International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 04 Issue: 07 | July -2020 ISSN: 2582-3930

PROGRAMMING STANDARD PARSER COMPILER

Sable Nilesh[1], Akshay Rajput[2], Madhuri Bobade[3], SonaliWadikar*[4]

Computer Engineering, JSPM's ICOER, Pune, India

Abstract— This project application consists of team management, setting the programming standards according to organization requirement and during development of the product, the system is cross verifying the stetted programming standards. The verification system is our parser compiler system (PCS).In PCS, the developed code is parsed which contains lexical analyser process, semantic analyser process. Lexical analyser process happens after successful compilation of developed code. Any IT organization worked on various projects using many technologies. Each technology (programming languages) has its pro's and con's.

Keywords—(Programming Standards (PS), Compiler, Parser, String tokenize, Lexical analyser, Semantic analyser, Programming Standard analyser.

INTRODUCTION

In today's world, there are many IT industries working on various projects using many types of technologies. There are number of types of programming languages available for project development But when there is involvement of humans, there should be some rules and regulations set to follow. The intention is not only to complete the given task or project within the setted deadline but also that project or work should be done with some standards, which is more important. While working in IT industry, only completing the given project within the given time period is not important but completing the task with good outline is important. Outline is in the sense of following the standards which are already setted by the organization or that also can be setted by the higher authority of the organization. For eg. Company owner, project manager, etc.

Program standard compiler is working on divide, check and uploads strategy. This can say as an application which will be worked as functional tester of already developed program.

In PCS, already developed program will uploaded then will be comparing with the standards which are setted by the organization or the higher authority of the organization like project manager or company owner or it can be developer also. Then that program will get scanned and after successful completion of scanning and comparison of the program that checked program will get saved in database. For maintaining the work ethics and maintaining the work standard this application will help the organizations and companies better.

RELATED WORK:

As this application is developed using various programming languages and various types of software's so we have studied about the installation of various software's and packages needed for installation of that software's. Also studied various types of new concept regarding the project like Lexical analyzer, Semantic Analyzer about various organizations standards to follow in their working environment. In one of research performance characteristics of data mining applications using the Nu-Mine Bench written in C/C++ with the exception of AFI GETI are

parallelized with open directives. Choose the example of Intel as the order architecture processor for our study for two main reasons, first Itanium arguably relies on sophisticated compiler optimizations to achieve its performance making the proper platform for evaluating the effectiveness of compiler directed prefetching. Itanium sets a hardware performance counters that enable us to study performance characteristics without cumbersome simulations or intrusive instrumentations. 2.1. Phases of Compiler: In order to construct process the compiler is implemented in phase's lexical analysis syntax analysis code generation. Lexical analysis in grouping the input string into words known as a lexeme or lexical token is string of input characters which is taken as a unit and passed on to the next phase of compilation. Key words - While, Void, if. For Identifier - Declared by the programmer Operator - - +,-,*,/,=,. The output of the lexical phase is a stream of tokens corresponding to the words, builds tables which are used by subsequent phases of compiler. Symbol table stores all Ch. Raju ET al, International Journal of Computer Science and Mobile Computing Vol.2 Issue. 10, October- 2013, pg. 115-125 identifiers used in the source program including relevant information and attributes of the identifiers. Syntax Analysis Phase is a parser is critical to understand both phase and the study language in general will check for proper syntax issue appropriate error messages and determine the underlying structure of the source program. Syntax tree each interior node represents an operator or control structure and each leaf node represents an operand. A statement if (expr) stmt1 else stmt 2 could be implemented as a node having three children one for conditional expression one for the true part and one for else statement. While control structure would have two children one for loop condition and one for statement to be repeated, compound statement could have an unlimited number of children one for each statement in the compound statement. Assembly language aware that the computer is capable of executing only a limited number of primitive operations on operands with numeric memory address all encoded as binary values. Code generation translated to machine language instructions or assembly language which translated to machine language instruction in which the assembler invoked to produce the object program. For target machines with several CPU registers the code generator is responsible for register allocation. Local optimization is optional is needed only to make the object program more efficient which involves examining sequences of instructions to find unnecessary or redundant instructions. Local optimization is often called machine dependent optimization source program result in Load operand into a register, add the other operand to the register, store the result.

PROPOSED SYSTEM:

In Parser Compiler System (PCS), The PL provides a compiler which checks the programming languages used. Programming structure by the inbuilt development kit provided by the programming itself. Although, the compiler parses the program and we get the successful output but, the organization has its own standards in the program which is not possible for the compiler to check each and every organization standards. Due to this limitation, we are proposing the programming parser compiler system. Parser compiler system will parse the program according to the programming standards stetted by the organization.

In regular execution process of program, compiler checks the functional testing of the program whether it has any syntactical error or not. If the program is error free then only it will shows you the correct output. In proposed system also, we take some examples of standards imported on the system which should be followed by the organization. It will also manage the data of the different projects and the people working on that project.

SYSTEM IMPLEMENTATION AND DEVELOPMENT:

In computer technology, a parser is a program, usually part of acompiler, that receives input in the form of sequential source program instructions, interactive online commands, markup tags, or some other defined interface and breaks them up into parts (for example, the nouns

(objects), verbs (methods), and their attributes or options) that can then be managed by other programming (for example, other components in a compiler). A parser may also check to see that all input has been provided that is necessary. A parser is a compiler or interpreter component that breaks data into smaller elements for easy translation into another language. A parser takes input in the form of a sequence of tokens or program instructions and usually builds a data structure in the form of a parse tree or an abstract syntax tree.

Parsing, syntax analysis, or syntactic analysis is the process of analyzingastringofsymbols, either innatural language,computerlanguagesordata structures, conforming to the rules of aformal grammar. The term parsing comes from Latin pars, meaningpart(of speech).[1]

The term has slightly different meanings in different branches oflinguisticsandcomputerscience. Traditionalsentenceparsing is often performed as a method of understanding the exact meaning of a sentence or word, sometimes with the aid of devices such assentence diagrams. It usually emphasizes the importance of grammatical divisions such assubjectandpredicate.

Withincomputationallinguisticsthe term is used to refer to the formal analysis by a computer of a sentence or other string of words into its constituents, resulting in aparsetreeshowing their syntactic relation to each other, which may also containsemanticand other information. Some parsing algorithms may generate a parse forest or list of parse trees for asyntacticallyambiguousinput.[2]

The term is also used inpsycholinguisticswhen describing language comprehension. In this context, parsing refers to the way that human beings analyze a sentence or phrase (in spoken language or text) "in terms of grammatical constituents, identifying the parts of speech, syntactic relations, etc."[1]This term is especially common when discussing what linguistic cues help speakers to interpretgarden-path sentences.

Within computer science, the term is used in the analysis ofcomputer languages, referring to the syntactic analysis of the input code into its component parts in order to facilitate the writing ofcompilersandinterpreters. The term may also be used to describe a split or separation.

Parser :

A parser is a software component that takes input data (frequently text) and builds adata structure– often some kind ofparsetree,abstract syntax treeor other hierarchical structure, giving a structural representation of the input while checking for correct syntax. The parsing may be preceded or followed by other steps, or these may be combined into a single step. The parser is often preceded by a separatelexicalanalyzer, which creates tokens from the sequence of input characters; alternatively, these can be combined inscannerlessparsing. Parsers may be programmed by hand or may be automatically or semi-automatically generated by aparsergenerator. Parsing is complementary totemplating, which produces formatted output. These may be applied to different domains, but often appear together, such as thescanf/printfpair, or the input (front end parsing) and output (back end code generation) stages of a compiler.

The input to a parser is often text in somecomputer language, but may also be text in a natural language or less structured textual data, in which case generally only certain parts of the text are extracted, rather than a parse tree being constructed. Parsers range from very simple functions such asscanf, to complex programs such as the frontend of aC++ compilerortheHTMLparser of aweb browser. An important class of simple parsing is done usingregularexpressions, in which a group of regular expressions defines aregularlanguageand a

regular expression engine automatically generating a parser for that language, allowing pattern matching and extraction of text. In other contexts regular expressions are instead used prior to parsing, as the lexing step whose output is then used by the parser.

The use of parsers varies by input. In the case of data languages, a parser is often found as the file reading facility of a program, such as reading in HTML orXMLtext; these examples aremarkuplanguages. In the case ofprogramming languages, a parser is a component of acompilerorinterpreter, which parses

thesourcecodeofacomputer programming languageto create some form of internal representation; the parser is a key step in thecompiler frontend. Programming languages tend to be specified in terms of adeterministic context-free grammarbecause fast and efficient parsers can be written for them. For compilers, the parsing itself can be done in one pass or multiple passes – seeone-pass compilerandmulti-pass compiler.

The implied disadvantages of a one-pass compiler can largely be overcome by addingfix-ups, where provision is made for code relocation during the forward pass, and the fix-ups are applied backwards when the current program segment has been recognized as having been completed. An example where such a fix-up mechanism would be useful would be a forward GOTO statement, where the target of the GOTO is unknown until the program segment is completed. In this case, the application of the fix-up would be delayed until the target of the GOTO was recognized. Conversely, a backward GOTO does not require a fix-up, as the location will already be known.

Context-free grammars are limited in the extent to which they can express all of the requirements of a language. Informally, the reason is that the memory of such a language is limited. The grammar cannot remember the presence of a construct over an arbitrarily long input; this is necessary for a language in which, for example, a name must be declared before it may be referenced. More powerful grammars that can express this constraint, however, cannot be parsed efficiently. Thus, it is a common strategy to create a relaxed parser for a context-free grammar which accepts a superset of the desired language constructs (that is, it accepts some invalid constructs); later, the unwanted constructs can be filtered out at thesemantic analysis(contextual analysis) step.

For example, inPythonthe following is syntactically valid code:

```
x = 1 Print(x)
```

The following code, however, is syntactically valid in terms of the context-free grammar, yielding a syntax tree with the same structure as the previous, but is syntactically invalid in terms of thecontext-sensitive grammar, which requires that variables be initialized before use:

```
x = 1 Print(y)
```

Rather than being analyzed at the parsing stage, this is caught by checking the values in the syntax tree, hence as part of semantic analysis: contextsensitive syntax is in practice often more easily analyzed as semantics.

MATHEMATICAL MODEL:

INPUT: Consider Program Sentence as S and Program Standard as, $\sum_{n=1}^{N} K$
Where,
K is Program Standard and n is number of program standards.
OUTPUT:

S C K
Equation 1: $K = |N_1| + |N_2| + \text{-----} |N_n|$
Where, N is number of Programming Standards This step is required to check all program standards.
Equation 2: $S \subset \sum_{n=1}^{N} K = \emptyset$
If S is not consisting of all the K (program standard inputs)
then S does not follow complete program standards.
Equation 3: $S \supset \sum_{n=1}^{N} K \neq \emptyset$
If S is consisting of all K(program standard inputs) then S follows complete program standards.

PROGRAM STANDARD PARSER ALGORITHM:

INPUT: Java Program Statement and Program Standards.
Input Program Standards:
1. Variables without final is small chars.
2. Variables with final is capital.
3. Class names are always started with capital.
4. Method names are always started with small.
5. Package names are small letters
6. Interface names are started with capital letters.

OUTPUT: Program Verified According to

Program Standards.

Steps:

Program parser compiler scans the java program. This process is lexical analysis.

String tokenize process takes place wherein the scanned text is broken into ―tokens‖.

Program standards are scanned (whichever applicable).The tokens are checked syntactically. Semantic analysis process takes place where parser parses according to rules of program standards that are provided.

Intermediate code result generation occurs which stores results from semantic analysis. Program standard parser compiler results are displayed.

JAVA PARSER COMPILER PROJECT WORKFLOW:

This project is kept in view related to the programming standards or structures that is set by individual persons or companies. Although each and every development kit performs programming standards such as based on keywords, structures and functions. Our project is superset of the existing development kit and performs additional programming standards. We have developed a parser which performs parsing of programs based on features such as

- Package names small
- Class names initial capital
- Interface names initial capital
- Variables small
- Final variables capital
- Method names start with small

Also, we have provided management facility to the project keeping in mind organization needs. There are three users/actors in the project as company Owner, team lead and developer as members.

RESULT:

Using programming standard parser compiler, organization programming standard setted are met accurately.

CONCLUSION:

The main conclusion to be drawn from the above explanation and result is that (1) the influence of the programming standard of the program language can be hardly being underestimated. By uses of programming standard parser compiler, we can guarantee that programming standard will be (2) strictly followed according to organization requirement. Further make program standard parser compiler algorithm can be used a crossed different programming language platform based on program languag (3) e structure.

REFERENCES:

1)Y. Artzi, D. Das, and S. Petrov, ―Learning compact lexicons for ccg semantic parsing,‖ in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar:
Association for Computational Linguistics, October 2014, pp. 1273–
1283. [Online]. Available: http://www.aclweb.org/anthology/D14-1134

2) C. Quirk, R. Mooney, and M. Galley,
―Language to code: Learning semantic parsers for if-this-then-that recipes,‖ in Proceedings of the
53rd Annual Meeting of the Association for Computational Linguistics
(ACL-15), Beijing, China, July 2015, pp. 878– 888. [Online]. Available: http://www.cs.utexas.edu/users/ai-lab/pubview.php?PubID=127514

3) X. Leroy, S. Blazy, D. Kästner, B. Schommer, M. Pister, and C.
Ferdinand, ―CompCert-A Formally Verified Optimizing Compiler,‖ in
ERTS 2016: Embedded Real Time Software and Systems, 8th European Congress, 2016.

4)Sable NileshPopat*, Y. P. Singh," Efficient Research on the Relationship Standard Mining Calculations in Data Mining" in *Journal of Advances in Science and Technology | Science & Technology,* Vol. 14, Issue No. 2, September-2017, ISSN 2230-9659.

5)Sable NileshPopat*, Y. P. Singh," Analysis and Study on the Classifier Based Data Mining Methods" in *Journal of Advances in Science and Technology | Science & Technology,* Vol. 14, Issue No. 2, September-2017, ISSN 2230-9659

6) S.C. Johnson, ―Yacc: Yet Another Compiler-Compiler,‖ tech. rep.,
Bell Laboratories. 1974

7)A. V. Aho, R. Sethi, and J. D. Ullman, Compilers, principles, techniques,
and tools, Addison Wesley, 1985

8)S. Buchwald, "Optgen: A generator for local optimizations," in
International Conference on Compiler Construction, 2015.

Authors Profile :

Dr Nilesh Sable pursued PhD in Computer Science & Engineering 2018 Kalinga University.
His current research interests are Data Mining, Network Security, and Cloud Computing. Having a work experience of 5 years in
Teaching.

Mr. Akshay Rajputpursed Diploma in Computer Engineering from MSBTE, Maharashtra in 2015 and He is currently pursuing Bachelor's Degree in Computer Engineering from JSPM's Imperial College of Engineering and Research, Wagholi . His current research interests are Computer Security based education.

Miss. Sonali Wadikar pursed Diploma in Computer Engineering from Autonomous,
Maharashtra in 2015 and she is currently pursuing Bachelor's Degree in Computer Engineering from JSPM's Imperial College of Engineering and Research, Wagholi.
His current research interests are Programming based education.

Miss. Madhuri Bobade pursed Diploma in Computer Engineering from MSBTE,
Maharashtra in 2015 and she is currently pursuing Bachelor's Degree in Computer Engineering from JSPM's Imperial College of Engineering and Research, Wagholi.
His current research interests are Programming based education.

Mr.KambleShubhampursed Diploma in Computer Engineering from MSBTE, Maharashtra in 2015 and He is currently pursuing Bachelor's Degree in Computer Engineering from JSPM's Imperial College of Engineering and Research, Wagholi
His current research interests are Computer Security based education.