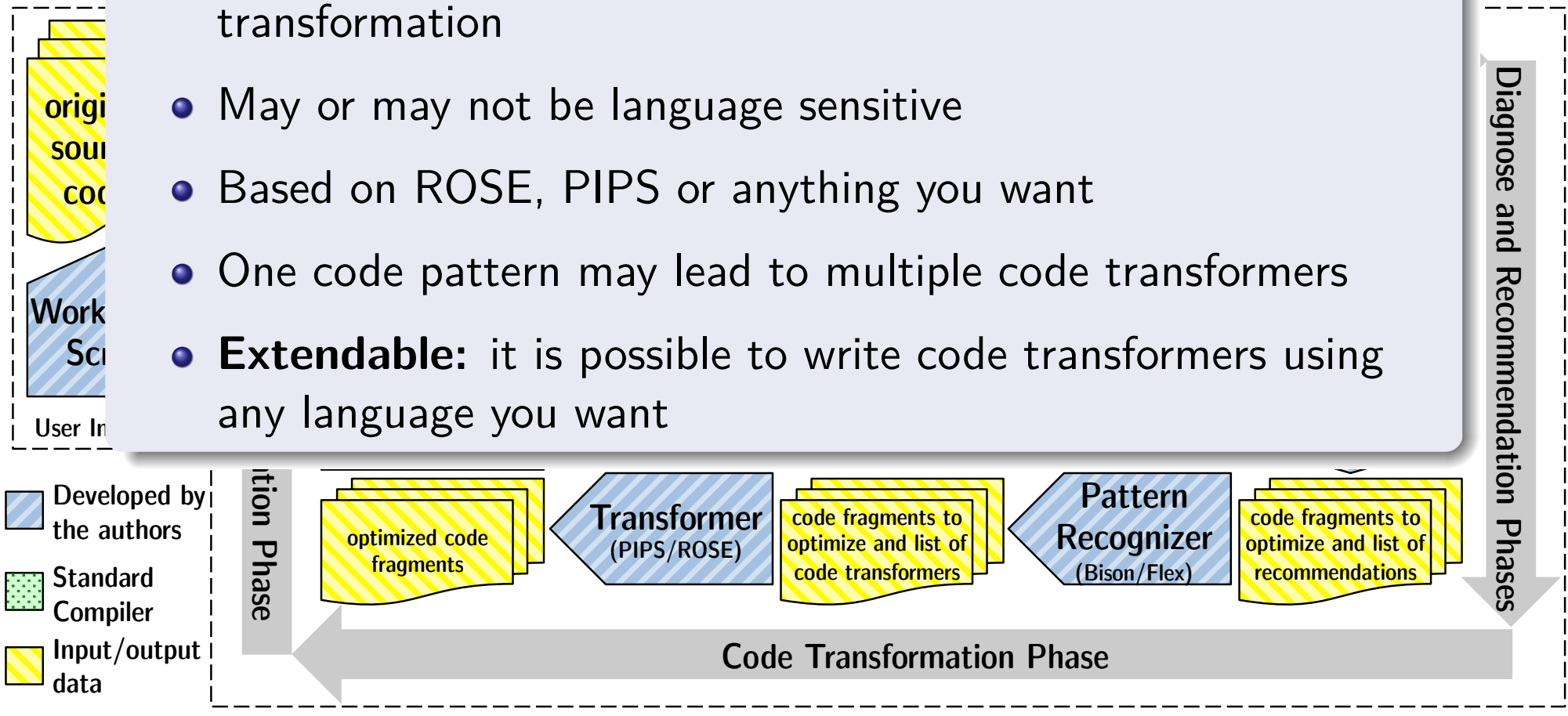
New Version: Transformer

- Implements the recommendation by applying source code transformation
- May or may not be language sensitive
- Based on ROSE, PIPS or anything you want
- One code pattern may lead to multiple code transformers

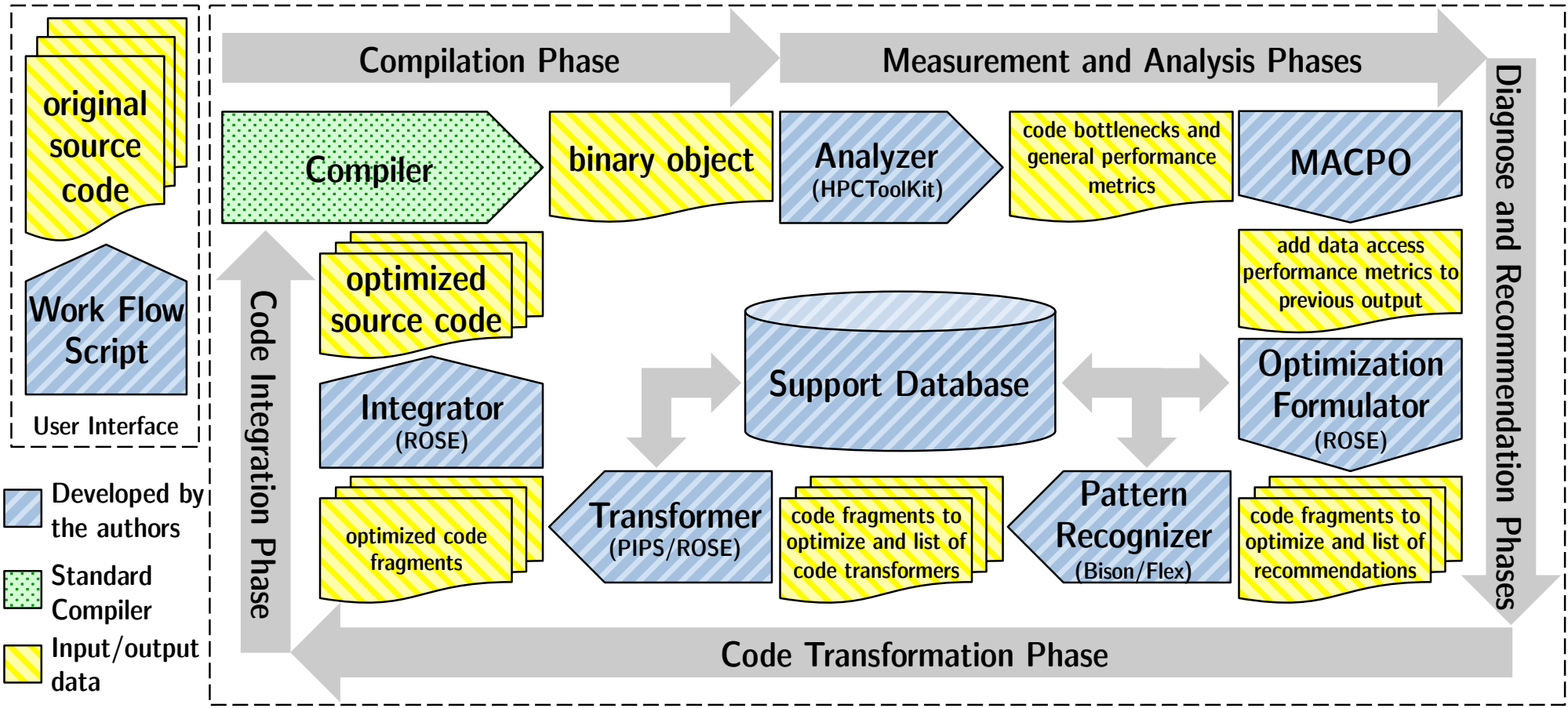


New Version: Transformer

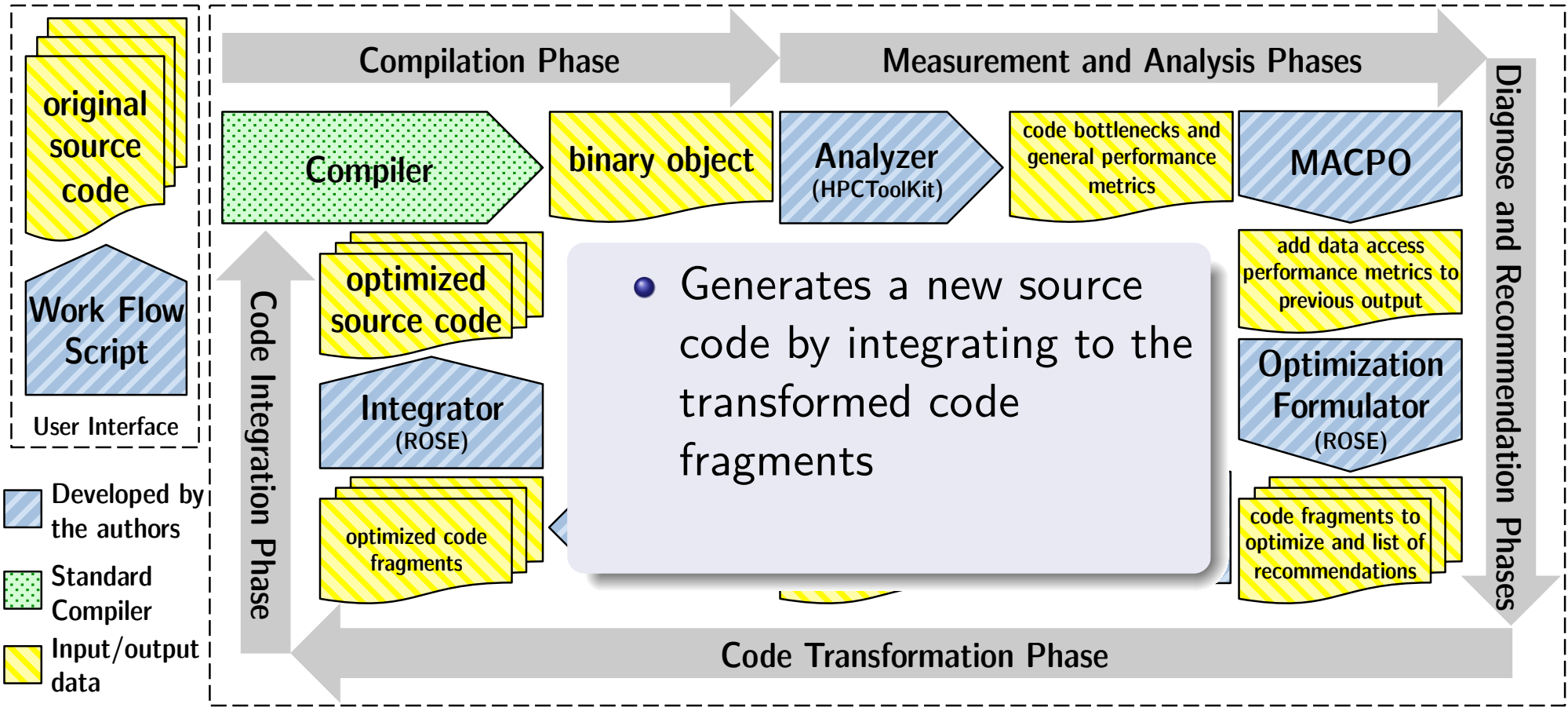
- Implements the recommendation by applying source code transformation
- May or may not be language sensitive
- Based on ROSE, PIPS or anything you want
- One code pattern may lead to multiple code transformers
- **Extendable:** it is possible to write code transformers using any language you want



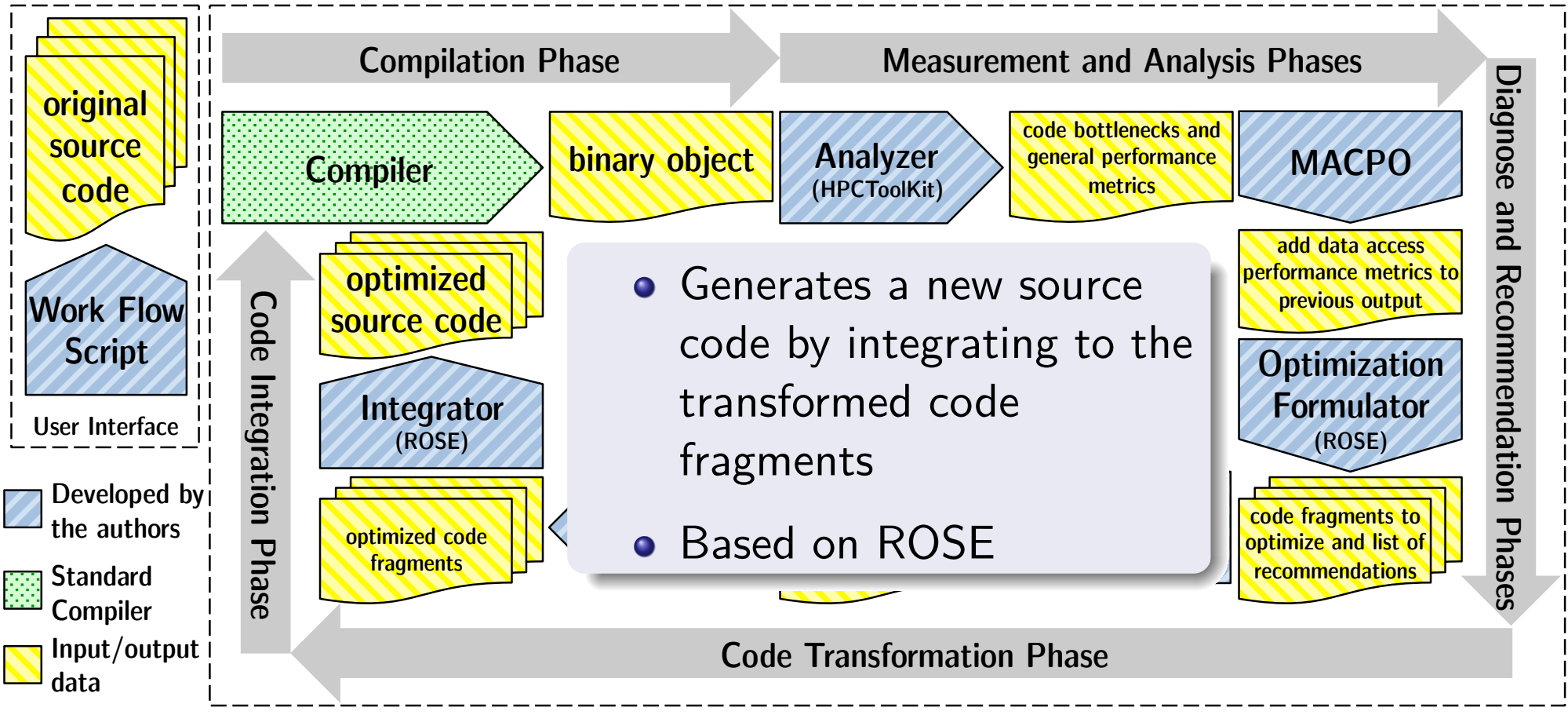
New Version: Integrator



New Version: Integrator



New Version: Integrator



New Version: Key Points

New Version: Key Points

Why is this performance optimization "architecture" strong?

New Version: Key Points

Why is this performance optimization "architecture" strong?

- Each piece of the tool chain can be updated/upgraded individually

New Version: Key Points

Why is this performance optimization "architecture" strong?

- Each piece of the tool chain can be updated/upgraded individually
- It is flexible: you can add new metrics as well as plug new tools to measure application performance

New Version: Key Points

Why is this performance optimization "architecture" strong?

- Each piece of the tool chain can be updated/upgraded individually
- It is flexible: you can add new metrics as well as plug new tools to measure application performance
- It is extendable: new recommendations, transformations and strategies to select recommendations (we are counting on you!)

New Version: Key Points

Why is this performance optimization "architecture" strong?

- Each piece of the tool chain can be updated/upgraded individually
- It is flexible: you can add new metrics as well as plug new tools to measure application performance
- It is extendable: new recommendations, transformations and strategies to select recommendations (we are counting on you!)
- Multi-language, multi-architecture, open-source and built on top of established tools

New Version: Key Points

Why is this performance optimization "architecture" strong?

- Each piece of the tool chain can be updated/upgraded individually
- It is flexible: you can add new metrics as well as plug new tools to measure application performance
- It is extendable: new recommendations, transformations and strategies to select recommendations (we are counting on you!)
- Multi-language, multi-architecture, open-source and built on top of established tools
- Easy to use and lightweight!

Agenda

Conclusions

Conclusions

- This is the first end-to-end open-source performance optimization tool (as far as we know)

Conclusions

- This is the first end-to-end open-source performance optimization tool (as far as we know)
- It will become more and more powerful as new recommendations, transformations and features are added

Conclusions

- This is the first end-to-end open-source performance optimization tool (as far as we know)
- It will become more and more powerful as new recommendations, transformations and features are added
- Different from (most of) the available performance optimization tools, there is no "big code" (to increase in complexity until it become unusable or too hard to maintain)

Next Steps

Next Steps

Major Goals

Next Steps

Major Goals

- Improve analysis based on the data access (short term)

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)
- New algorithms for recommendations selection (short term)

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)
- New algorithms for recommendations selection (short term)
- Add support to MPI-related recommendations (medium term)

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)
- New algorithms for recommendations selection (short term)
- Add support to MPI-related recommendations (medium term)
- Add support to MPI-related code transformations (long term)

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)
- New algorithms for recommendations selection (short term)
- Add support to MPI-related recommendations (medium term)
- Add support to MPI-related code transformations (long term)

Minor Goals

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)
- New algorithms for recommendations selection (short term)
- Add support to MPI-related recommendations (medium term)
- Add support to MPI-related code transformations (long term)

Minor Goals

- Support "Makefile"-based source code/compilation tree

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)
- New algorithms for recommendations selection (short term)
- Add support to MPI-related recommendations (medium term)
- Add support to MPI-related code transformations (long term)

Minor Goals

- Support "Makefile"-based source code/compilation tree
- Make the required packages installation process easier

TACC THE UNIVERSITY OF TEXAS AT AUSTIN

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)
- New algorithms for recommendations selection (short term)
- Add support to MPI-related recommendations (medium term)
- Add support to MPI-related code transformations (long term)

Minor Goals

- Support "Makefile"-based source code/compilation tree
- Make the required packages installation process easier
- Add a test suite based on established benchmark codes

Next Steps

Major Goals

- Improve analysis based on the data access (short term)
- Increase the number of recommendations and possible code transformations (continuously)
- New algorithms for recommendations selection (short term)
- Add support to MPI-related recommendations (medium term)
- Add support to MPI-related code transformations (long term)

Minor Goals

- Support "Makefile"-based source code/compilation tree
- Make the required packages installation process easier
- Add a test suite based on established benchmark codes
- Easy-to-use interface to manipulate the support database

TACC THE UNIVERSITY OF TEXAS AT AUSTIN

Thank You