Introduction to System Software

Chapter 1

OVERVIEW

- A computer processes digital data.
- A user solves a problem by writing and/or running a program written in a *high-level programming language* like C.
- Inside computer, *system programs* called *compiler* and *assembler* break the user program down into assembly code (*instruction set*) and then into binary machine code.
- The machine code is processed.

Compiler

Assembler

Application software, a program in C:

```
swap (int  v[ ], int  k)
{int  temp;
        temp = v[k];
        v[k] = v[k+1];
        v[k+1] = temp;
}
```

assembly language program:

```
swap;
        muli    $2,     $5, 4
        add     $2,     $4, $2
        lw      $15,    0 ($2)
        lw      $16,    4 ($2)
        sw      $16,    0 ($2)
        sw      $15,    4 ($2)
        jr      $31
```

binary machine code:

```
00000000101000010000000000011000
00000000000110000001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

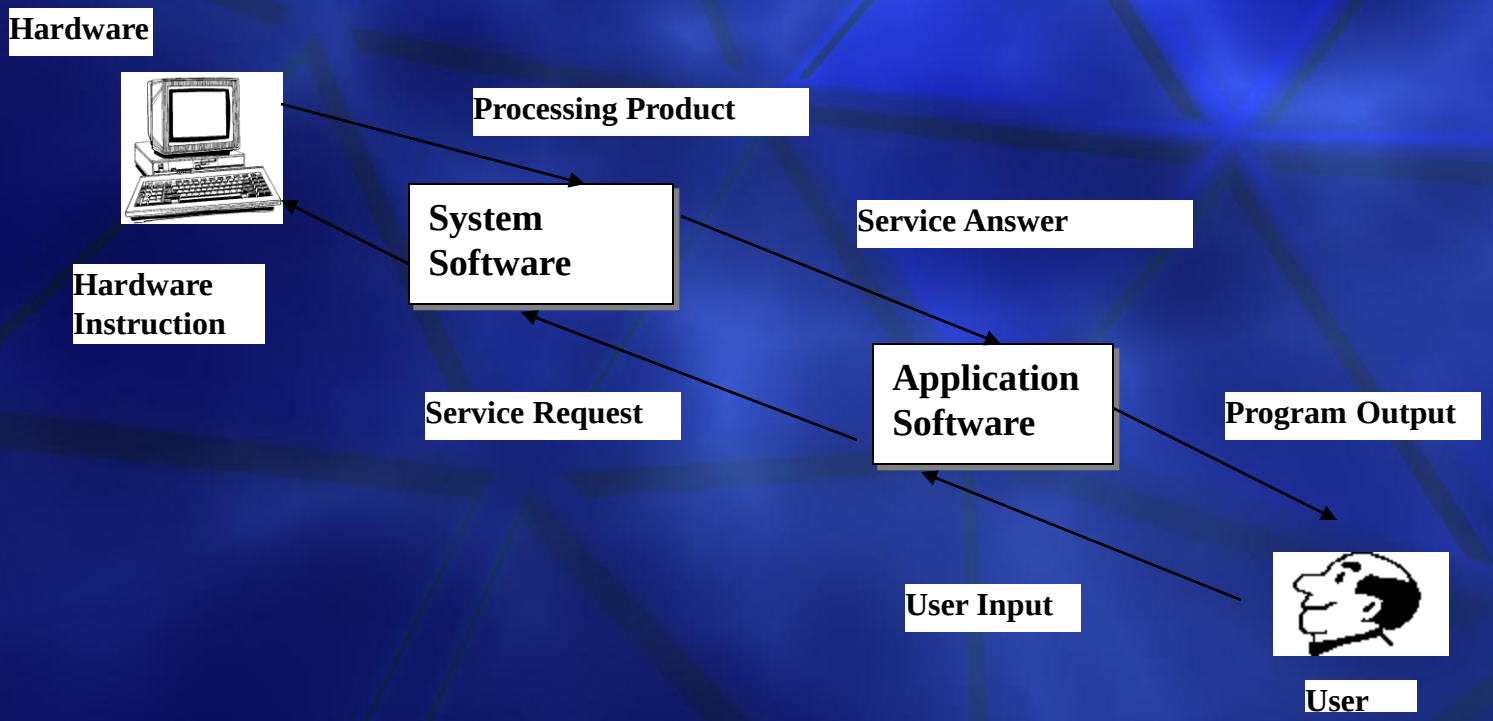
1.1 What is System Software?

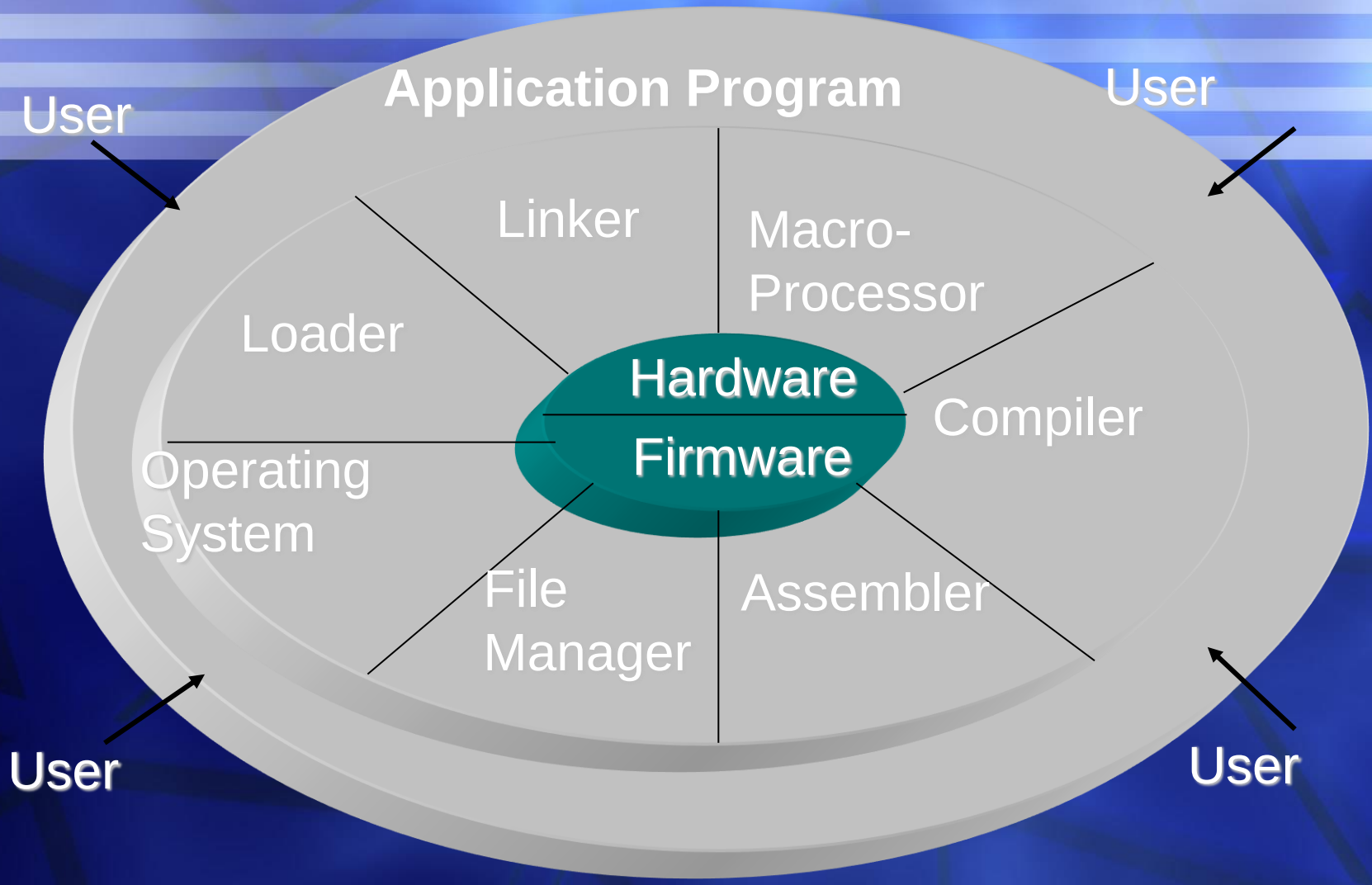
- System software consists of the program that control or maintain the operations of the computer and its devices.
- System software serves as the interface between the user, the application software and the computer hardware.

extra..

- System software:
  - Consist of a variety of programs that support the operation of a computer.
  - Makes it possible for the user to focus on an application or other problem to be solved, without needing to know the details of how the machine works internally.
- That is, systems software functions as a *bridge* between computer system hardware and the application software.
- System software coordinates the various parts of the computer system and mediates between application software and computer hardware

Relationship between user, application system, operating system and hardware.





Relationship between Hardware, System Software, Application Software and Users

extra..

Relationship

- types of software are interrelated and can be thought of as a set of nested boxes,
- each of which must interact closely with the other boxes surrounding it.
- System software—consisting of operating systems, language translators, and utility programs—controls access to the hardware.
- Application software such as……must work through the system software to operate.
- The user interacts primarily with the application software.

- Firmware
  - is a combination of software and hardware. ROMs, PROMs and EPROMs that have data or programs recorded on them are firmware
- Assembler
  - is program that change the source program in assembly language to object program in machine language.
- Loader
  - is a system software that process instruction in the object file & place the object file in a suitable location in the physical memory
- Linker
  - to perform the linking operations
- OS
  - The system software that manages and controls the computer's activities.
- Compiler
  - Program that read program (source code) written in a language (source language), & later interpret into other programming language (target language)

1.2 System Software & Application Software

Application Software

- Emphasized on solving problem and use computer as tools.
- Manage application data such as insert, delete, update, sorting.
- Provide service to end user.
- Normally written in high level language such as C, C++.
- Used only when the application is needed.
- Only perform single task.
- The program normally work one process at one time and step by step.

System Software

- Support to users and computer operation.
- Manage machine resources.
- Provide service to application software and programmer.
- Normally written in assembly language but now also written in C
- Used when the computer is turn on.
- Can perform more than one task or multitasking.
- Normally control many processes at the same time.

Differences between system software& application software

- There are two major types of software: system software and application software. Each kind performs a different function.
- System software
  - is a set of generalized programs that manage the computer's resources, such as the central processor, communications links, and peripheral devices.
  - Programmers who write system software are called *system programmers.*
- Application software
  - describes the programs that are written for or by users to apply the computer to a specific task.
  - Programmers who write application software are called *application programmers*

Function Of System Software

- Control system
- Manage machine resources
- Provides services to programmer or to application software
- Often written in assembler
- May run at all times computer is on
- Almost always a package- one off development of system software would be prohibitive
- Common handles several processes at once

Function Of Application Software

- Carries out application
- Manage application data
- Provides services to end user
- Usually written in a high level language
- Only runs for duration of a given application
- May be a software package or a bespoke (one-off) development
- Program is usually just a single step-by-step process.

1.3 Category of System Software

- Two types of system software are
  - Operating system
  - Utility programs
- Several types of utility programs are provided with an operating system.
- Other utility programs are available as programs separate from the operating system.

Function of Operating System

- Starting a computer.
- As a user interface.
- Offering the application program.

Function of Utility Programs

1. File Manager
2. Image Viewer
3. Uninstaller
4. Disk Scanner
5. Disk Defragmenter
6. Diagnostic Utility
7. Backup Utility
8. Screen Saver
9. Antivirus Program

1.4 Category of Application Software

- Productivity – Ms Office
- Education - Encyclopedia
- Entertainment - Games
- Business – Accounting and Inventory

1.5 Why using System Software

- To avoid write program in machine language.
- Get program into memory.
- For embedded system.
- Programmers productivity.
- Control of peripherals.

Book:

- System Programming & Operating Systems
- D. M. Dhamdhere

Software classification

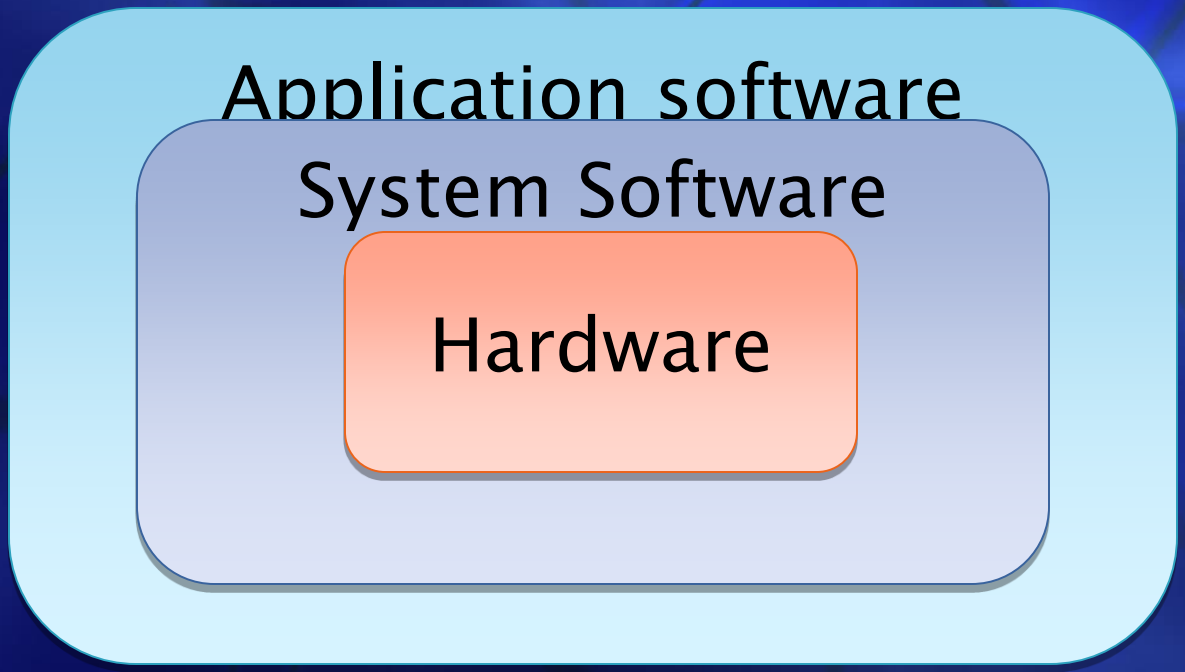
- Software can be classified into
  - System software:
  - **System software** (or **systems software**) is computer software designed to operate and control the computer hardware and to provide a platform for running application software.

    - System software is collection of software program that perform a variety of functions like IO management, storage management,generation and execution of programs etc.
      - Operating Systems
      - Compiler / Assembler (utility softwares)
      - Device Drivers

  - Application software:
    - Application software is kind of software which is designed for fulfillment specialized user requirement.
      - MS Office
      - Adobe Photoshop

Cont.

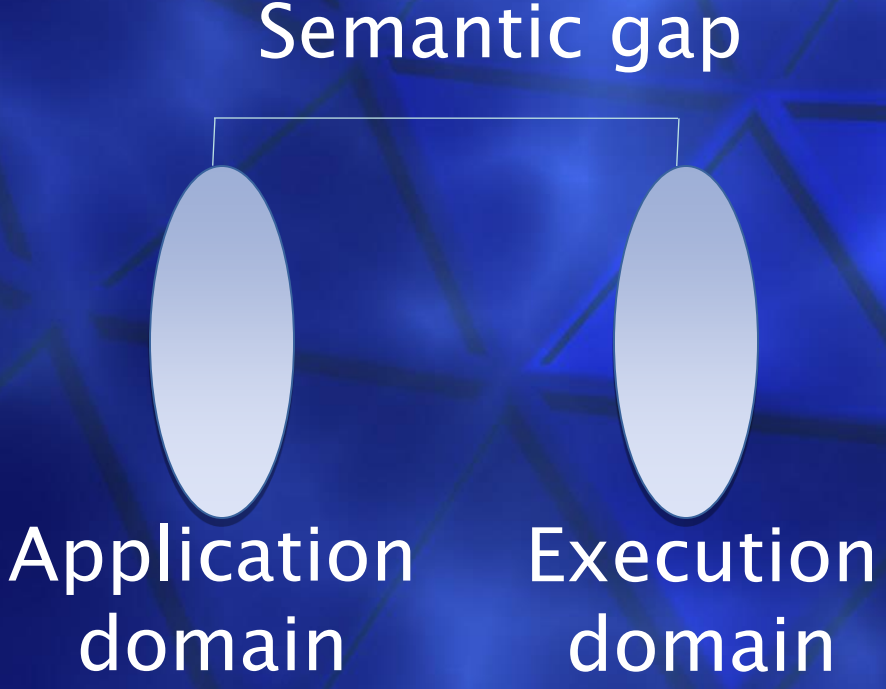
- The system software work as middleware between application software and hardware.



Chapter 1
Language Processors

System Software

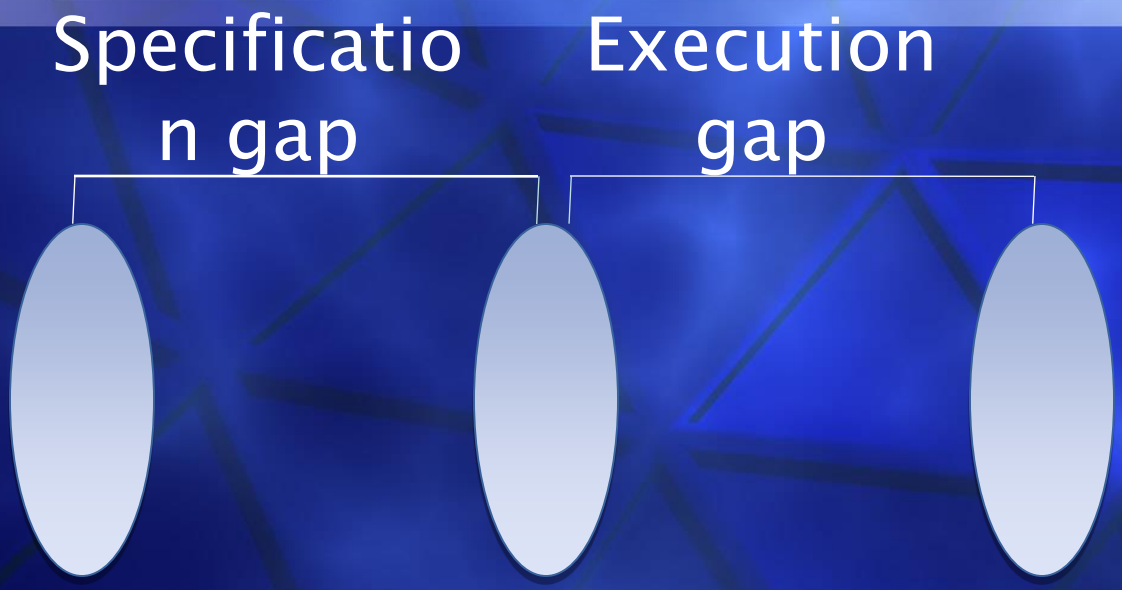
- Language processors (Why?)
  - Language processing activities arise due to the differences between the manner in which a software designer describes the ideas concerning the behavior of software and the manner in which these ideas are implemented in computer system.
  - The designer expresses the ideas in terms related to the application domain of the software.
  - To implement these ideas, their description has to be interpreted in terms related to the execution domain.



- The term semantics to represent the rules of meaning of a domain, and the term semantic gap to represent difference between the semantics of two domains.

- The semantic gap has many consequences, some of the important are
  - Large development times
  - Large development effort
  - Poor quality software.

- these issues are tackled by software engneering thru' use of methodologies and programming languages.

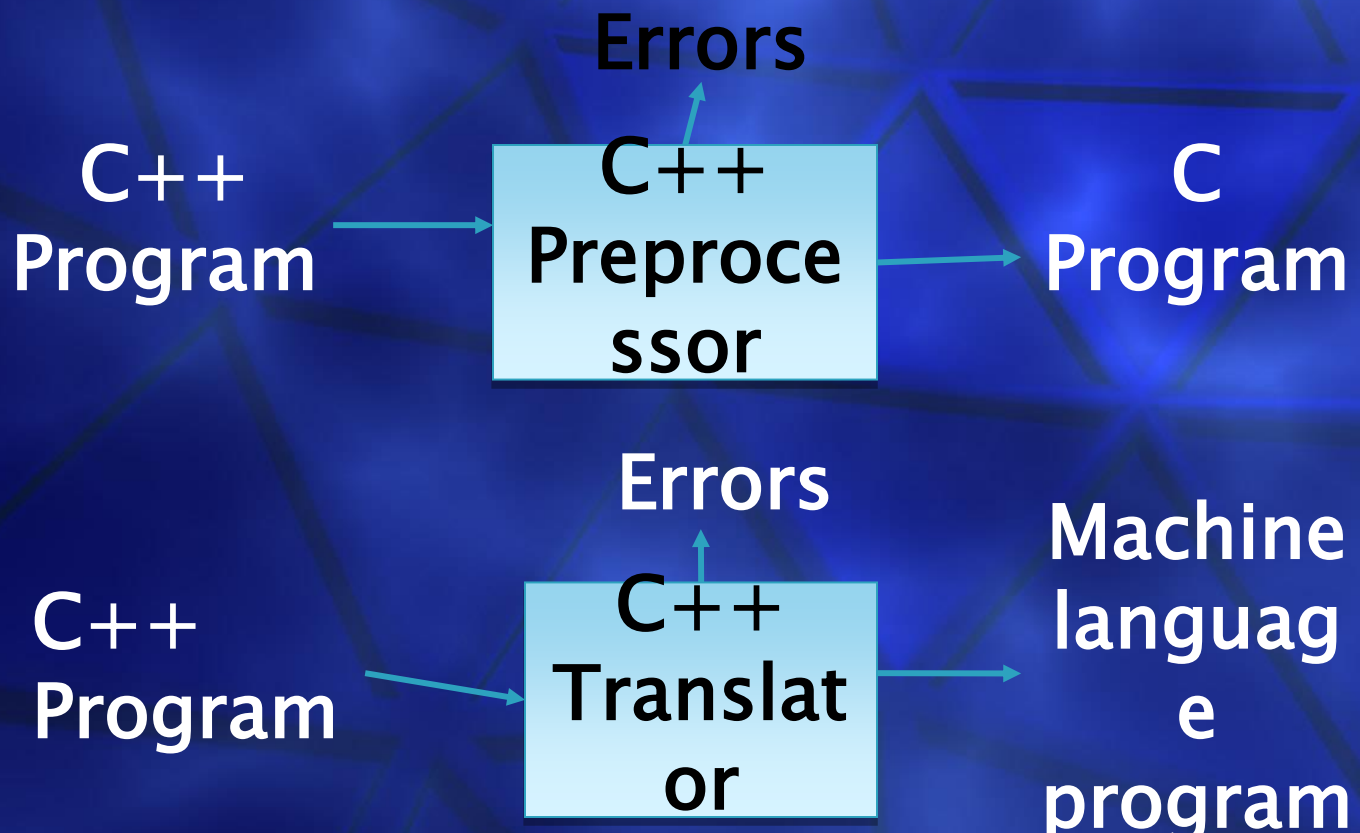


- s/w development team
- Programing language processor

Cont.

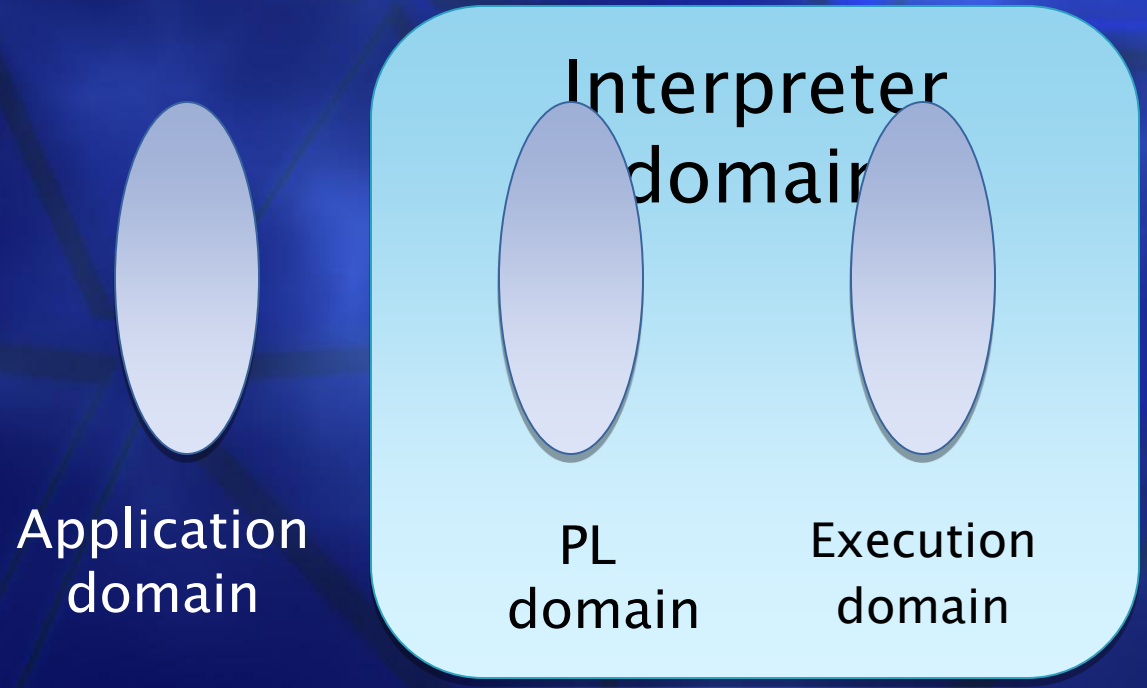
- Language processor: A language processor is software which bridge a specification or execution gap.
  - A Language Translator
  - De-translator
  - Preprocessor
  - Language migrator

Example


Errors
C++
Program
C++
Preproce
ssor
C
Program
Errors
C++
Program
C++
Translat
or
Machine
languag
e
program

Interpreter

- An interpreter is language processor which bridges an execution gap without generating a machine language program that means the execution gap vanishes totally.

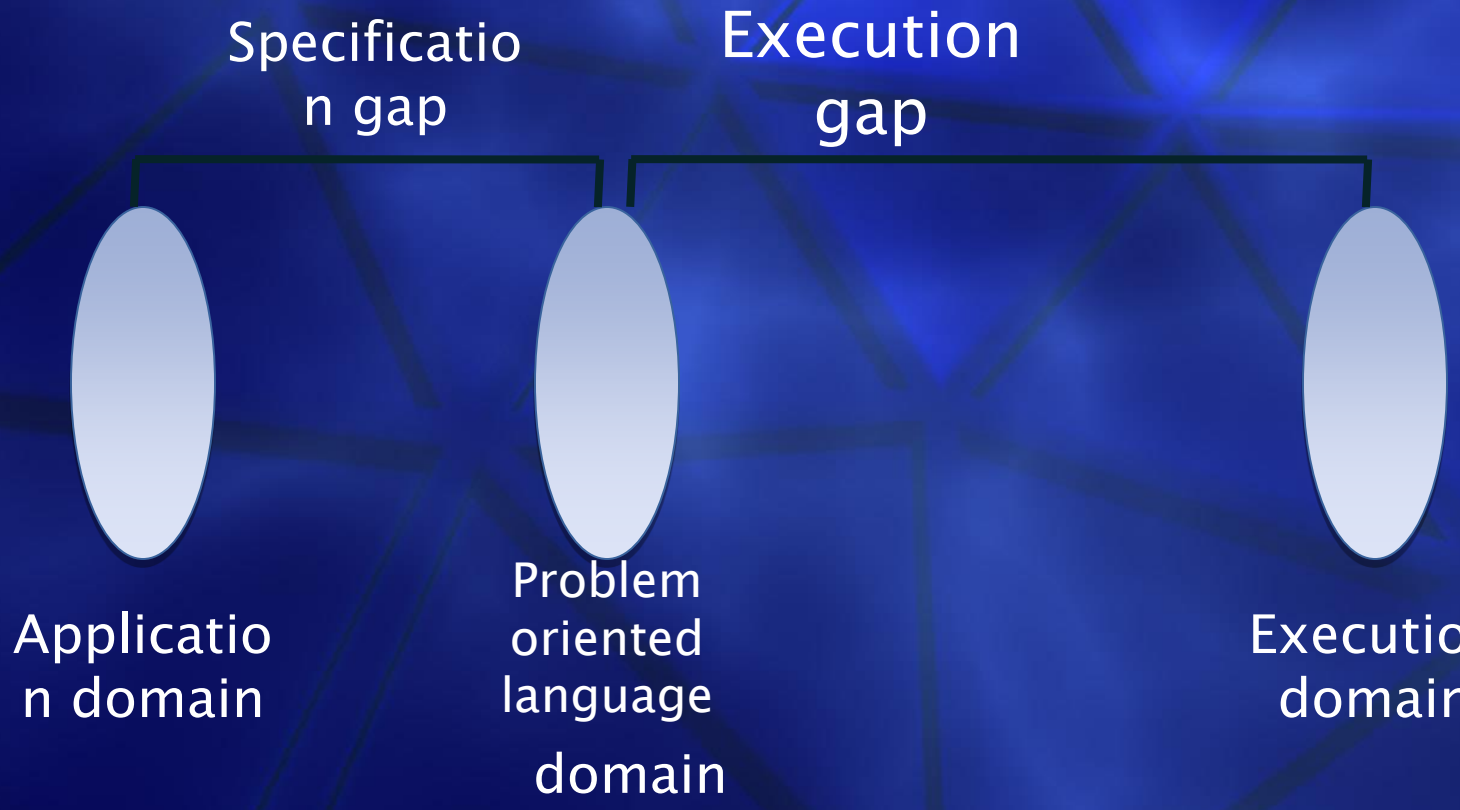


Problem oriented language

- Three consequences of the semantic gap are in fact the consequences of specification gap.
- A classical solution is to develop a PL such that the PL domain is very close or identical to the application domain.
- Such PLs can only used for specific applications, they are problem oriented languages.

Procedure oriented language

- A procedure oriented language provides general purpose facilities required in most application domains.



Language Processing Activities

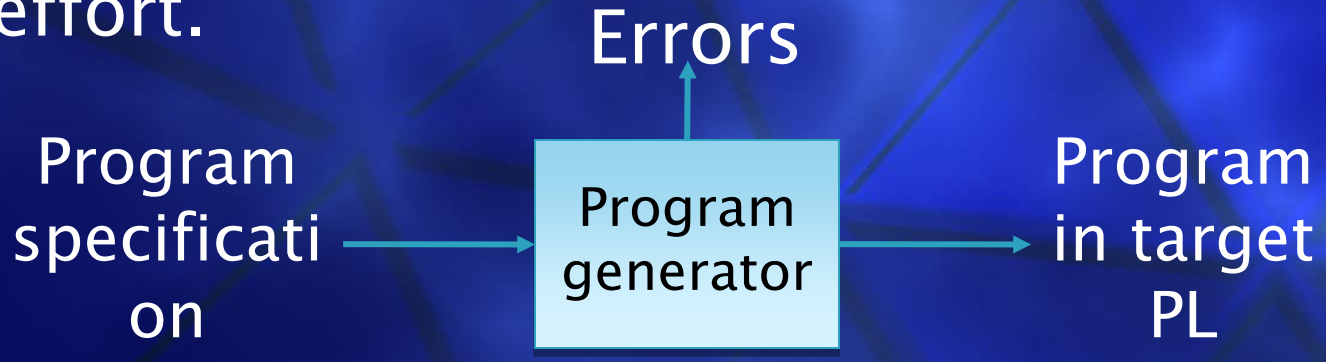
- Fundamental activities divided into those that bridge the specification gap and execution gap.
  - Program generation activities
  - Program execution activities

Cont.

- Program generation activities
  - A program generation activity aims at automatic generation of a program.
  - A source language is a specification language of an application domain and the target language is procedure oriented PL.
  - Program generator introduces a new domain between the application and PL domain , call this the program generator domain.
  - Specification gap now between Application domain and program generation domain, reduction in the specification gap increases the reliability of the generated program.

Cont.

- This arrangement also reduces the testing effort.

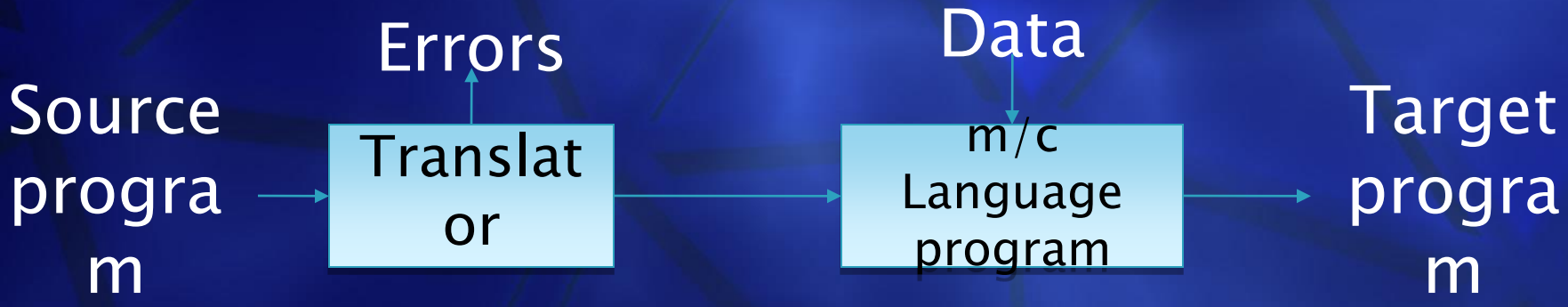


Cont.

- Program Execution
  - Two popular model
    - Program Translation
    - Program Interpretation

Cont.

- The program translation model bridges the execution gap by translating a sources program into program in the machine or assembly language of the computer system, called target program.



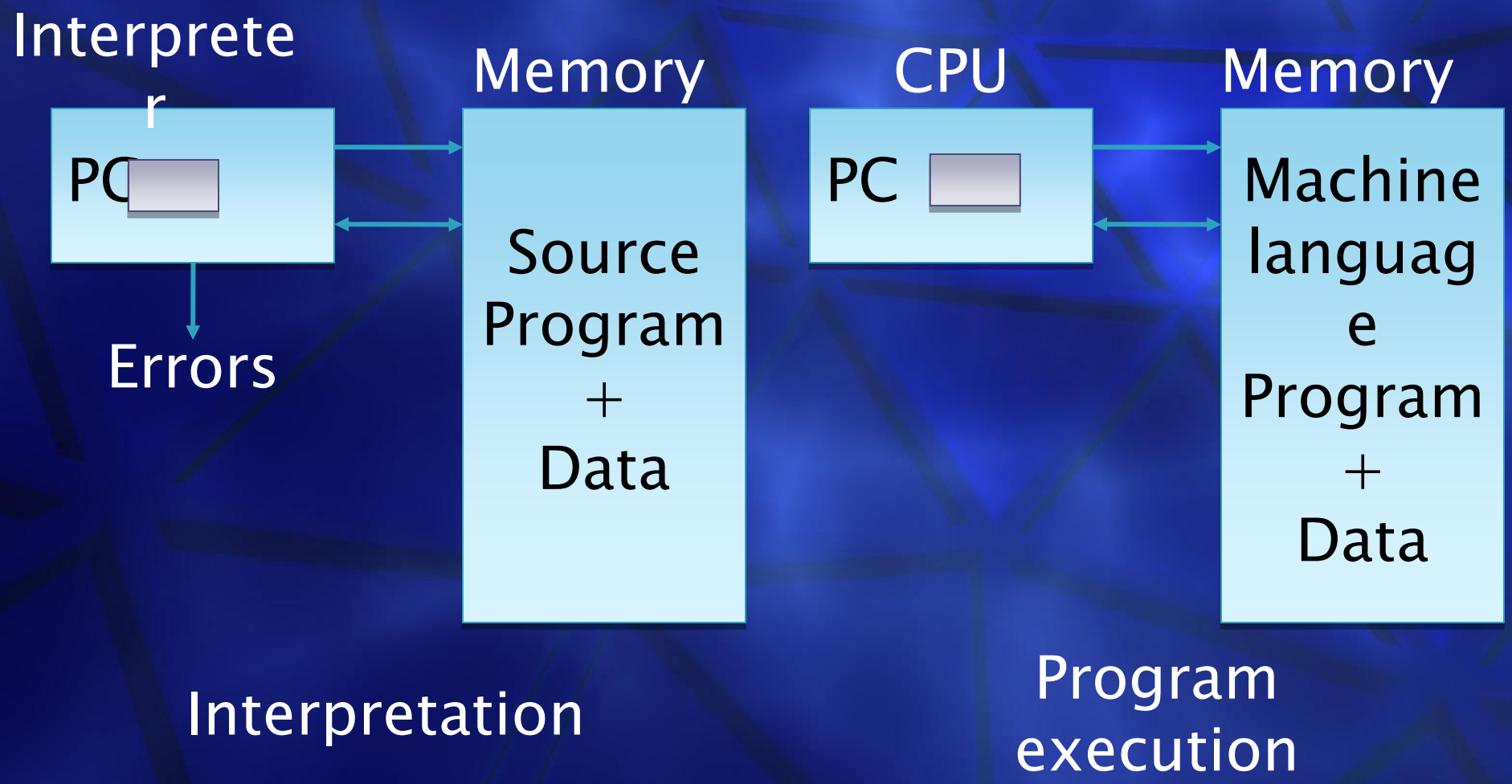
Cont.

- Characteristics of the program translation model:
  - A program must be translated before it can be executed
  - The translated program may be saved in a file. The saved program may be executed repeatedly.
  - A program must be retranslated following modifications.

Cont.

- **Program interpretation:**
- during interpretation interpreter takes source program statement, determines its meaning and performs actions which implement it.
- The function of an interpreter is same as the execution of machine language program by CPU.

Cont.


Interpreter
PC
Errors
Memory
Source Program + Data
CPU
PC
Memory
Machine language Program + Data
Interpretation
Program execution

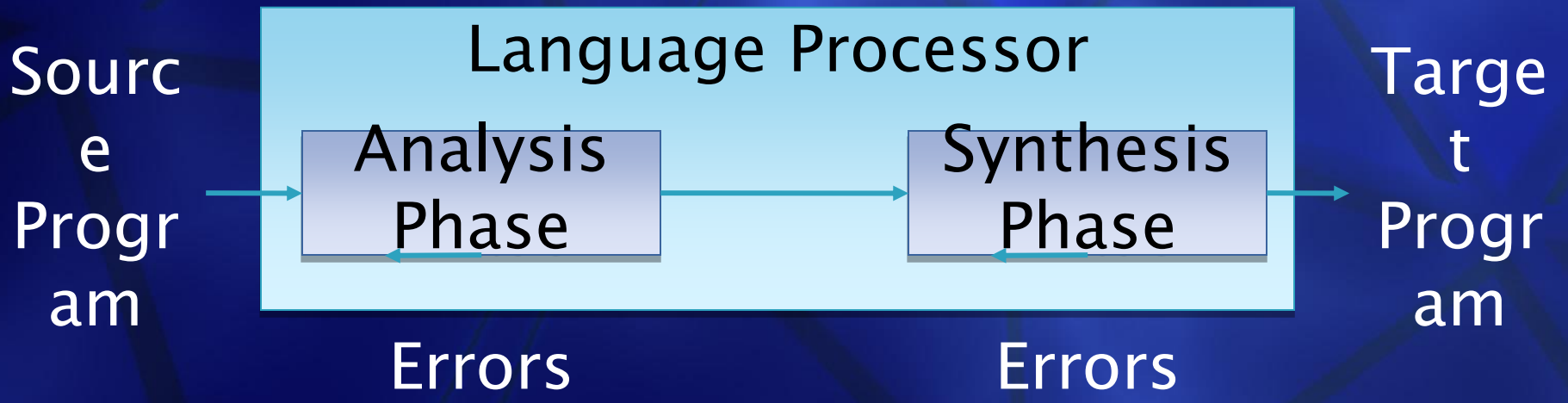
Cont.

- Characteristics
  - The source program is retained in the source form itself, no target program form exists,
  - A statement is analyzed during its interpretation.
- Comparison
  - In translator whole program is translated into target and if modified the source program, whole source program is translated irrespective to size of modification.
  - That not the in case of interpreter, interpretation is slower than execution of m/c language program.

Fundamental of Language Processoing

- Language Processing = Analysis of SP + Synthesis of TP.
- Analysis phase of Language processing
- **Lexical rules** which govern the formation of valid lexical units in the source language.
- **Syntax rules** which govern the formation of the valid statements in the source language.
- **Semantic rules** which associate meaning with the valid statements of the language.

- The synthesis phase is concerned with the construction of target language statement which have same meaning as a source statement.
  - Creation of data structures in the target program(**memory allocation**)
  - Generation of target code.(**Code generation**)



```
percent_profit  = (profit*100) / cost_price;
```

Lexical Analysis

Syntax Analysis

Semantic Analysis

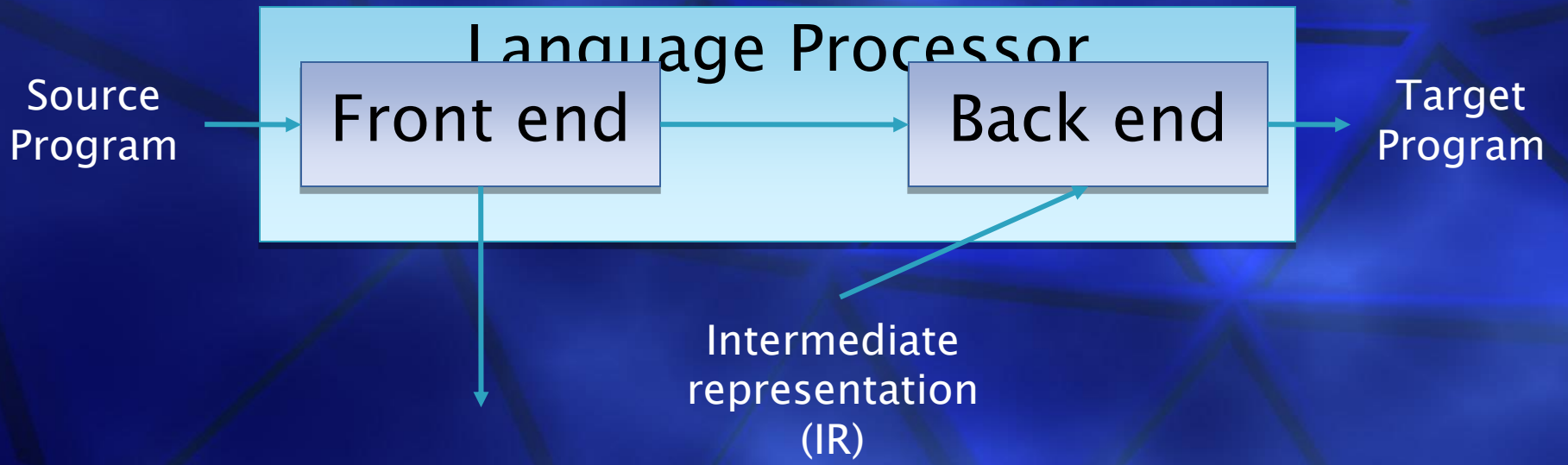
Cont.

- **Forward references**: for reducing execution gap the language processor can be performed on a statement by statement basis.
- Analysis of source statement can be immediately followed by synthesis of equivalent target statements. But this may not feasible due to :**Forward reference**
- “A forward reference of a program entity is a reference to the entity which precedes its definition in the program.”

Cont.

- **Language processor pass**: “A language processor pass is the processing of every statement in a source program, or its equivalent representation, to perform a language processing function.”
- **Intermediate representation(IR)**: “An intermediate representation is a representation of a source program which reflects the effect of some, **but not all**, analysis and synthesis tasks performed during language processing.”

Cont.

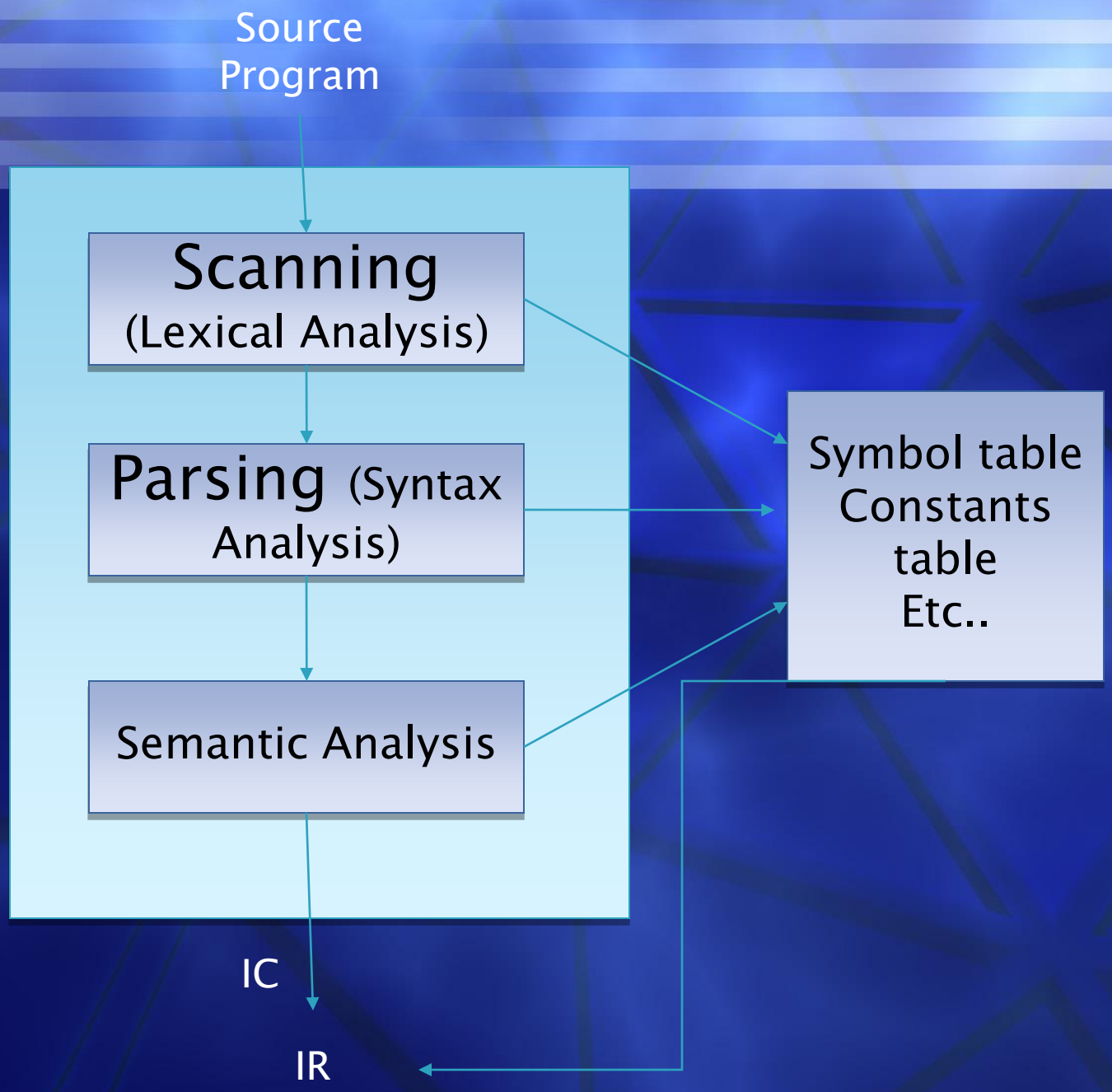

Language Processor
Source Program
Front end
Back end
Target Program
Intermediate representation (IR)

Cont.

- **Semantic Action**: “All the actions performed by the front end, except lexical and syntax analysis, are called semantic action.
  - Checking semantic validity of constructs in SP
  - Determining the meaning of SP
  - Constructing an IR

Toy Compiler

- The Front End
  - The front end performs lexical, syntax and semantic analysis of the source program, each kind of analysis involves the following functions:
    - Determine validity of source statement from the view point of the analysis.
    - Determine the ‘content’ of a source statement
      - For lexical, the lexical class to which each lexical unit belongs.
      - Syntax analysis it is syntactic structure of source program.
      - Semantic analysis the content is the meaning of a statement.
    - Construct a suitable representation of source statement for use by subsequent analysis function/synthesis phase.


Source Program
Scanning (Lexical Analysis)
Parsing (Syntax Analysis)
Semantic Analysis
Symbol table
Constants table
Etc..
IC
IR

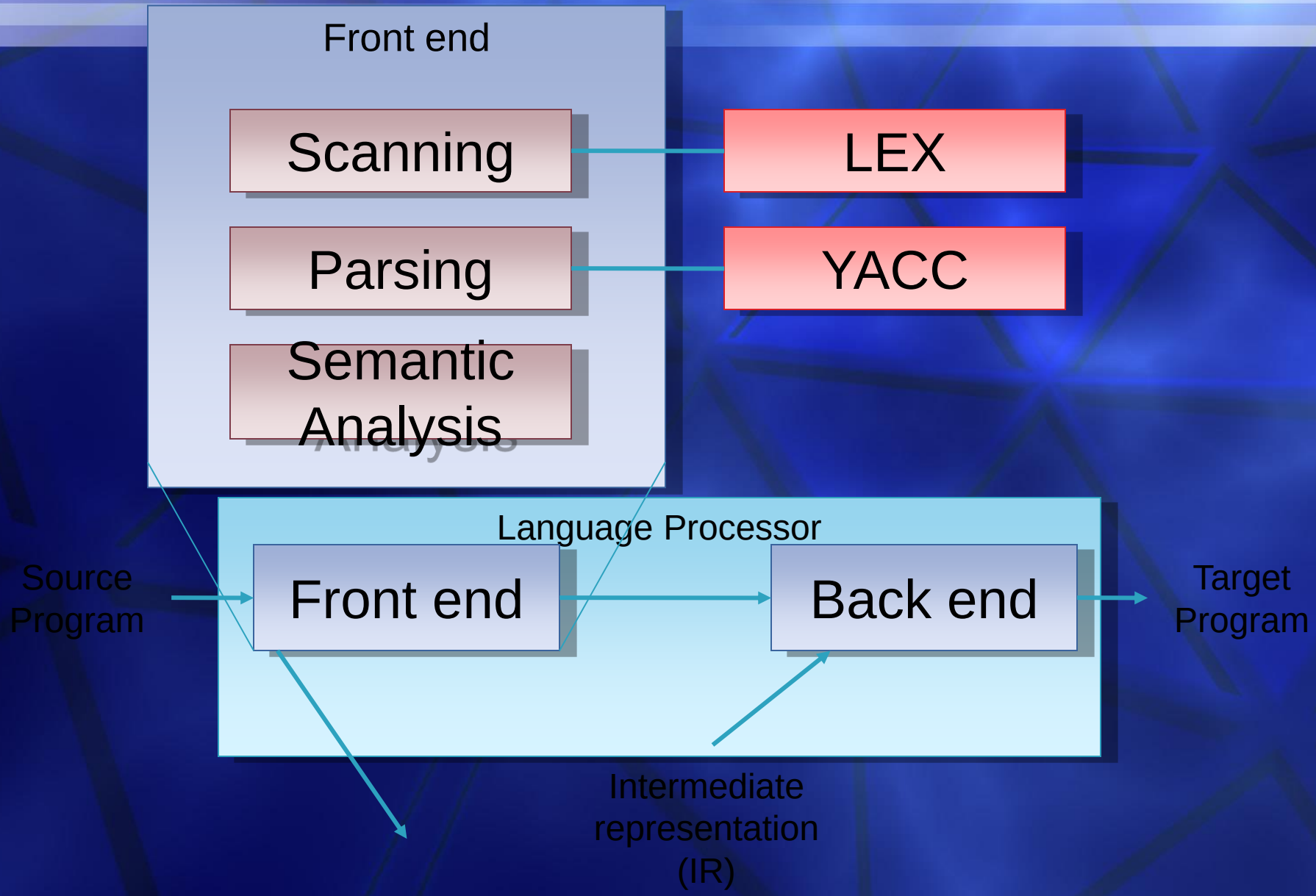
- Out put of front end produced two components: (IR)
  - Table of information
    - The symbol table which contain information concerning all identifier used in the source program.
  - An intermediate code (IC) which is a description of the source program.
    - The IC is a sequence of IC units, each IC unit representing the meaning of one action in SP. IC units may contain references to the information in various table.

Cont.

- Lexical Analysis (Scanning):
  - Lexical analysis identifies the lexical units in source statement
  - it then classifies the unit into different classes
  - Ex. Id's, Constant reserved id's etc. and enters them into different tables.
  - This classification may be based on the nature of a string or on the specification of the source language.
  - Lexical analysis build descriptor called token, for each lexical unit. It contain two fields class code and number in class
    - Class code: identifies the class to which a lexical unit belongs.
    - Number in class: entry number of lexical unit in the relevant table.

- Syntax Analysis(Parsing)
  - Syntax analysis process the string token built by lexical analysis to determine the statement class e.g. assignment statement, if statement, etc.
  - Syntax Analysis builds an IC which represents the structure of the statement.
  - IC is passed to semantic analysis to determine the meaning of the statement.

- Semantic analysis
  - Semantic analysis of declaration statements differs from the semantic analysis of imperative statements.
  - The former results in addition of information to the symbol table e.g. Type, length and dimensionality of variables.
  - The latter identifies the sequence of action necessary to implement the meaning of a source statement.
  - When semantic analysis determines the meaning of a sub tree in the IC, it adds information to a table or adds an action to sequence of the action.
  - The analysis ends when the tree has been completely processed. The update tables and the sequence of action constitute the IR produced by the analysis phase.


Front end
Scanning
Parsing
Semantic Analysis
LEX
YACC
Language Processor
Source Program
Front end
Back end
Target Program
Intermediate representation (IR)