TEAM PRESENTATION #3
-CFG GENERATOR

TEAM [T2]

200811415 김영현

200811457 조성우

Speaker → 200811465 허준행

Project Overview

- We develop the **CFG Generator** based on our last presentation.

- Just like the RVC project, SASD(Structured Analysis and Structured Design) is a main technique in this system development.

- We follow the proposed SRS(Software Requirement Specification).

Contents

- **S**tructured **A**nalysis
  - Environmental Model
    - Statement of Purpose
    - System Context Diagram
    - Event List
  - Behavioral Model
    - Data Flow Diagram
    - Data Dictionary
    - Process Specification

- **S**tructured **D**esign
  - Implementation Model
    - Structured Chart

ENVIRONMENTAL MODEL

Statement of Purpose (1/2)

CFG(Control Flow Graph) Generator

- Receiving a C source code, A CFG Generator outputs converted CFG report with a text(*.txt) file.

- The C source code should be convertible code.
  - It has 100~200 lines including main function.
  - It is a single-file that doesn't have user defined header files.
  - It doesn't include pointers.

- This CFG Generator's execution environment is Cygwin using CUI(Character User Interface), and execution command is
$ ./ExecutableFileName CFileName ReportFileName
  - When a user inputted invalid command, the program show *'help'* that includes command syntax.

Statement of Purpose (2/2)

- When C source code inputted successfully, the program shows *"success"* message. Or in error case, the program shows *"error"* and terminates the program.

- Before the program converting CFG, shows *"converting"* message.

- When converting CFG, the program shows execution order of c source code and outputs report file.

  - Converted blocks are printed in a table format having fields :

    [block#] / Type / Line / Description

  - Converted edges are printed in a table format having fields :

    [edge#] / Type / Source block / Destination block

- After report generating process, the program shows the name of report file.

- The program is developed based on SASD.
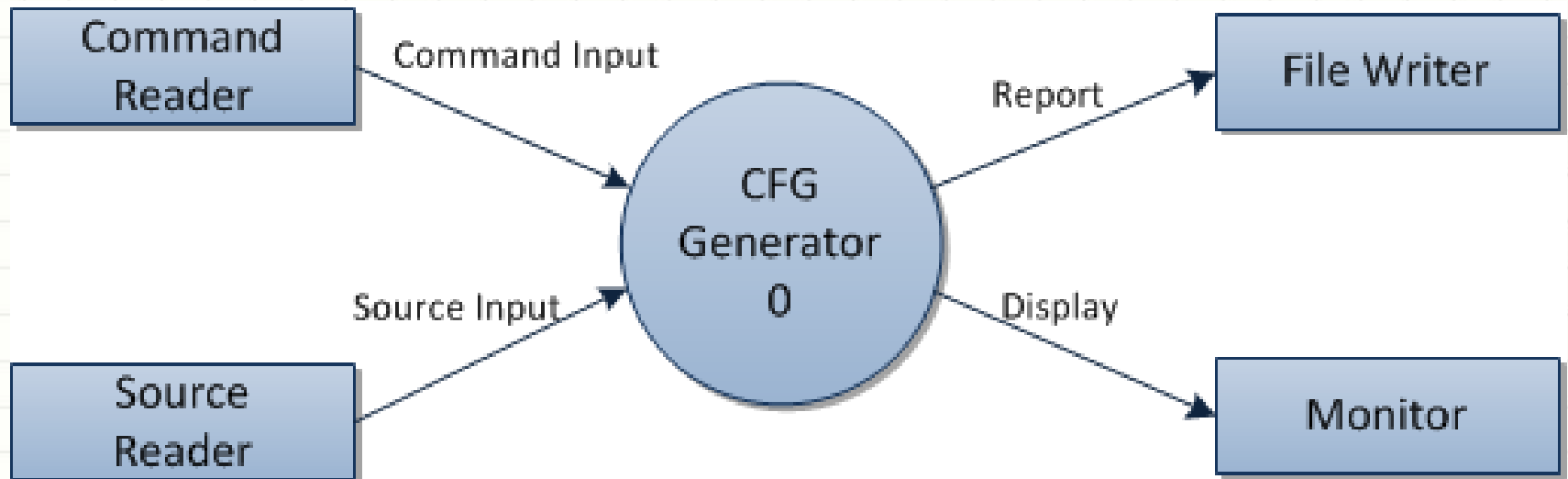
System Context Diagram
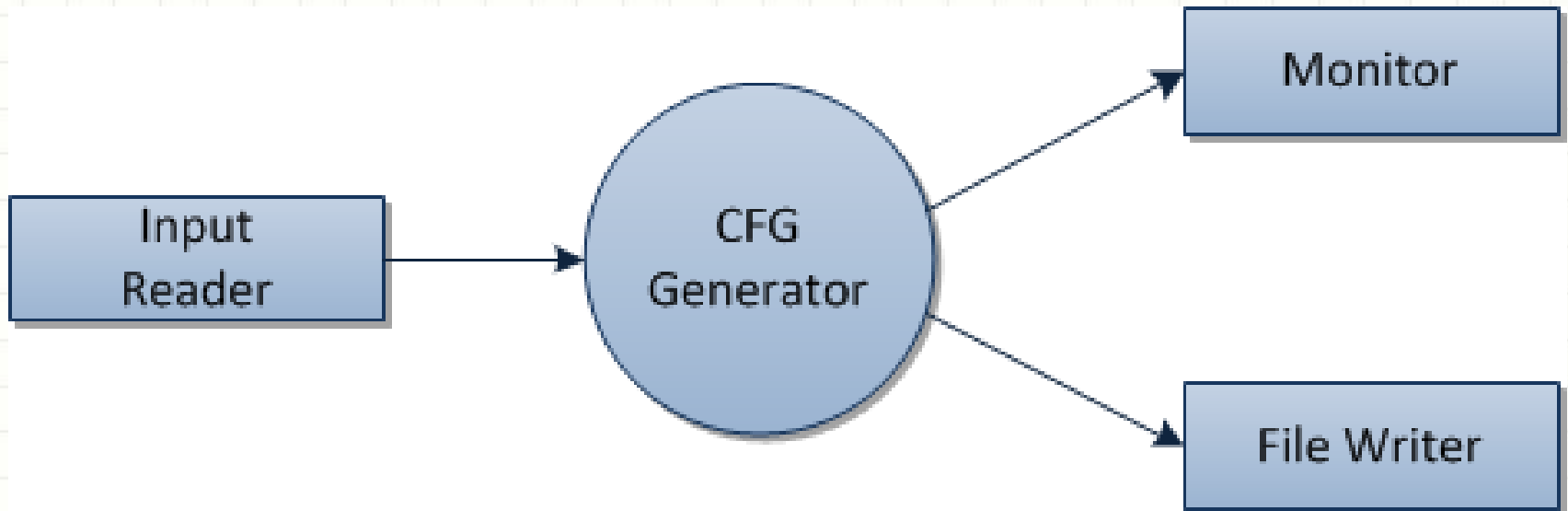
Event List

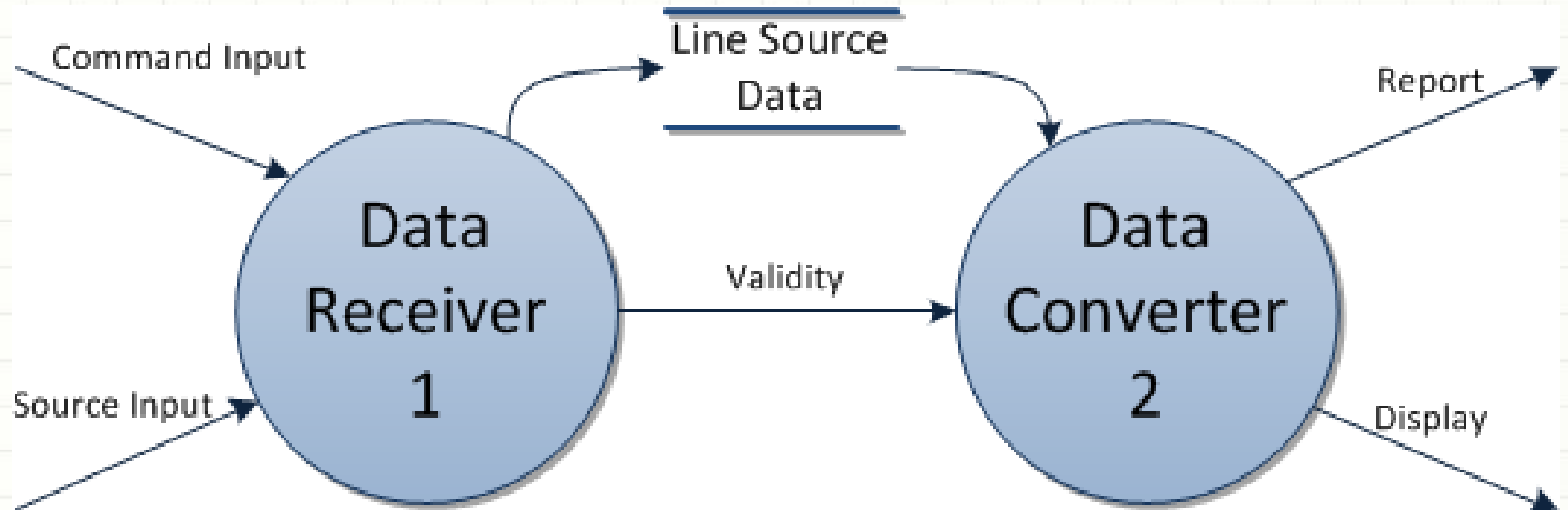| Input/Output Event	Description
Command Input	Receives a command from **Input Reader**
Source Input	Receives a C source code from **Input Reader**
Display	Prints conversions, and system messages to **Monitor**
Report	Prints table of 'blocks' and 'edges' of converted CFG to **File Writer**
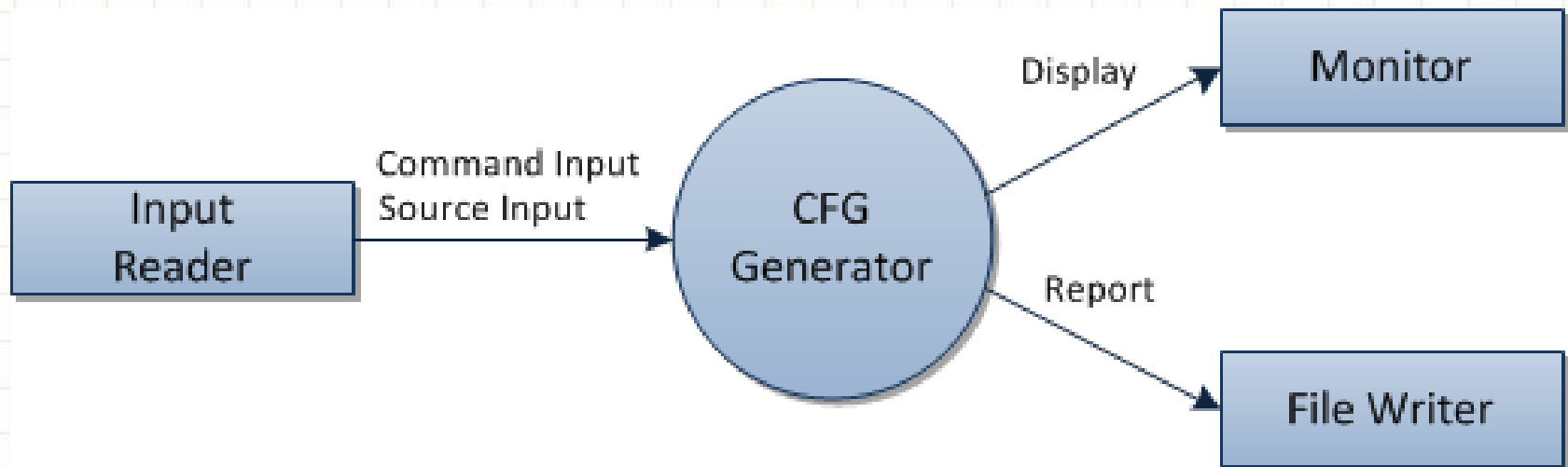
BEHAVIORAL MODEL

Data Flow Diagram – level 0

Data Flow Diagram – level 1

Data Dictionary – level 1

Data name	Description
Command Input	It is received command from **Command Reader**
Source Input	It is received C source code from **Source Reader**
Validity	An Integer value which shows states of system (i.e. validation of input data) **0**: ready to parse / **1**: unpermitted command / **2**: invalid filename
Report	It is an output data—table of 'blocks' and 'edges' of converted CFG—being sent to **File Writer**
Display	It is an output data—conversion information and system messages—being sent to **Monitor**

Line Source Data	Description
Line Data	It consists of line number and each line of code

Data Flow Diagram – level 2
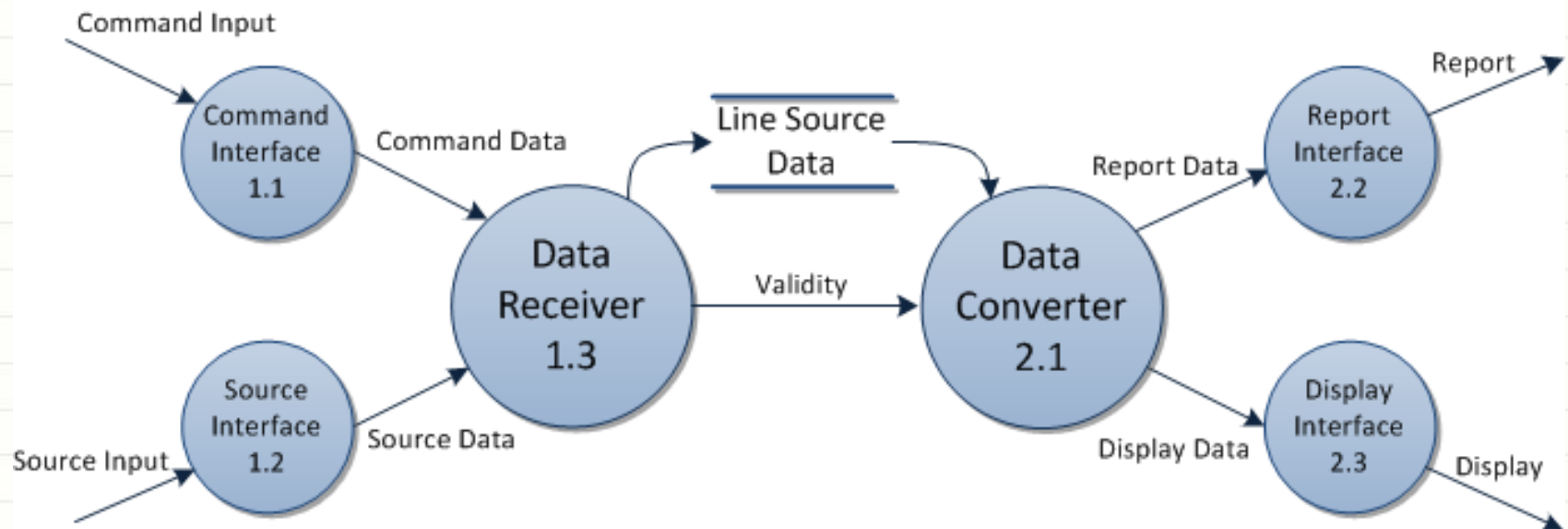
Data Dictionary – level 2

Data name	Description
Command Data	It is a processed data that will have **Data Receiver** check validity
Source Data	It is a processed line-by-line source input
Report Data	It is a data generated by system in order to be printed in file after being sent to **Report Interface**
Display Data	It is a data generated by system in order to be printed on display after being sent to **Display Interface**

Data Flow Diagram – level 3 (1/2)
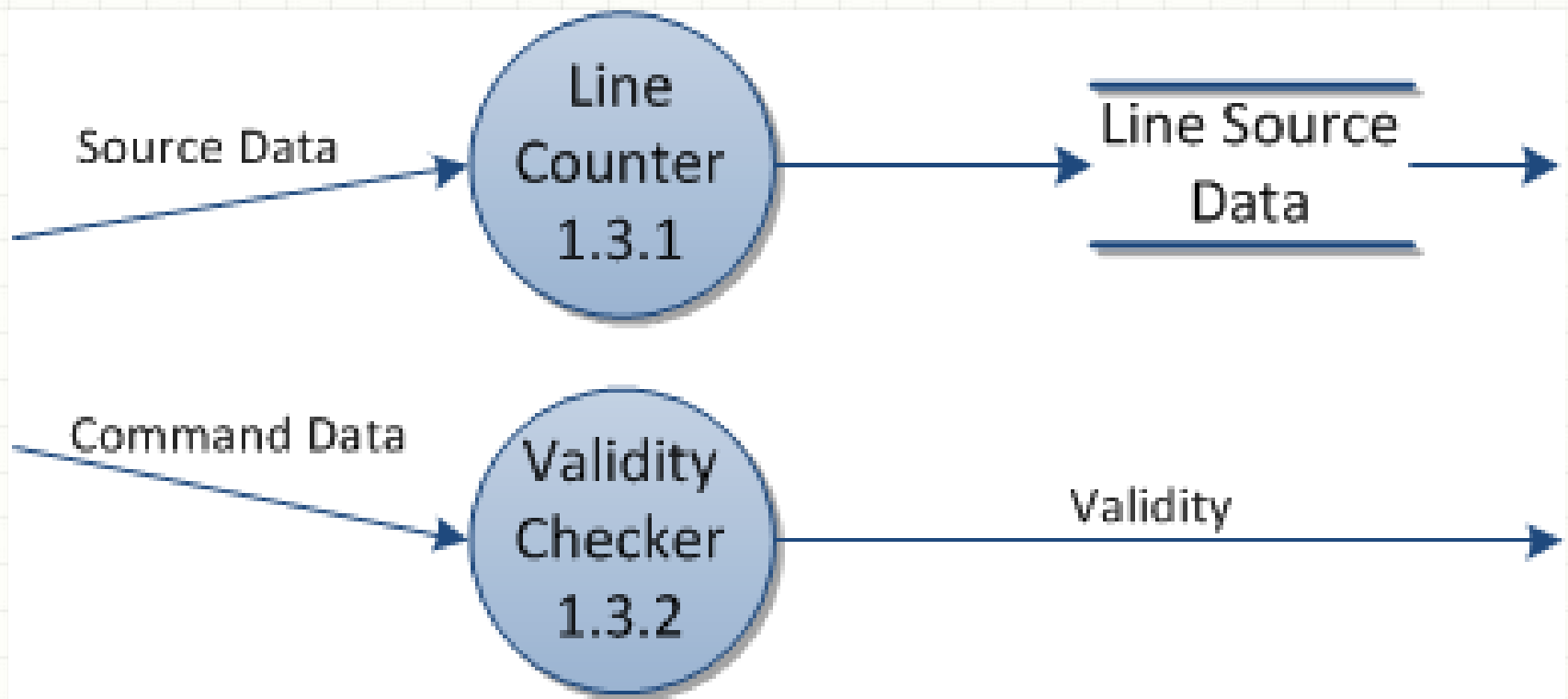
Data Flow Diagram – level 3 (2/2)
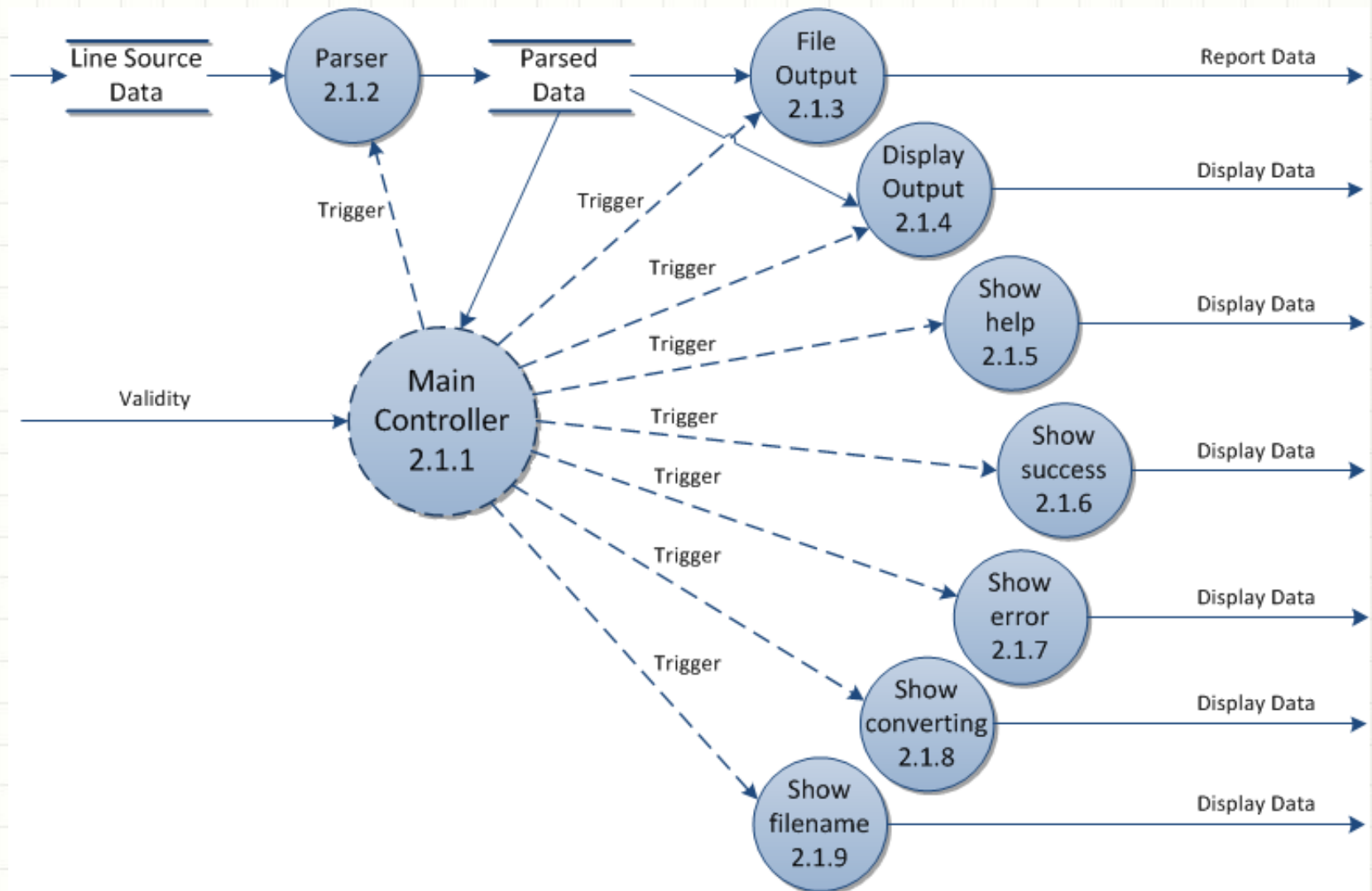
Data Dictionary – level3

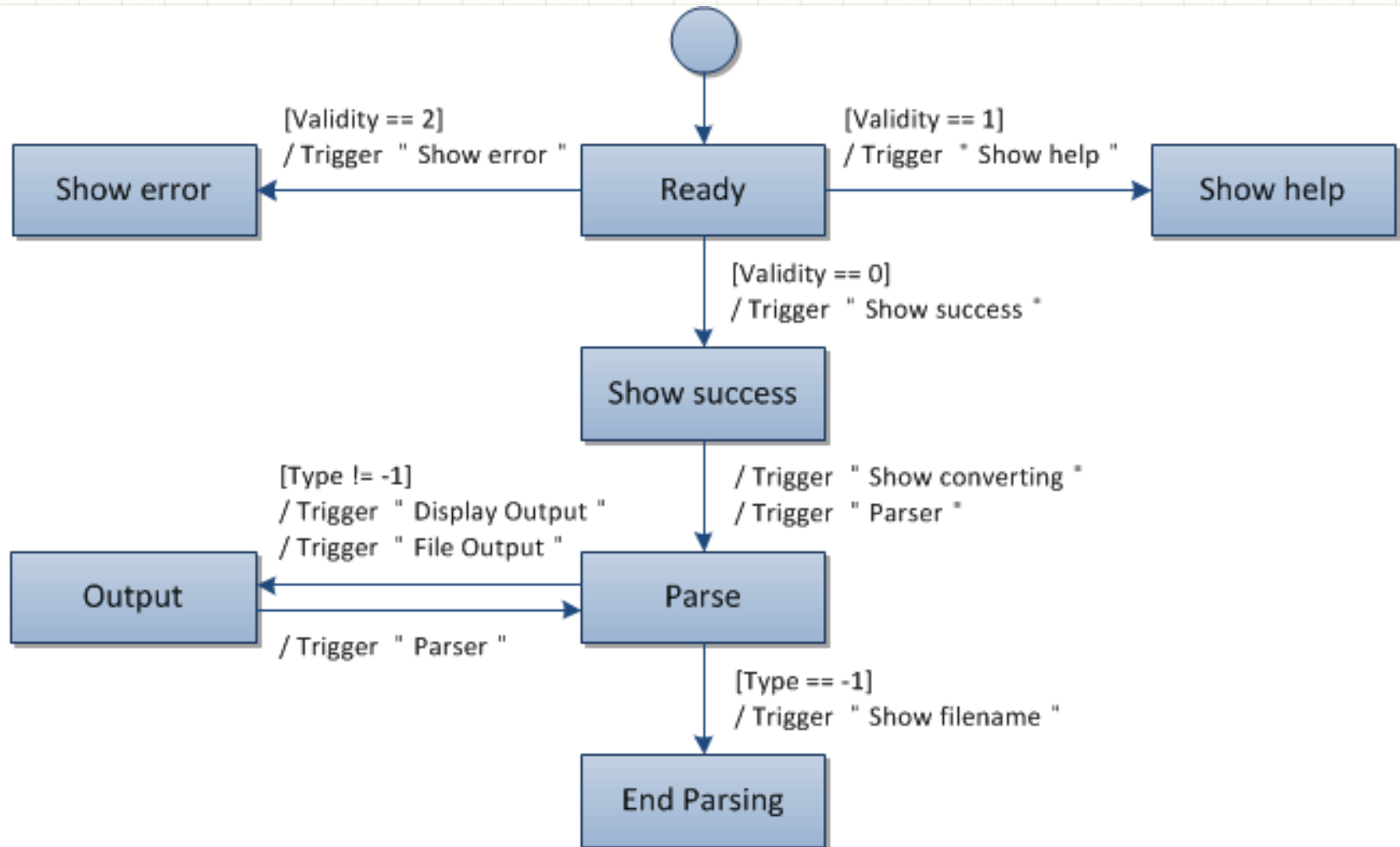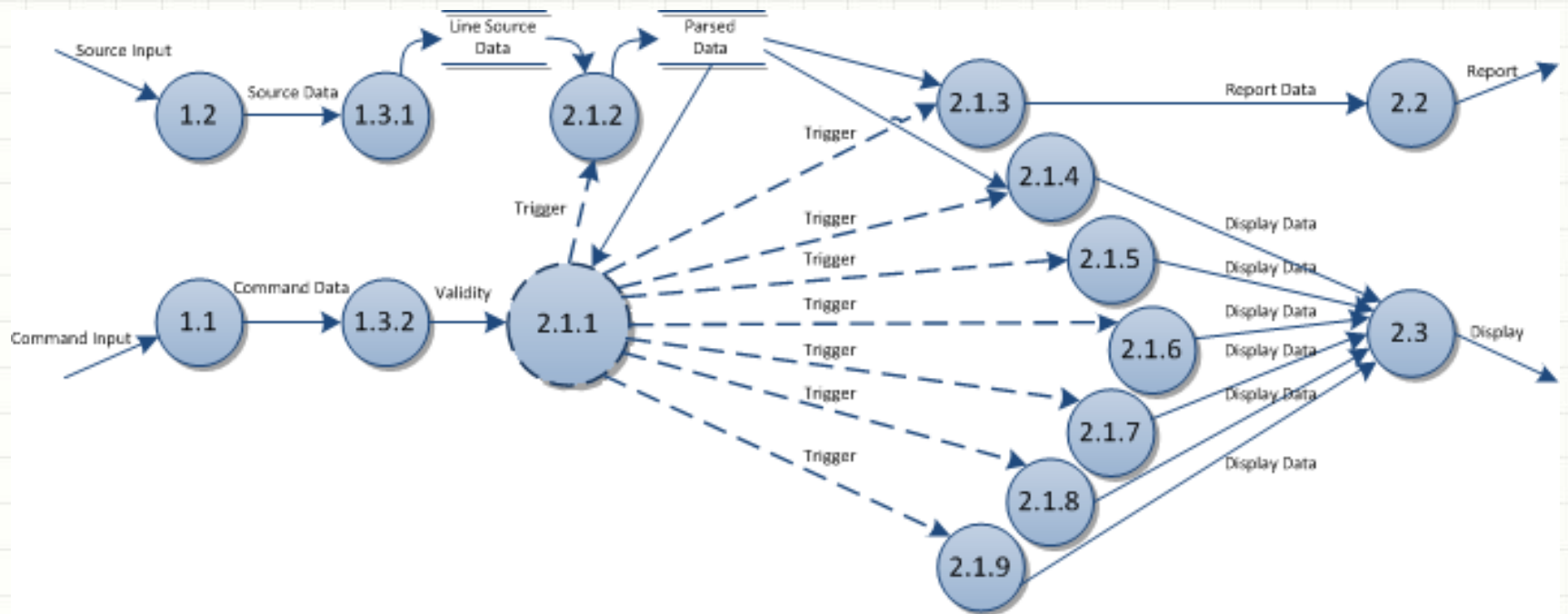| Parsed Data	Description

CFG Data	*** 'blocks' and 'edges' data of converted CFG by **Parser** ***
	If there is no more data to parse, Type value is set to -1.
	- Type/Information1/Information2/Information3/Information4/Information5
	(Integer / Integer / Integer / Integer / Integer / String)
	
	Type is (**0**: Block, **1**: Edge, **-1**: None)
	
	[1] Type == 0
	  information1 : Block Number
	  information2 : Block Type
	  (**0**: Entry Block / **1**: Exit Block / **2**: Loop Header / **3**: Block of the others)
	  information3 : Start Line Number
	  information4 : End Line Number
	  information5 : Description
	
	[2] Type == 1
	  information1 : Edge Number
	  information2 : Edge Type (**0**: Back Edge / **1**: Normal Edge)
	  information3 : Source Block Number
	  information4 : Destination Block Number
	  information5 : Not Used
	
	[3] Type == -1
	  All information is not used

Data Flow Diagram – level 4

Data Flow Diagram – Overall

Process Specification

Reference No.	1 . 1
Name	Command Interface
Input	Command Input
Output	Command Data
Process Description	Receives a **Command Input** of the ***Command Reader***, and converts it to **Command Data** that the system can make use of.

Reference No.	1 . 2
Name	Source Interface
Input	Source Input
Output	Source Data
Process Description	Reads a **Source Input** of the ***File Reader*** line by line, and converts it to **Source data**.

Process Specification

Reference No.	1 . 3 . 1
Name	Line Counter
Input	Source Data
Output	Line Data
Process Description	Numbers off the received **Source Data** in order, and saves the data to *Line Source Data*.

Reference No.	1 . 3 . 2
Name	Validity Checker
Input	Command Data
Output	Validity
Process Description	After Checking format and filename of a received **Command Data**, It assigns an integer validity value into **Validity** and sends the **Validity** to *Main Controller*.

Process Specification

Reference No.	2 . 1 . 1
Name	Main Controller
Input	Validity, CFG Data
Output	Trigger
Process Description	It is a main controller that determines CFG Generator's state based on inputs (**Validity**, and **CFG Data** in *Parsed Data*) and then makes CFG Generator command correct action by triggering corresponding process.

Reference No.	2 . 1 . 2
Name	Parser
Input	Line Data, Trigger
Output	CFG Data
Process Description	It parses a received **Line Data** to a **CFG Data** and assigns that into *Parsed Data*. In the event there's no more data to read, It delivers an integer value **-1** to *Parsed Data* in order to inform the *Main Controller* of end of parsing.

Process Specification

Reference No.	2 . 1 . 3
Name	File Output
Input	CFG Data, Trigger
Output	Report Data
Process Description	After receiving **CFG Data** from *Parsed Data*, It Output **Report Data** intended to be printed in file to *Report Interface*.

Reference No.	2 . 1 . 4
Name	Display Output
Input	CFG Data, Trigger
Output	Display Data
Process Description	After receiving **CFG Data** from *Parsed Data*, It Output **Display Data** intended to be printed on display to *Display Interface*.

Process Specification

Reference No.	2 . 1 . 5
Name	Show help
Input	Trigger
Output	Display Data
Process Description	After sending a **Display Data** relevant to *'help'* message(including command syntax) to **Display Interface**, It terminates the program.

Reference No.	2 . 1 . 6
Name	Show success
Input	Trigger
Output	Display Data
Process Description	It sends a **Display Data** relevant to *"success"* message to **Display Interface**. Its message means that C source code is inputted successfully.

Process Specification

Reference No.	2 . 1 . 7
Name	Show error
Input	Trigger
Output	Display Data
Process Description	After sending a **Display Data** relevant to *"error"* message to *Display Interface*, It terminates the program. Its message means that C source code is not inputted successfully.

Reference No.	2 . 1 . 8
Name	Show converting
Input	Trigger
Output	Display Data
Process Description	It sends a **Display Data** relevant to *"converting"* message to *Display Interface*. Its message means starting conversion.

Process Specification

Reference No.	2 . 1 . 9
Name	Show filename
Input	Trigger
Output	Display Data
Process Description	After sending a **Display Data** including the name of report file to *Display Interface*, It terminates the program.

Reference No.	2 . 2
Name	Report Interface
Input	Report Data
Output	Report
Process Description	After receiving all **Report Data**, It sorts all 'blocks' and 'edges' of data in control flow order and assigns it into **Report**. It enables CFG Generator to report with a text file by sending **Report** to *File Writer*.

Process Specification

Reference No.	2 . 3
Name	Display Interface
Input	Display Data
Output	Display
Process Description	Sends a **Display** which is converted in the practical form from a received **Display Data** to *Monitor* in order to print a data immediately on display.

Resources

- Control flow graph – Wikipedia, the free encyclopedia, http://en.wikipedia.org/wiki/Control_flow_graph

- 최은만, <Software Engineering>, 정익사, 2008, Ch3~4.

QUESTIONS ?