Code Transformation by Direct Transformation of ASTs

Type	Conference Paper
Author	M. Rizun
Author	J.-C. Bach
Author	S. Ducasse
URL	http://doi.acm.org.ezproxy.library.yorku.ca/10.1145/2811237.2811297
Series	IWST '15
Place	New York, NY, USA
Publisher	ACM
Pages	7:1–7:7
ISBN	978-1-4503-3857-8
Date	2015
DOI	10.1145/2811237.2811297
Proceedings Title	Proceedings of the International Workshop on Smalltalk Technologies
Date Added	2018-08-27, 8:04:57 PM
Modified	2018-08-27, 8:04:57 PM

Tags:

AST, code transformation, refactoring, rewriting

Notes:

Fine-grained and Accurate Source Code Differencing

Type	Conference Paper
Author	Jean-Rémy Falleri
Author	Floréal Morandat
Author	Xavier Blanc
Author	Matias Martinez
Author	Martin Monperrus
URL	http://doi.acm.org.ezproxy.library.yorku.ca/10.1145/2642937.2642982
Series	ASE '14
Place	New York, NY, USA

Publisher	ACM
Pages	313–324
ISBN	978-1-4503-3013-8
Date	2014
DOI	10.1145/2642937.2642982
Proceedings Title	Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering
Date Added	2018-08-27, 8:05:42 PM
Modified	2018-08-27, 8:05:42 PM

Tags:

ast, program comprehension, software evolution, tree differencing

Notes:

Implementing Real-time Collaboration in TouchDevelop Using AST Merges

Type	Conference Paper
Author	Jonathan Protzenko
Author	Sebastian Burckhardt
Author	Micha\l Moskal
Author	Jedidiah McClurg
URL	http://doi.acm.org.ezproxy.library.yorku.ca/10.1145/2846661.2846672
Series	MobileDeLi 2015
Place	New York, NY, USA
Publisher	ACM
Pages	25–27
ISBN	978-1-4503-3906-3
Date	2015
DOI	10.1145/2846661.2846672
Proceedings Title	Proceedings of the 3rd International Workshop on Mobile Development Lifecycle
Date Added	2018-08-27, 8:06:32 PM
Modified	2018-08-27, 8:06:32 PM

Tags:

collaborative editing, diff, Merge

Notes:

Object-Oriented Reverse Engineering Coarse-grained, Fine-grained, and Evolutionary Software Visualization

Type	Book
Author	Michele Lanza
Date	2003
Date Added	2018-08-27, 8:04:13 PM
Modified	2018-08-27, 8:04:13 PM

Notes:

RASCAL: A Domain Specific Language for Source Code Analysis and Manipulation

Type	Conference Paper
Author	Paul Klint
Author	Tijs van der Storm
Author	Jurgen Vinju
URL	http://dx.doi.org.ezproxy.library.yorku.ca/10.1109/SCAM.2009.28
Series	SCAM '09
Place	Washington, DC, USA
Publisher	IEEE Computer Society
Pages	168–177
ISBN	978-0-7695-3793-1
Date	2009
DOI	10.1109/SCAM.2009.28
Proceedings Title	Proceedings of the 2009 Ninth IEEE International Working Conference on Source Code Analysis and Manipulation
Date Added	2018-08-27, 7:01:34 PM

Modified 2018-08-27, 7:01:34 PM

Tags:

meta-programming, source code analysis, source code manipulation, transformation

Notes:

- language which provides both source-code analysis and manipulation (SCAM).
- emphasizes ‘expressiveness, safety and usability’ (Klint, van der Storm, Vinju 2)
- available as an eclipse plugin or a CLI tool.

- aims to be well-suited for large projects and refactoring

interaction with the language is done using 'concrete syntax' such that its usage may be akin to the simplicity of using other CLI tools like GREP and AWK.

- "...source code analysis and transformation is a form of programming." (2)

TWEAST: A Simple and Effective Technique to Implement Concrete-syntax AST Rewriting Using Partial Parsing

Type	Conference Paper
Author	Akim Demaille
Author	Roland Levillain
Author	Benoît Sigoure
URL	http://doi.acm.org.ezproxy.library.yorku.ca/10.1145/1529282.1529710
Series	SAC '09
Place	New York, NY, USA
Publisher	ACM
Pages	1924–1929
ISBN	978-1-60558-166-8
Date	2009
DOI	10.1145/1529282.1529710
Proceedings Title	Proceedings of the 2009 ACM Symposium on Applied Computing
Date Added	2018-08-27, 7:23:26 PM
Modified	2018-08-27, 7:23:26 PM

Tags:

C++, compiler design, concrete syntax, parsing, program transformation, rewrite rules

Notes:

noteworthy for our interest, as it is used for creating and manipulating abstract syntax trees of other languages. it implements a concrete syntax, whereas we are interested in manipulating the AST in its graphed form.

perhaps of note for Graham: "A compiler for a simple language (Tiger) written in C++ serves as an example, featuring transformations in concrete syntax..." (1924)

like RASCAL, TWEAST also uses a 'concrete syntax'

observes that changing the ASTs directly is presents a challenge due to "...the reqrie rules are expressed in abstract syntax -- the syntax of the transforming language, not the transformed one." (1924)

*supports partial parsing of an AST. sub-ASTs represented in an object as both abstract format and as a string

limitation (as of the time of the article, follow up to see if its been rectified) is that TWEAST can use concrete syntax to "...produce ASTs, not to match them." (1928)