



MySQL's Concrete Architecture (Query Processor)

Tabs vs. Spaces

Varsha Ragavendran

Nisha Sharma

Davood Anbarnam

Ayman Abualsunun

Anton Sitkovets

Glib Sitiugin

Kevin Arindaeng

Introduction



Query Processor is within the Logical Layer of MySQL

- Processes queries
- Determines how to execute the query
- Query must be semantically and syntactically valid
- Executes the Query in optimized manner

Overview



Derivation Process

System Architecture

Overall Query Processor Architecture Analysis

Subsystem Analysis

Use Cases

Conclusions

Lessons Learned

Derivation Process

Derivation Process

System Architecture

Overall Query Processor Architecture Analysis

Subsystem Analysis

Use Cases

Conclusions

Lessons Learned

```
8 sys.argv[1] = .contain file
9 sys.argv[2] = directory of MySQL source
10 sys.argv[3] = keyword to search for
11
12 '''
13 import sys
14 import re
15 import string
16 import os
17
18 # open the .contain file
19 f = open(sys.argv[1], 'r')
20 for line in f.readlines():
21     # get the absolute path of each file in the .contain file
22     mysql_file_path = os.path.join(sys.argv[2], os.path.normpath(line.split()[-1]))
23     # make sure it's a file before opening; some of them are directories
24     if os.path.isfile(mysql_file_path):
25         with open(mysql_file_path) as mysql_file:
26             try:
27                 for mysql_file_line in mysql_file:
28                     # if the keyword is in your file, print out file to system out, then go onto the next file.
29                     if sys.argv[3] in mysql_file_line:
30                         print(mysql_file_path)
31                         break
32             # some files give a UnicodeDecodeError when analyzing its text; skip them
33             except UnicodeDecodeError:
34                 pass;
```

<https://github.com/azkevin/EECS4314/blob/master/A2/a2data.py>

Derivation Process

Derivation Process

System Architecture

Overall Query Processor Architecture Analysis

Subsystem Analysis

Use Cases

Conclusions

Lessons Learned

```
Karin@KevinA-YOGA720: ~/git/EECS4314/A2 (master)
$ python a2data.py "C:\\Users\\Karin\\git\\EECS4314\\A2\\A2Data\\MySQL_UnderstandFileDependency.contain" "C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2" "ddl"
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\ndb\\test\\include\\NdbTest.hpp
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\ndb\\test\\include\\NdbRestarts.hpp
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\innobase\\dict\\dict0crea.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\innobase\\dict\\dict0mem.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\innobase\\dict\\dict0dd.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\innobase\\dict\\dict0dict.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\innobase\\dict\\dict0stats.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\innobase\\dict\\dict0load.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\extra\\libedit\\chared.c
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\extra\\libedit\\refresh.c
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\storage\\innobase\\page\\page0page.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\dd\\types\\system_view_definition.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\dd\\types\\object_table_definition.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_cmd_ddl_table.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\json_binary.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\replication.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_lex.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_truncate.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\handler.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\ha_ndbcluster.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_parse.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_tablespace.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_table.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\rpl_binlog_sender.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_admin.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\parse_tree_nodes.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_class.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\table_cache.cc
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\rpl_rli_pdb.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\mdl.h
C:\\Users\\Karin\\Desktop\\School\\EECS4314\\Assignment 2\\mysql-server-mysql-8.0.2\\sql\\sql_partition_admin.cc
```

Look for keywords, comments, name of the file, and file directory structure then add relevant files of the subsystem to the .contain file

System Architecture

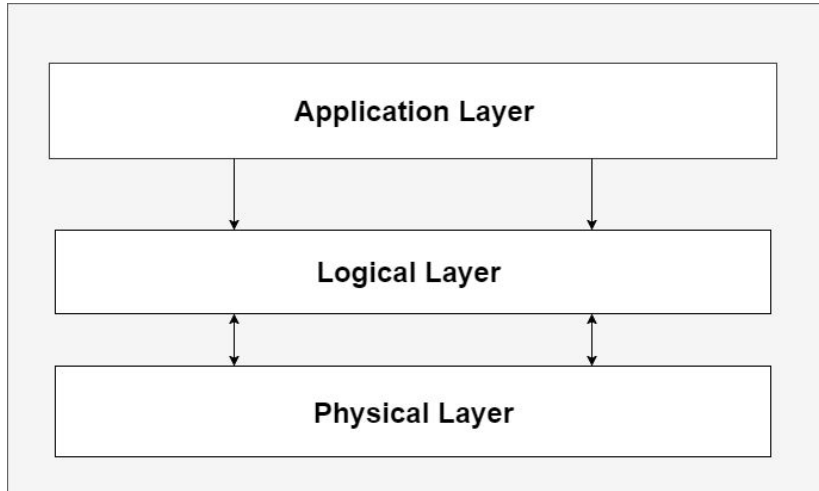


Key Findings:

- ❖ Concrete Architecture reveals that there Exist
 - Some new dependencies
 - Two way dependencies between different components and subsystems instead of One way dependencies

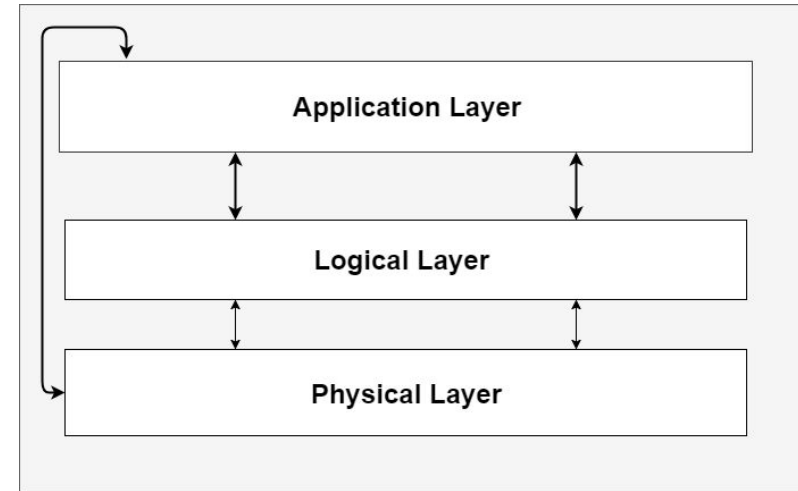
System Architecture

Key Findings:

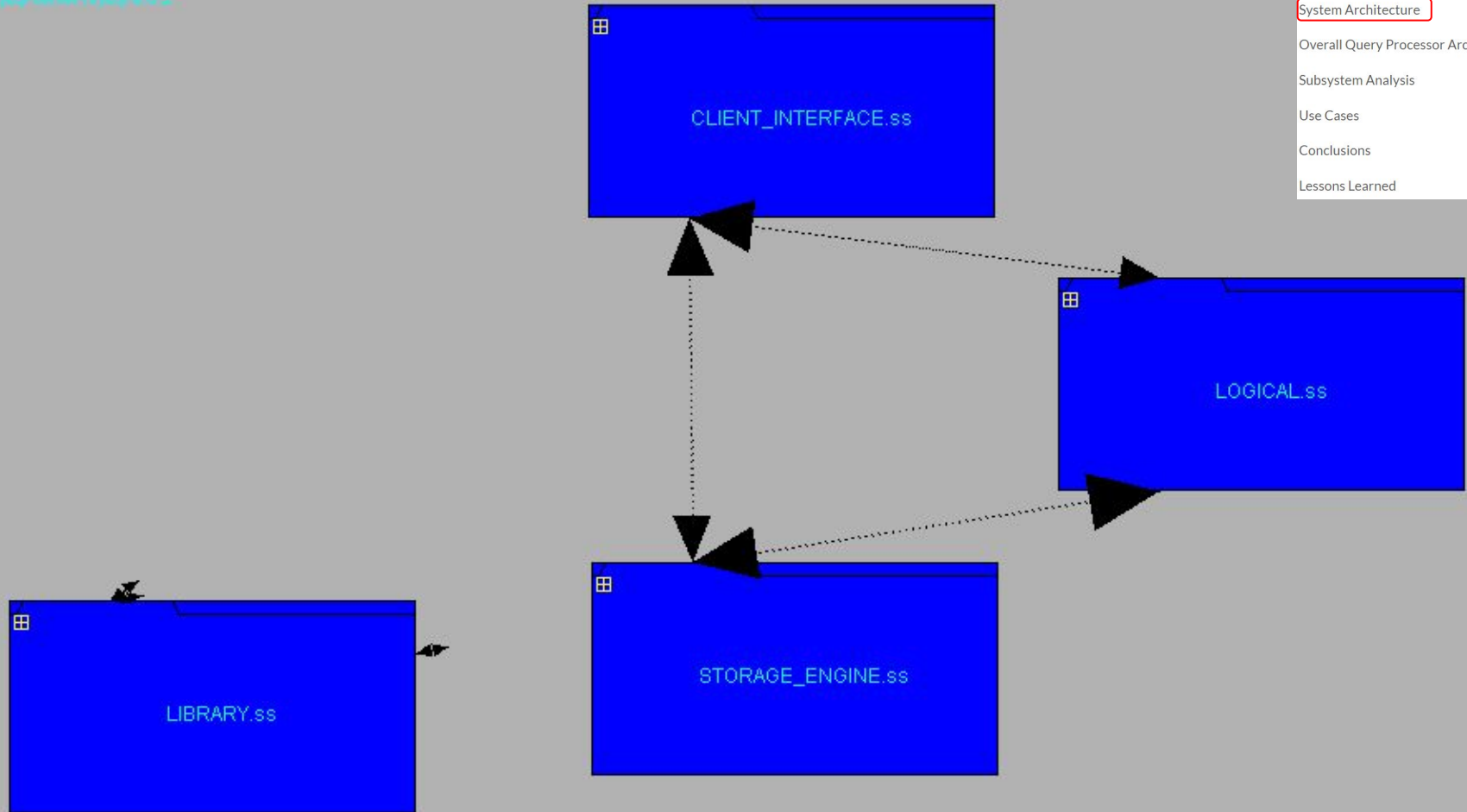


Simpler Conceptual Architecture

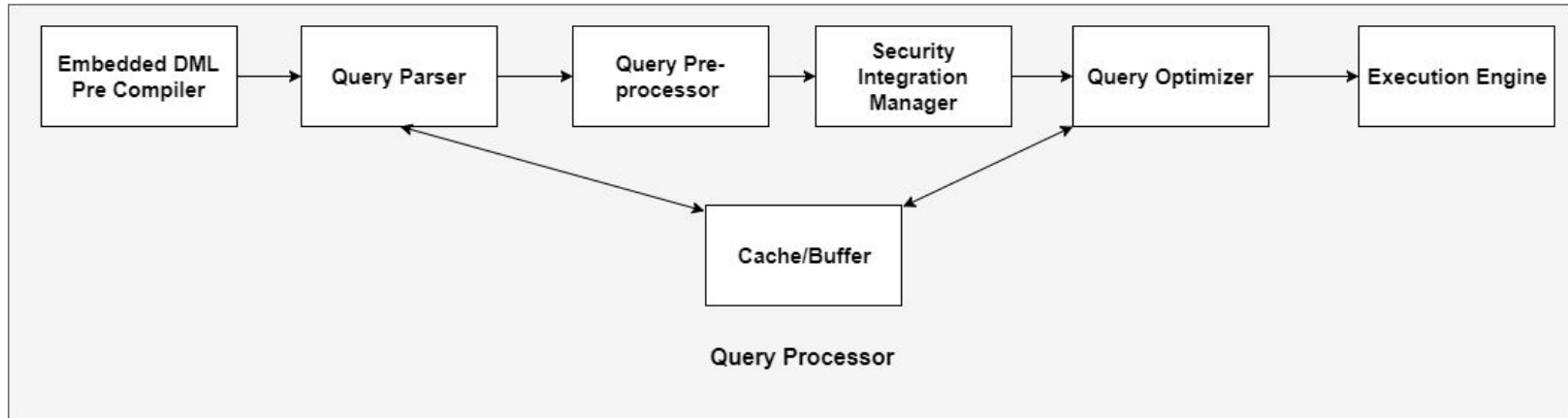
VS.



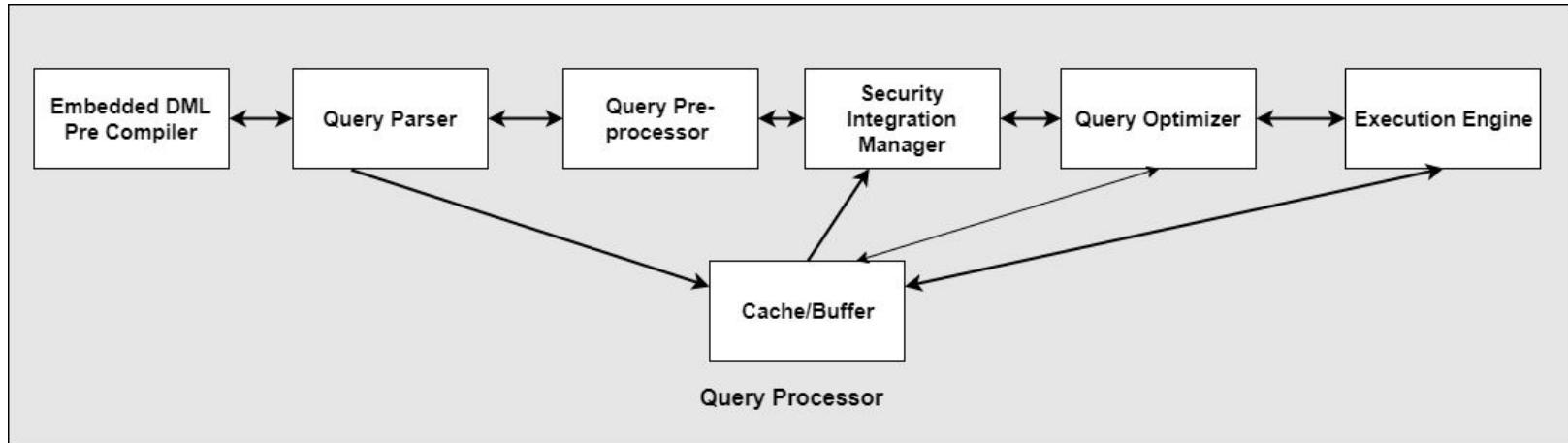
Simpler Concrete Architecture



Query Processor Conceptual Architecture



Query Processor Concrete Architecture



Changes are represented as bolded arrows

DDL Compiler

- Data Description Language (DDL): Syntax to create or alter the structure of the database.

Example SQL Statements:

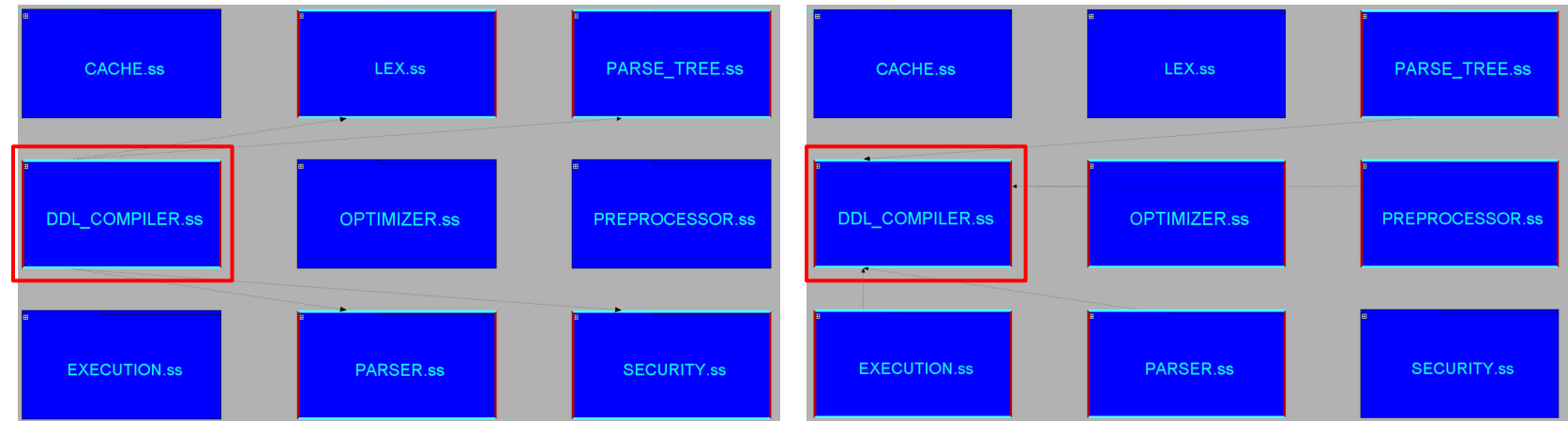
- CREATE
- ALTER
- DROP
- RENAME
- TRUNCATE

```
2  
3 • CREATE TABLE `data_log` (  
4     `REC_ID` int(11) NOT NULL AUTO_INCREMENT,  
5     `DATESTAMP` datetime DEFAULT NULL,  
6     `SCANNER_ID` varchar(25) NOT NULL,  
7     `BARCODE` varchar(255) DEFAULT NULL,  
8     `WEIGHT` DOUBLE DEFAULT NULL,  
9     PRIMARY KEY (`REC_ID`)  
10 );  
11
```

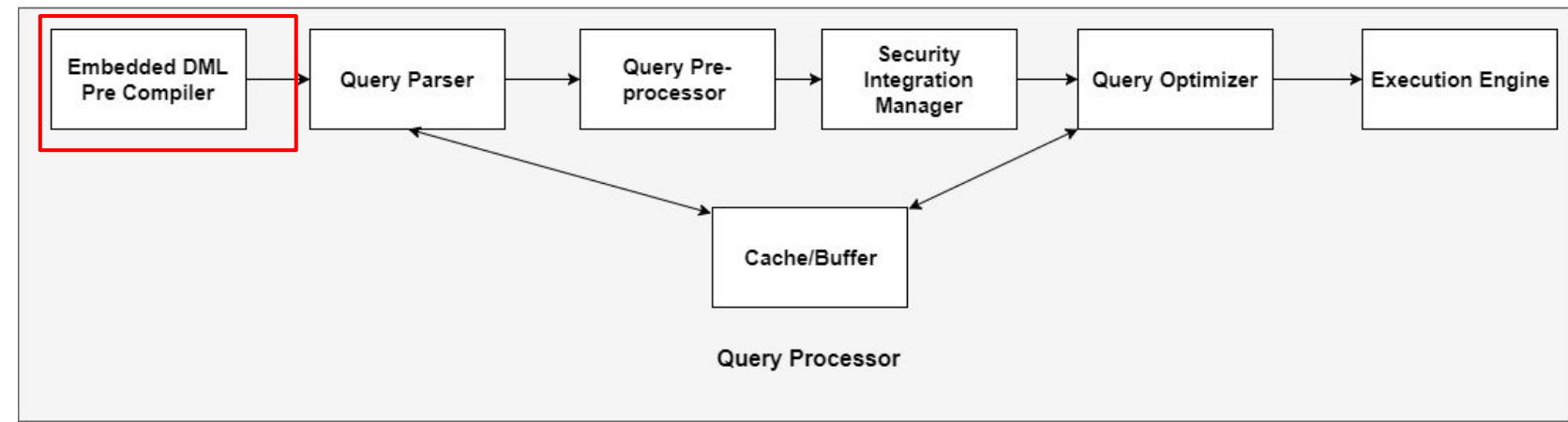
The DDL Compiler converts these statements into machine/object code for MySQL usage.

DDL Compiler

- Derivation Process
- System Architecture
- Overall Query Processor Architecture Analysis
- Subsystem Analysis**
- Use Cases
- Conclusions
- Lessons Learned



Concrete



Conceptual

DDL Compiler

Derivation Process

System Architecture

Overall Query Processor Architecture Analysis

Subsystem Analysis

Use Cases

Conclusions

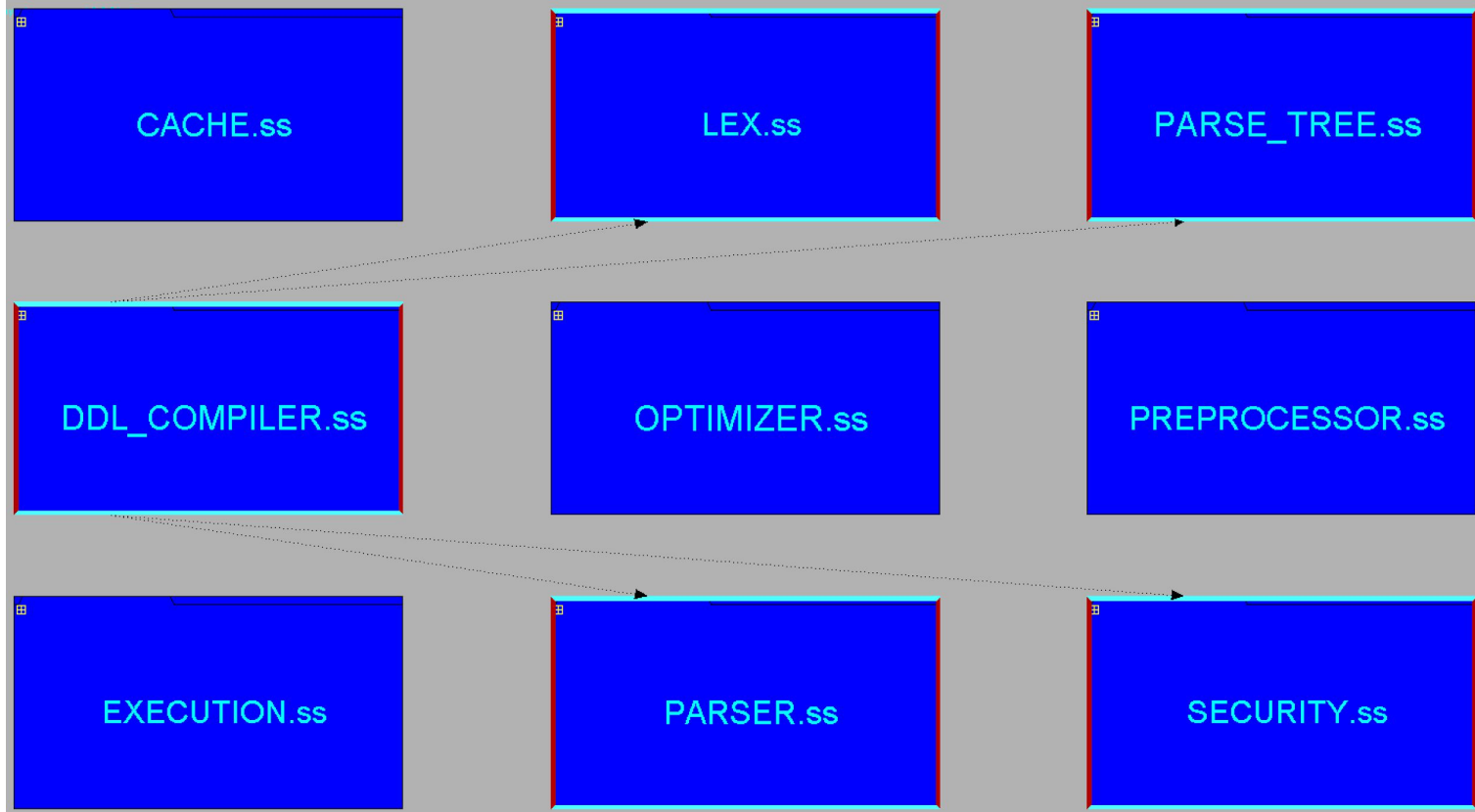
Lessons Learned

LEX

PARSE_TREE

PARSER

SECURITY

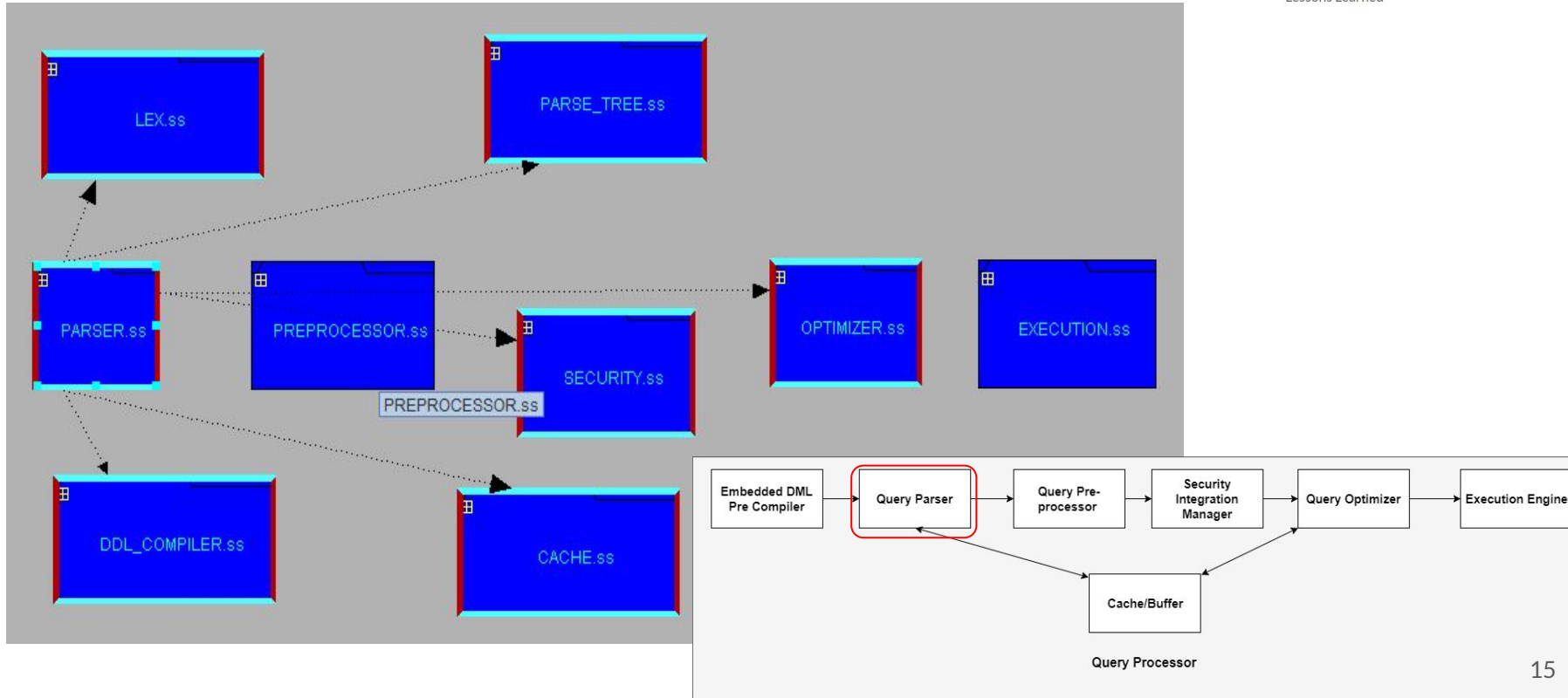


Query Parser

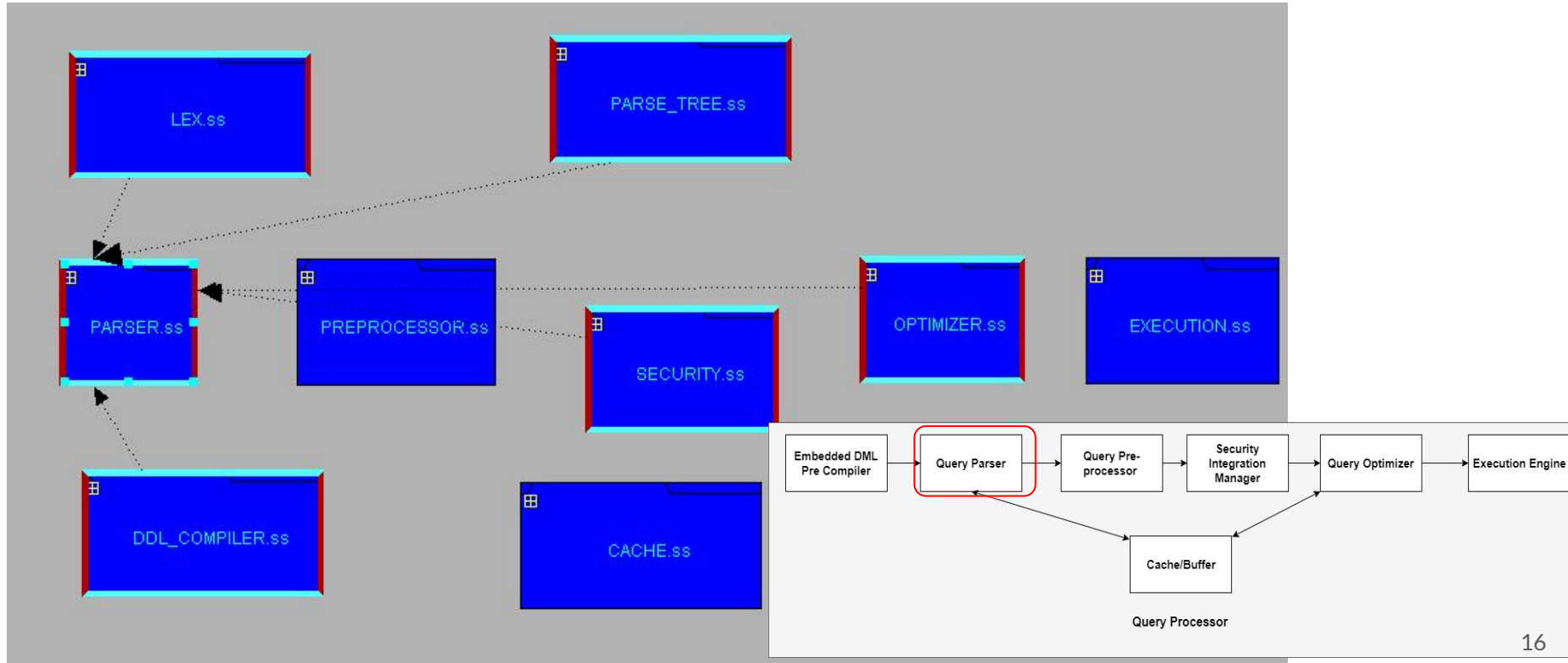


- 1) Interpreter and visitor design patterns.
- 2) Conceptual vs concrete architecture.
- 3) Parser to other subsystems (security, DD_compiler, Cache , Parse Tree, LEX, Optimizer).
- 4) Other Subsystems to Parser (Optimizer, Security, DD_compiler , Parse Tree, LEX).
- 5) New subsystems.

Parser to other systems dependencies



Other systems to Parser dependencies

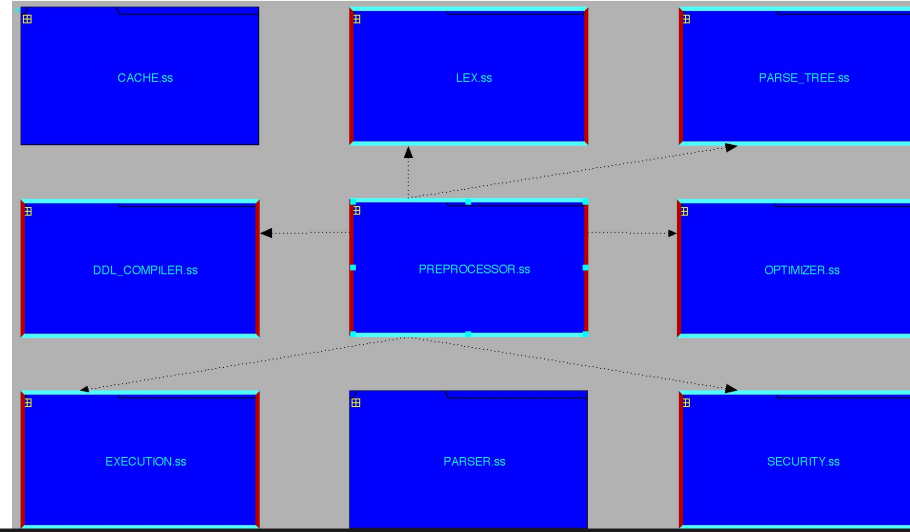


Query Preprocessor

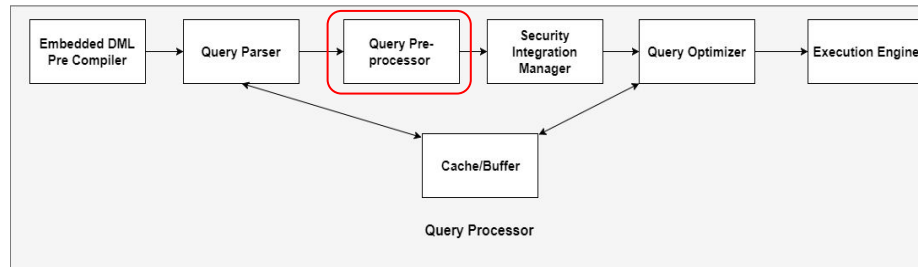


- Performs semantic validation
 - Verify that the relations and views are in the database schema
 - Verify that attributes have a corresponding relation specified
 - Verify attribute types
- Parse tree is valid only if it is syntactically and semantically valid

Preprocessor depending on other systems

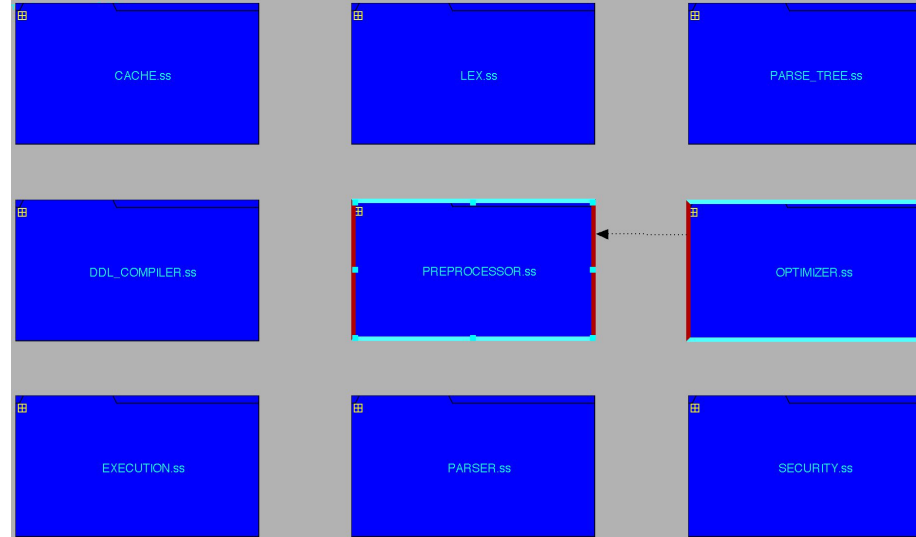


Concrete

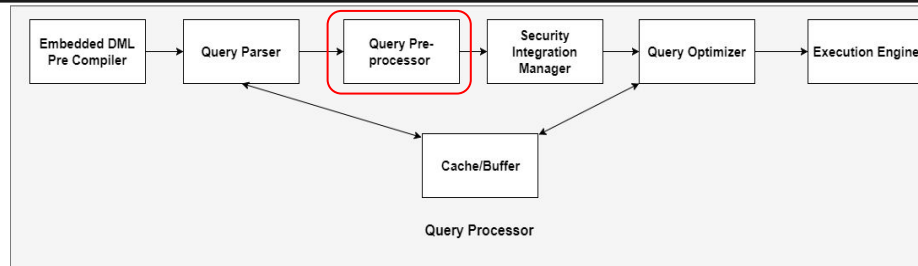


Conceptual

Other Systems depending on Preprocessor



Concrete



Conceptual

Security Integration Manager

Derivation Process

System Architecture

Overall Query Processor Architecture Analysis

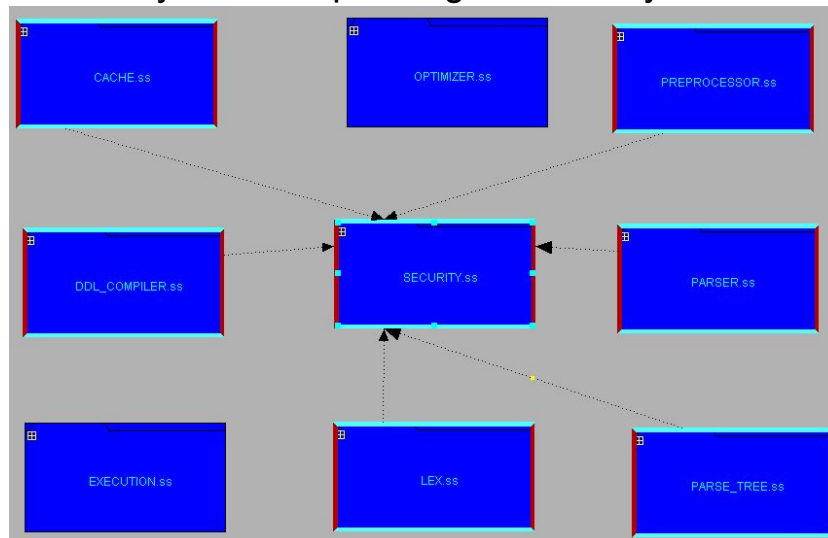
Subsystem Analysis

Use Cases

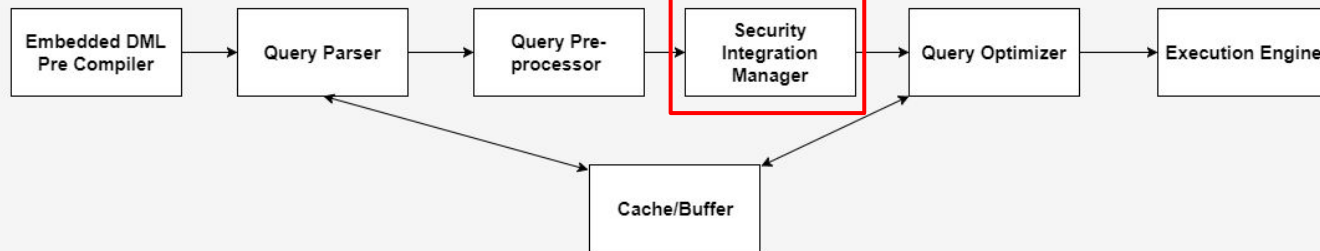
Conclusions

Lessons Learned

Other Systems Depending on Security



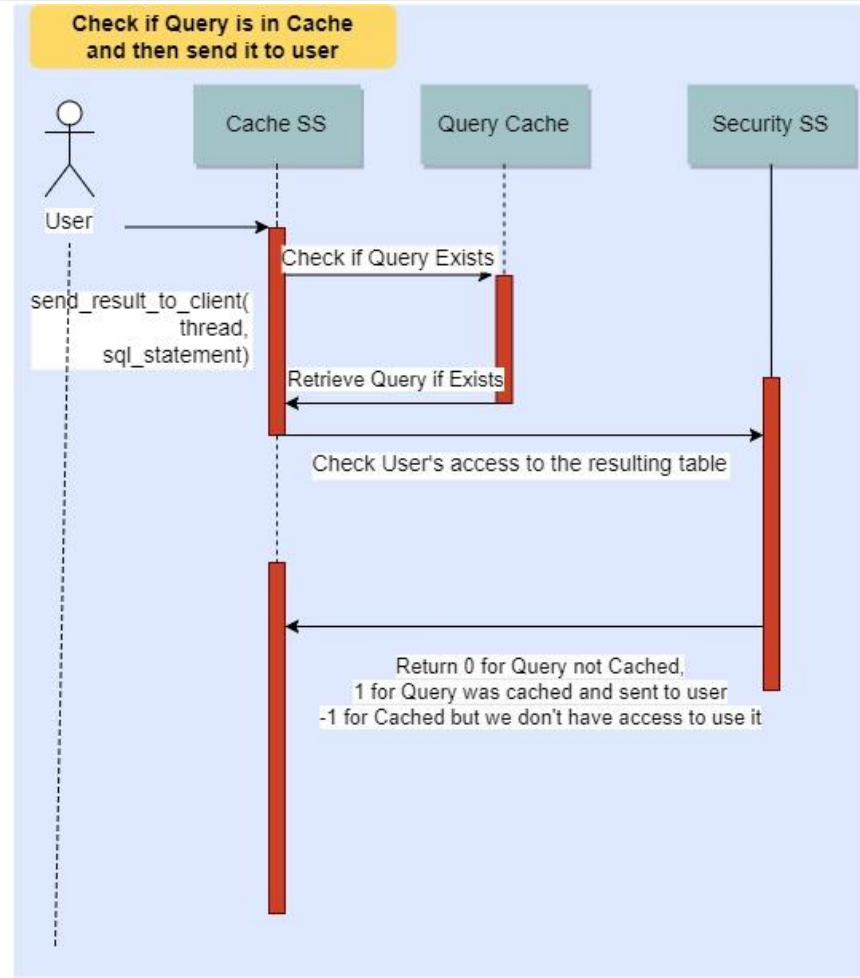
Concrete



Query Processor

Conceptual

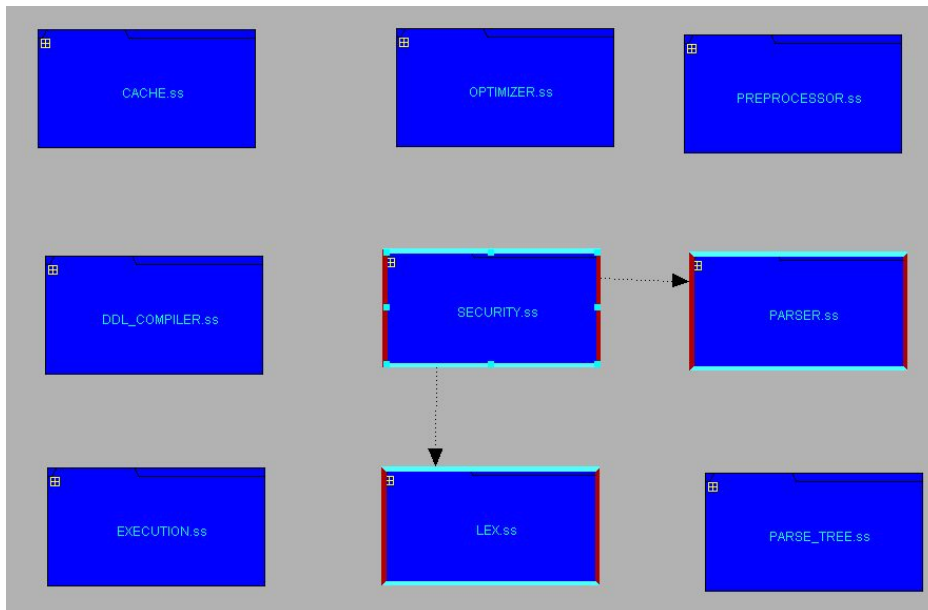
Use Case



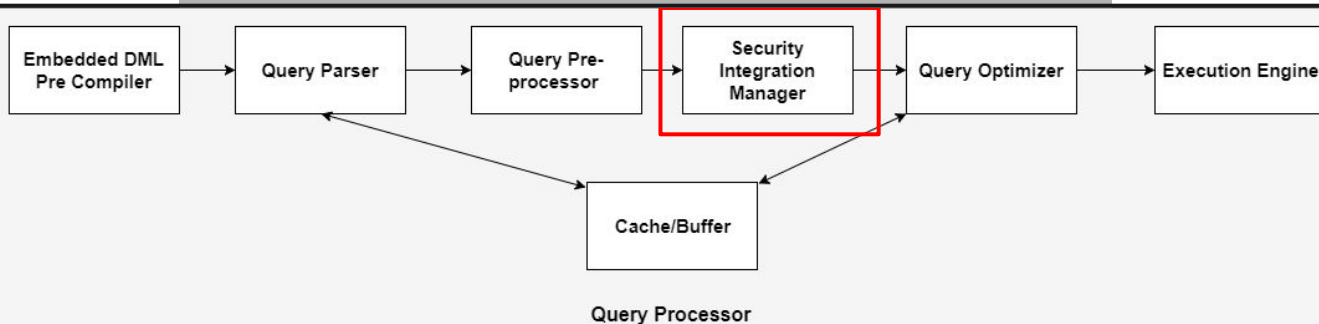
Security Integration Manager

Security depending on other systems

Derivation Process
System Architecture
Overall Query Processor Architecture Analysis
Subsystem Analysis
Use Cases
Conclusions
Lessons Learned



Concrete

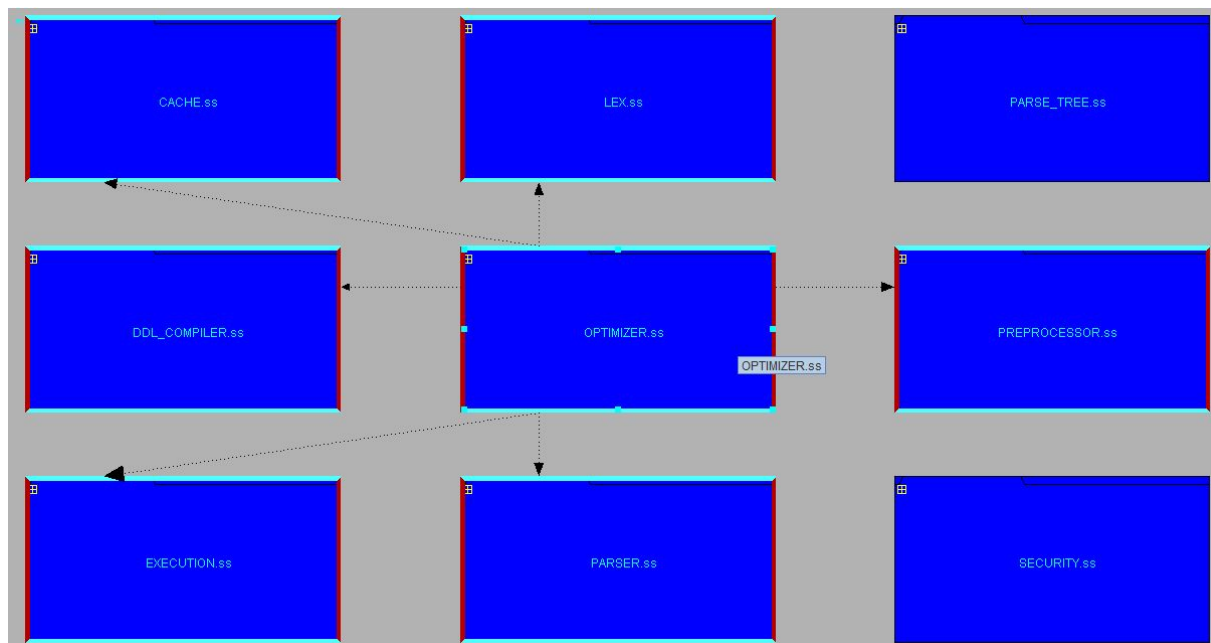


Conceptual

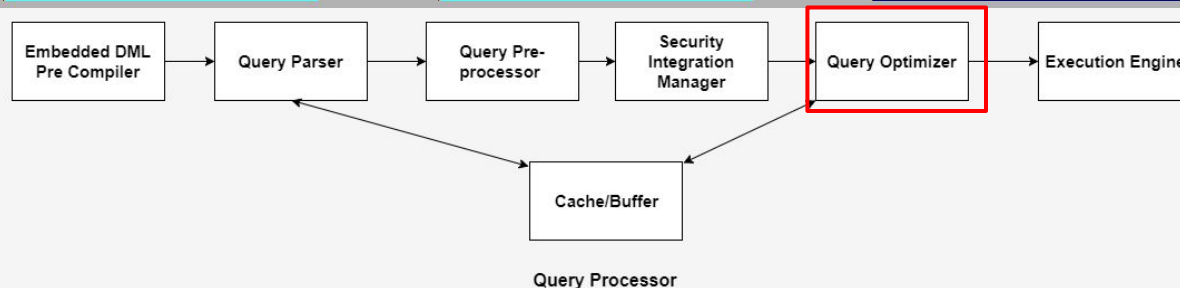
Optimizer to other systems dependencies

- Derivation Process
- System Architecture
- Overall Query Processor Architecture Analysis
 - Subsystem Analysis**
 - Use Cases
 - Conclusions
 - Lessons Learned

Concrete



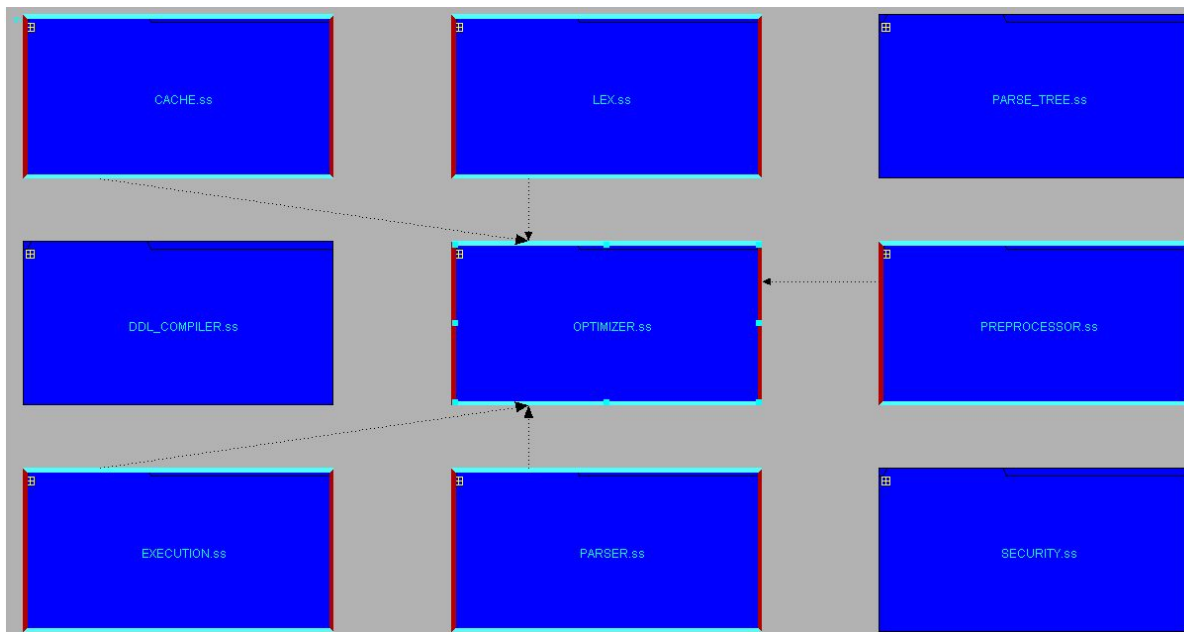
Conceptual



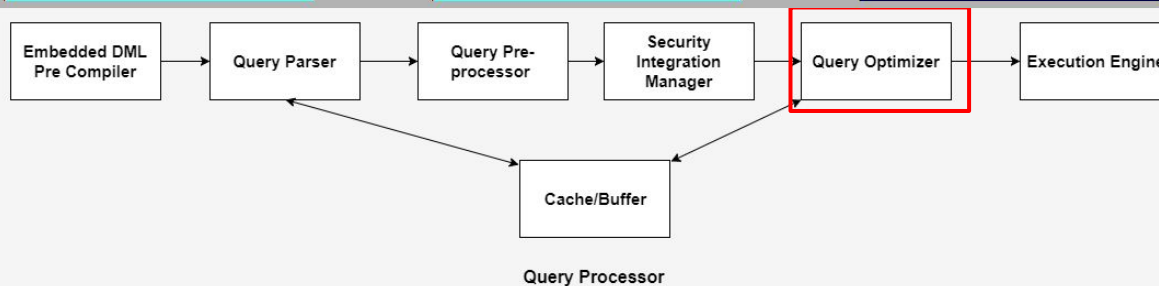
Other systems to Optimizer dependencies

Derivation Process
System Architecture
Overall Query Processor Architecture Analysis
Subsystem Analysis
Use Cases
Conclusions
Lessons Learned

Concrete



Conceptual



Execution Engine



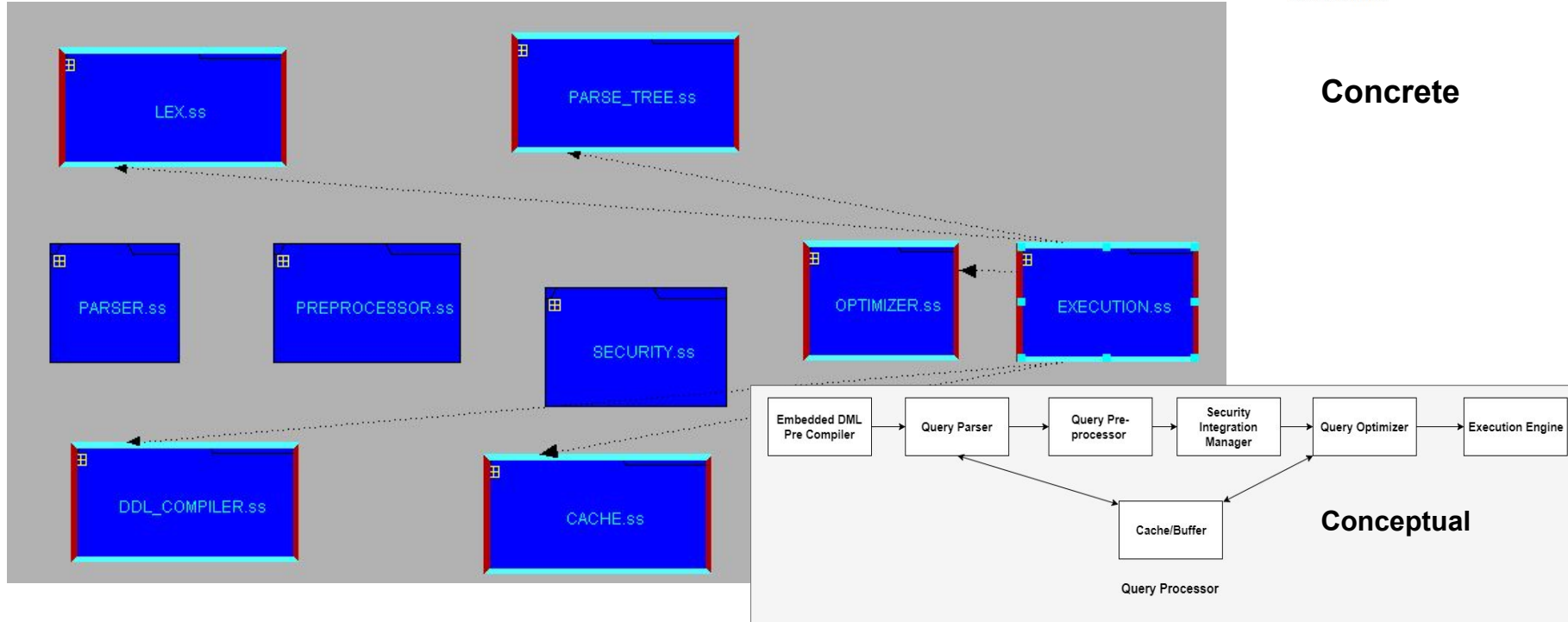
- 1) Execution engine to other subsystems dependencies.
- 2) Other subsystems to execution engine dependencies.
- 3) New subsystems.

Execution Engine

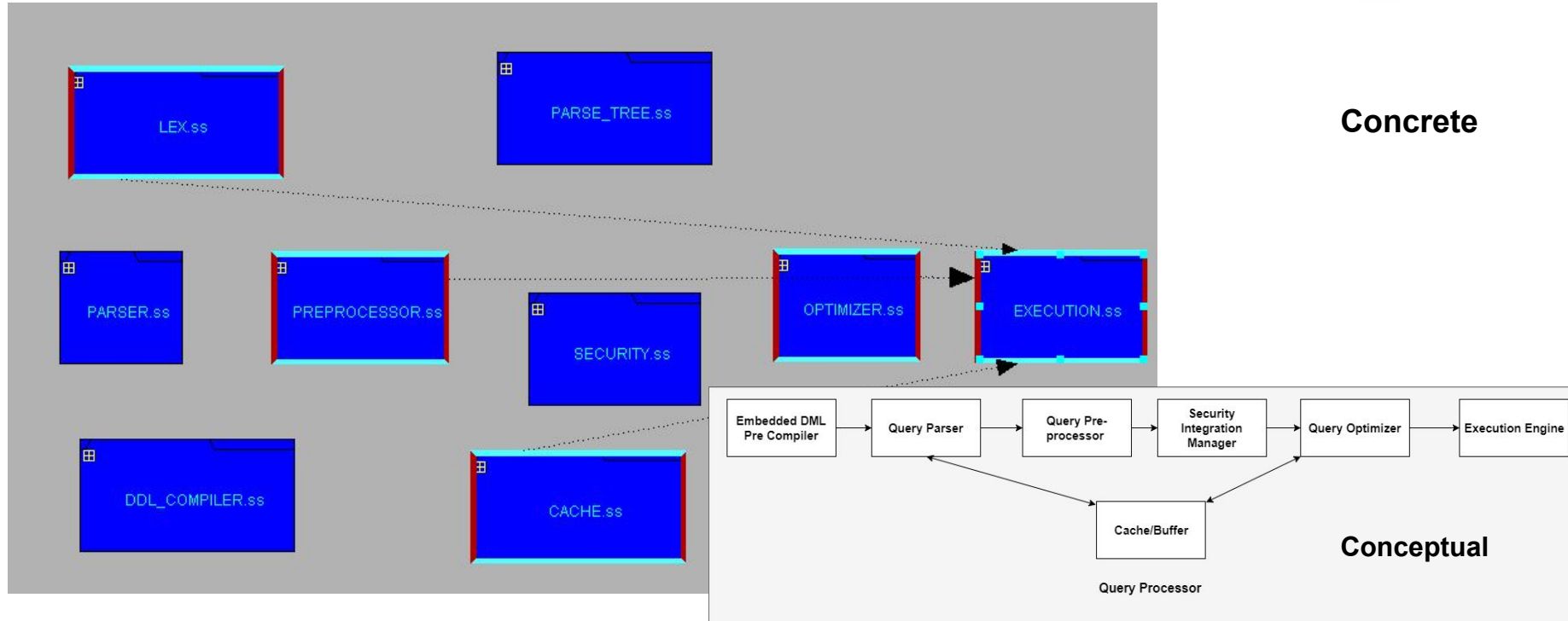


- 1) Execution engine to other subsystems dependencies.
- 2) Other subsystems to execution engine dependencies.
- 3) New subsystems.

Execution to other subsystems dependenc



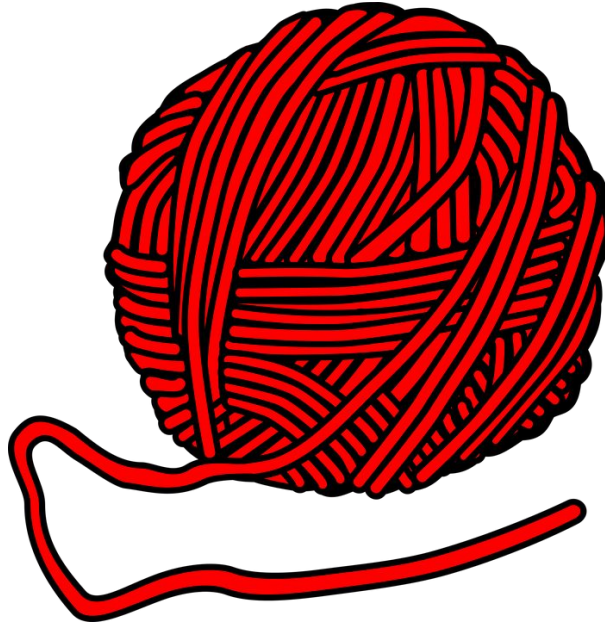
Other subsystems to execution dependent



Conclusions



-



Lessons Learned



- Lots of Work
- Iterative Process
- Automation
 - Scripts
 - Regex

It is difficult to derive concrete subsystems from looking at the source code and directory structure. Comments and documentation provide little help in understanding the concrete architecture.

Security Integration Manager is called many times throughout the query processing task by different subsystems, as opposed to just once.

Question Period

