PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 35, NO. 2, PP. 147–152 (1991)

SOFTWARE GENERATION BASED ON ATTRIBUTE GRAMMARS

T. GYIMÓTHY

Research Group on the Theory of Automata
Hungarian Academy of Sciences
Szeged, Hungary

Received: Sept. 5, 1990.

Abstract

In this paper a short overview is given of a software generator tool based on attribute grammars and the experiences are summarized with the use of this system for generating different types of software.

Keywords: software generation, attribute grammar.

1 Introduction

Software generation becomes more and more important in the process of software development. In many areas of software development, these generators are considered as convenient tools for producing high quality and easy-to-maintain products. In the case of large-size projects, a high-level specification of a system is indispensable. Attribute grammars (KNUTH, 1968) seem to be prospective candidates as good theoretical and practical models for such specifications.

We have developed a system called PROF-LP which can generate software from an attribute grammar specification. PROF-LP is a general purpose generator, it can generate language processors of different types. However, our experiences with the system suggest that in a number of cases a general purpose system is not suitable. Special purpose software generators are much more convenient tools for the generation of application programs. Hence, PROF-LP can be considered as a generator for different types of software generators.

In Section 2 a short overview of PROF-LP is presented, then a description of a large CASE system called SYS/3 is given in Section 3. SYS/3 contains several software generators which are generated by PROF-LP. In Section 4 the main features of THALES, a program package for plane geometry constructions are described. This system has a natural language interface for processing instructions formulated in English. The major part of the system is generated from an attribute grammar specification. An outline of future plans is presented in Section 5.

2 PROF-LP

PROF-LP (PROFessional Language Processor) is a software generator based on attribute grammars. The structure of the system can be seen in *Fig. 1*. There are two metalanguages in the system. The lexical metalanguage is used to describe the structure of basic symbols. Definitions for character sets and token class definitions, string transformations and screening rules are included. Token classes are given with regular expressions.

In the syntactic-semantic description, the user can define an attribute grammar. The specification of an attribute grammar is composed of the following parts:

— Pascal or C declarations,
— declaration of synthesized and inherited attributes with their Pascal or C types,
— declaration of nonterminals with their attributes,
— the productions with their associated semantic rules and code templates.

The evaluation order is determined by attribute dependencies, whereas the user has an explicit control over the code generation treewalk.

PROF-LP can process LL(1) grammars with L-attribute (BOCHMANN, 1976) or LALR(1) grammars with Ordered Attribute Grammars (KASTENS, 1980). The system has been extended with a backtrack parser generator for the generation language processors for natural language processing and syntactic pattern recognition. The generated backtrack parsers use the LL(1) parsing table, thus a lot of useless backtracks can be eliminated (GYIMÓTHY, 1987). A complete description of PROF-LP is given in (PROF-LP, 1986).

3 SYS/3

SYS/3 (SYStematic SYStem Development SYStem) is a CASE system for assisting in developing software products according to a pre-stated standard. SYS/3 assures the proper quality of a software during its entire life-cycle. *Fig. 2* shows the structure of the system.

From a functional point of view, SYS/3 has three subsystems: project management, control and development.

The management subsystem contains the functions used for organizing and managing the project. It handles the following objects: the users,

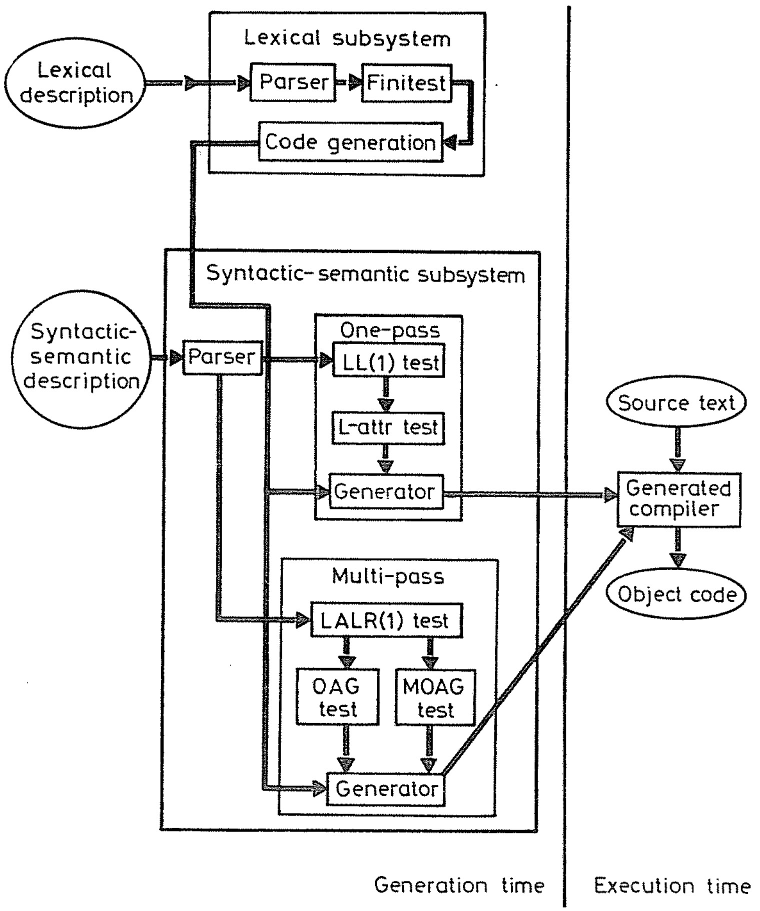

Fig. 1. The structure of PROF-LP

the workstations, and the tasks. The control subsystem contains a detailed diary for storing information about the system development process. This information is accessible using a comfortable menu system.

The development subsystem contains a configuration management program, document handling programs and some special purpose programs.

Major parts of SYS/3 are generated from high-level descriptions.

The user interface of the system is generated with the UIG general user interface generator developed by our research group. In the metalanguage of UIG, the user has to define only the structure of his/her fields, windows and screens. From this specification, UIG is capable of generating simple screen handler routines with formed input and output fields as

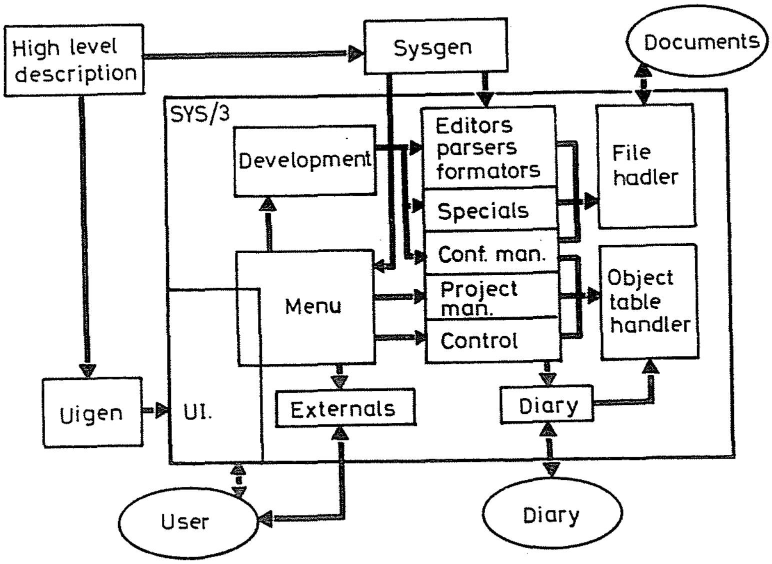

Fig. 2. The structure of SYS/3

well as help systems, structured file listers, menu windows and usual text editors.

SYS/3 contains built in syntax-directed editors and parsers for each type of documents (e. g. System Specification, Software Plan, Requirements Analysis, Test Specification, Preliminary Design Document, Detailed Design).

The editors and parsers are generated with the SYSGEN syntax-directed editor and parser generator system. SYSGEN can be considered as a modified version of PROF-LP.

As we mentioned above, the development subsystem of SYS/3 contains special purpose programs. One of them is the T-GEN test case generator system. The T-GEN system gives assistance to the functional testing of programs in C. Turbo Pascal and dBase/Clipper. T-GEN can generate LOTUS macros for testing LOTUS applications as well. The system can be considered as a meaningful extension of the category-partition method presented in (OSTRAND and BALCER, 1988). The standpoints of testing can be specified in a high-level description. From this specification, the system generates executable test cases.

SYS/3 contains a subsystem called PDL (Program Design Language), to assist in the preparation of Detailed Design document. PDL is based on the program design languages presented in (Software Engineering, 1986), but the original language has been augmented with numerous new features. Formal elements of PDL define the most important data types (SCALAR, ARRAY, MATRIX, LIST, CHAIN, STACK, TREE, ROW, SET, RECORD, FILE) and program structures (IF, CASE, DO, FOR, REPEAT, ESCAPE, EXIT, RETURN). From a PDL description, a Turbo Pascal source is generated.

The generators (UIG, SYSGEN, T-GEN, PDL) mentioned in this Section are generated with PROF-LP. A more detailed description of SYS/3 can be found in (TOCZKI et al, 1990).

4 THALES

THALES, a program package for plane geometry constructions, is equipped with a natural language interface developed on the basis of an attribute grammar specification. Constructions are carried out as series of instructions formulated in English.

The natural language interface of THALES consists of the following functional units:

— lexical-morphological processor,
— syntactic parser,
— semantic analyzer.

The lexical-morphological processor is based on a lexicon with a fixed number of items covering the sublanguage of plane geometry. The syntactic module represents a top-down, basically LL(1), algorithm augmented hand-written look-ahead for controlling the occasional LL(1) conflicts.

The range of syntactic and semantic coverage can be illustrated by the examples below.

(1) Draw two circles *c1, c2.*
(2) Across the points of intersection of the circles, draw a line *e*.
(3) On the line *e*, pick a point *P* that is outside the circles.
(4) Label the center of the circles by *A* and *B*.
(5) Draw a circle with center *P* and radius *AB*.
(6) Compute the area of triangle *APB*.

During the implementation of the semantic analyzer we tried to make a clear-cut distinction between the static and dynamic semantics of the sentences. Our experiences showed that the most important static semantic features (e. g. identification, type compatibility) of THALES-like natural language interfaces can be handled on the basis of a high-level description.

The identification problem, i. e. finding the appropriate defining occurrence, is usually not too difficult in traditional programming languages. Because of the ambiguity inherent in natural languages, a much more sophisticated algorithm is required in THALES-like natural language interfaces. A detailed description of THALES can be found in (FABRICZ et al, 1990).

5 Future Plans

As described above, the current version of PROF-LP can generate syntax-directed editors, interpreters, front-ends and compilers from an attribute grammar specification. Our aim has been to develop a new version of the system which can produce integrated development environment from a given specification of a language. This environment will contain a user-friendly menu-driven subsystem for the integration of the different generated tools of the given language.

References

BOCHMANN, G. V. (1976): Semantic Evaluation from Left to Right. *Comm. ACM*, Vol. 19, pp. 55–62.

FABRICZ, K. et al (1990): THALES: A Natural Language Interface for Plane Geometry Constructions. *Proceedings of COLING'90.* Helsinki, 1990. Vol. 1. pp. 44–46.

GYIMÓTHY, T. – TOCZKI, J. (1987): Syntactic Pattern Recognition in the HLP/PAS System. *Acta Cybernetica*, Vol. 8, No. 1. pp. 79–88.

KASTENS, U. (1980): Ordered Attribute Grammars. *Acta Informatica*, Vol. 13. pp. 229–256.

KNUTH, D. E. (1968): Semantic of Context-free Languages. *Math. Systems Theory*, Vol. 2. No. 2. pp. 127–145.

OSTRAND, T. J. – BALCER, M. J. (1988): The Category-Partition Method for Specifying and Generating Functional Tests. *Comm. ACM*, Vol. 31. No. 6. pp. 676–686.

PROF-LP Users' Guide, (1986), Vol. 1–3., Research Group on the Theory of Automata, Szeged.

Software Engineering, (1986), General Electric Company, Corporate Information Systems, Bridgeport, USA.

TOCZKI, J. et al. (1990): SYS/3 - a Software Development Tool. *Proceedings of Compiler Compilers.* Schwerin 1990. LNCS. No. 477. pp. 193–208.

Address:

Tibor GYIMÓTHY
Research Group on the Theory of Automata
Hungarian Academy of Sciences
Szeged, Aradi Vértanúk tere 1.
H-6720, Hungary

INFORMATION FOR AUTHORS

Submitting a Manuscript for Publication. Submission of a paper to this journal implies that it represents original work previously not published elsewhere, and that it is not being considered elsewhere for publication. If accepted for publication, the copyright is passed to the publisher: the paper must not be published elsewhere in the same form, in any language, without written consent of the executive editor. The first author will receive 50 free reprints.

Manuscripts should be submitted in English in two copies to the editors' office (see inner front cover). Good office duplicated copies are acceptable.

Periodica Polytechnica, Series Electrical Engineering is typeset using the TEX program with the AMSTEX macro package. Therefore, authors are encouraged to submit their contribution in this form too (e.g. on a 51/4" floppy diskette, readable on an IBM compatible PC), or at least the text of the article in a simple ASCII file. By this solution most of the typesetting errors can be avoided, and publishing time can be reduced. Papers prepared in TEX are to be prepared using the AMSTEX macro package, without setting the style (letter types, headers etc.), which will be adjusted by the Secretariat.

Compilation and Typing of Manuscripts. Contributions should be typed in double spacing on good quality ISO A4 paper with good quality black ribbon in the standard form: 25 lines of 50 characters on each page. One page may contain not more than 10 corrections (paints dò not count).

The maximum length of the manuscript is 20 pages including illustrations and tables. Use one side of the sheet only. Paragraphs are to be indented by 5 spaces and not to be preceded by a blank line.

A correctly compiled manuscript should contain the following items:

(1) Title page giving: the title (short, with subheading if necessary); the name(s) of the author(s); affiliation (institution, firm etc.) of the author(s), in English; grants, scholarships etc. during the research (in a footnote); an informative abstract of not more than 200 words with 3-5 keywords;
(2) Textual part of the contribution with a list of references;
(3) A separate sheet giving full name(s), postal address(es) of the author(s) to be used in correspondence and phone, telex and telefax numbers (if applicable);
(4) A separate sheet listing all figure captions and table headers;
(5) Illustrations and tables (please put your name on each sheet).

Abstract. A summary or abstract of about 100-200 words should be provided on the title page. This should be readable without reference to the article, and should indicate the scope of the contribution, including the main conclusions and essential original content.

Keywords are to be given for the purpose of data bases and abstracting; avoid too general keywords which provide no help in literature searching.

General rules for the text. Chapters are to be numbered with Arabic numbers in decimal hierarchy. However, do not refer to chapters by number, because in print the hierarchy of headings will be indicated by typesetting.

Wherever possible, mathematical equations should be typewritten, with subscripts and superscripts clearly shown. Metric (SI) units are to be used, other units may be given in parentheses. Equations must be numbered on the right side in parentheses. Handwritten or rare mathematical, Greek and other symbols should be identified or even explained if necessary in the margin. Letters denoting quantities are to be distinguished by different setting both in the formulae and in the text. Remember the rule that scalar quantities are to be denoted by italics (underline by hand in your manuscript), vectors by lower case bold type letters (underline doubled), and matrices by bold capitals (underline doubled). Dimensions (like cm, Ohm, V etc.) and standard function names (sin, ln, P etc.) are to be typeset in Roman typefaces (not in italics). A few important words may be distinguished by italic setting (underline).

Illustrations and Tables. Graphs, charts and other line illustrations should be drawn neatly in Indian ink. Computer printouts can only be used if they are of excellent quality (e.g. they are prepared using a laser printer). Figures should be submitted in an adequately large size for drawing. Rare or ambiguous letters and notations are to be explained beside the illustrations. All figures should be numbered with consecutive Arabic numbers, have descriptive captions, and be referred to in the text. Captions should be self-explanatory, not merely labels. Figures must not contain lengthy texts; use captions instead.

Number tables consecutively with Arabic numbers and give each a clear descriptive caption at the top. If possible avoid vertical rules in tables. Tables should be preferably submitted in camera-ready form.

The earliest acceptable position of figures and tables is to be indicated in the left margin.

References. In the text references should be made by the author's surname and year of publication in parentheses, e.g. (EARL and TÖRÖK, 1978) or ...was given by KISS (1976a). Where more than one publication by an author in one year is referred to, the year should be followed by a suffix letter (a, b etc.), the same suffix being given in the reference list. For the style of the reference list, which is to be given in alphabetical order, see the examples below for journal articles, conference papers and books.

EARL, J. - TÖRÖK, I. (1978): Partial Discharge Measurement in Cable Tests. *Periodica Polytechnica Ser. Electrical Engineering*, Vol. 22, No. 4, pp. 133–138.

KISS, S. (1976a): Roundoff Errors in FFT. *Proc. 5th IEEE Symposium on Signal Processing*, Boston (MA), May 3–5, 1976. New York, NY, IEEE Press, CH0092-2875/76, pp. 3.5–3.9.

KISS, S. (1976b): Ellenállások (Resistances). Budapest, Tankönyvkiadó. pp. 533–535. (in Hungarian)

More detailed guidelines for authors are available from the editors' office (see inner front cover).