SLEBoK

@
SLE
2017

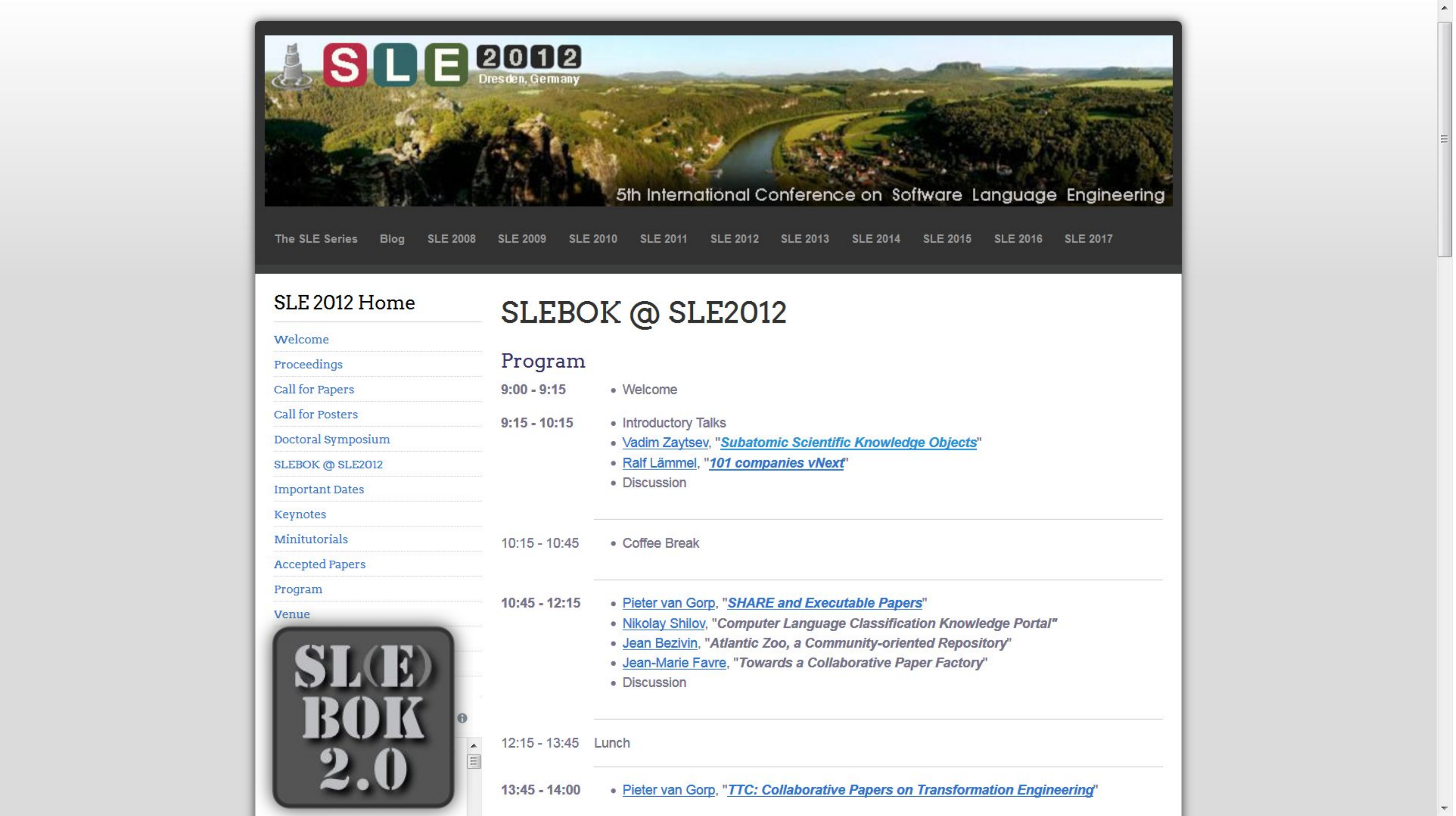
Body of Knowledge

- The complete set of concepts, terms and activities that make up a professional domain, as defined by the relevant society/ association
- Structured knowledge used by members of a discipline to guide their practice or work
- The prescribed aggregation of knowledge in a particular area an individual is expected to have mastered to be considered or certified as a practitioner
- A stepping stone to unifying community
- The systematic collection of activities and outcomes in terms of their values, constructs, models, principles and instantiations, which
    - arises from continuous discovery and validation work by members of the profession and
    - enables self-reflective growth and reproduction of the profession
- A set of accepted and agreed upon standards and nomenclatures of a field/profession
- A set of knowledge within a profession or subject area which is generally agreed as both essential and generally known

SWEBOK®

- https://www.computer.org/web/swebok/index
- Committee + 150 reviewers in 33 countries
- 15 knowledge areas:
  - Software requirements
  - Software design
  - Software construction
  - Software testing
  - Software maintenance
  - Software configuration management
  - Software engineering management
  - Software engineering process
  - ...

Roots of SLEBoK (25 Sep 2012)



SLEBOK @ SLE2012

Program

9:00 - 9:15
- Welcome

9:15 - 10:15
- Introductory Talks
- Vadim Zaytsev, "*Subatomic Scientific Knowledge Objects*"
- Ralf Lämmel, "*101 companies vNext*"
- Discussion

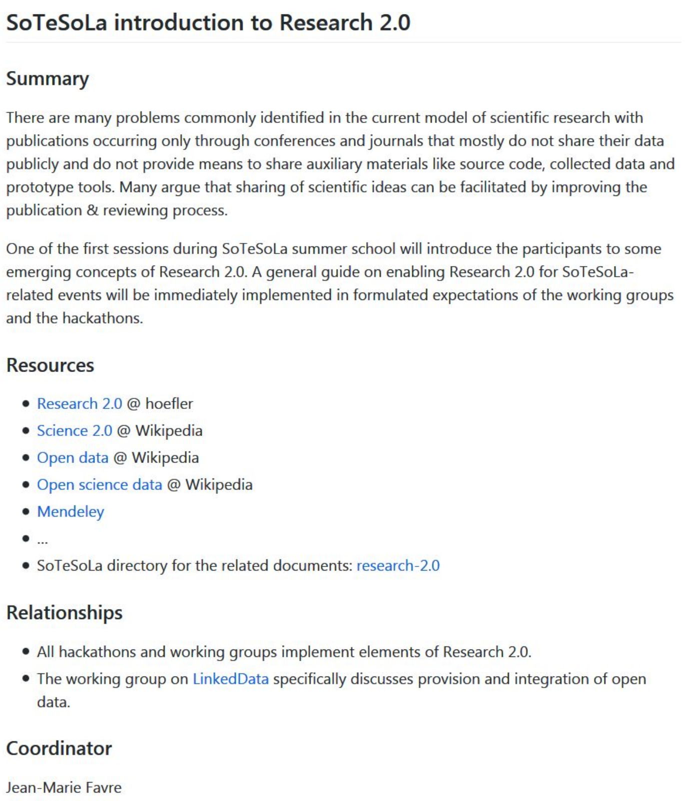
10:15 - 10:45
- Coffee Break

10:45 - 12:15
- Pieter van Gorp, "*SHARE and Executable Papers*"
- Nikolay Shilov, "*Computer Language Classification Knowledge Portal*"
- Jean Bezivin, "*Atlantic Zoo, a Community-oriented Repository*"
- Jean-Marie Favre, "*Towards a Collaborative Paper Factory*"
- Discussion

12:15 - 13:45 Lunch

13:45 - 14:00
- Pieter van Gorp, "*TTC: Collaborative Papers on Transformation Engineering*"

Roots of SLEBoK (Aug 2012)



SoTeSoLa introduction to Research 2.0

Summary

There are many problems commonly identified in the current model of scientific research with publications occurring only through conferences and journals that mostly do not share their data publicly and do not provide means to share auxiliary materials like source code, collected data and prototype tools. Many argue that sharing of scientific ideas can be facilitated by improving the publication & reviewing process.

One of the first sessions during SoTeSoLa summer school will introduce the participants to some emerging concepts of Research 2.0. A general guide on enabling Research 2.0 for SoTeSoLa-related events will be immediately implemented in formulated expectations of the working groups and the hackathons.

Resources

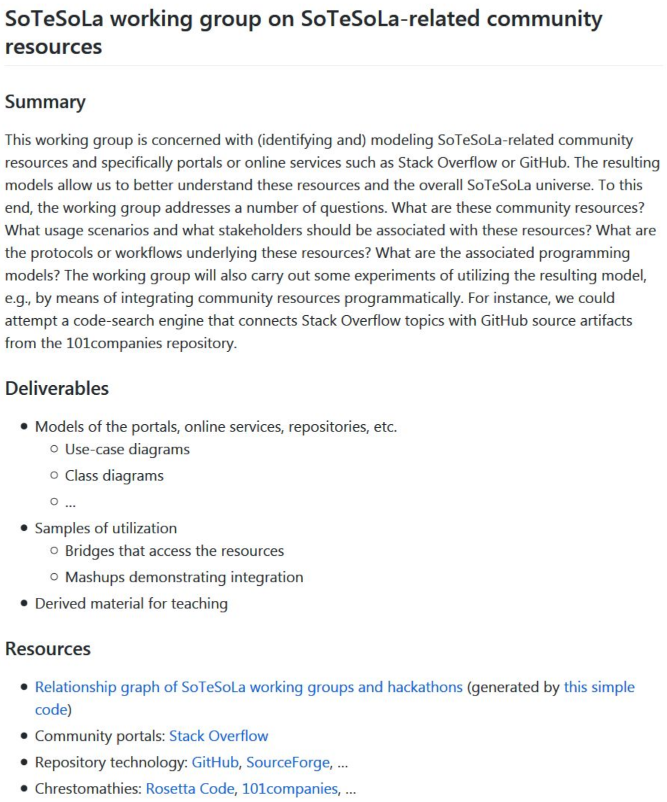
- Research 2.0 @ hoefler
- Science 2.0 @ Wikipedia
- Open data @ Wikipedia
- Open science data @ Wikipedia
- Mendeley
- ...
- SoTeSoLa directory for the related documents: research-2.0

Relationships

- All hackathons and working groups implement elements of Research 2.0.
- The working group on LinkedData specifically discusses provision and integration of open data.

Coordinator

Jean-Marie Favre



SoTeSoLa working group on SoTeSoLa-related community resources

Summary

This working group is concerned with (identifying and) modeling SoTeSoLa-related community resources and specifically portals or online services such as Stack Overflow or GitHub. The resulting models allow us to better understand these resources and the overall SoTeSoLa universe. To this end, the working group addresses a number of questions. What are these community resources? What usage scenarios and what stakeholders should be associated with these resources? What are the protocols or workflows underlying these resources? What are the associated programming models? The working group will also carry out some experiments of utilizing the resulting model, e.g., by means of integrating community resources programmatically. For instance, we could attempt a code-search engine that connects Stack Overflow topics with GitHub source artifacts from the 101companies repository.

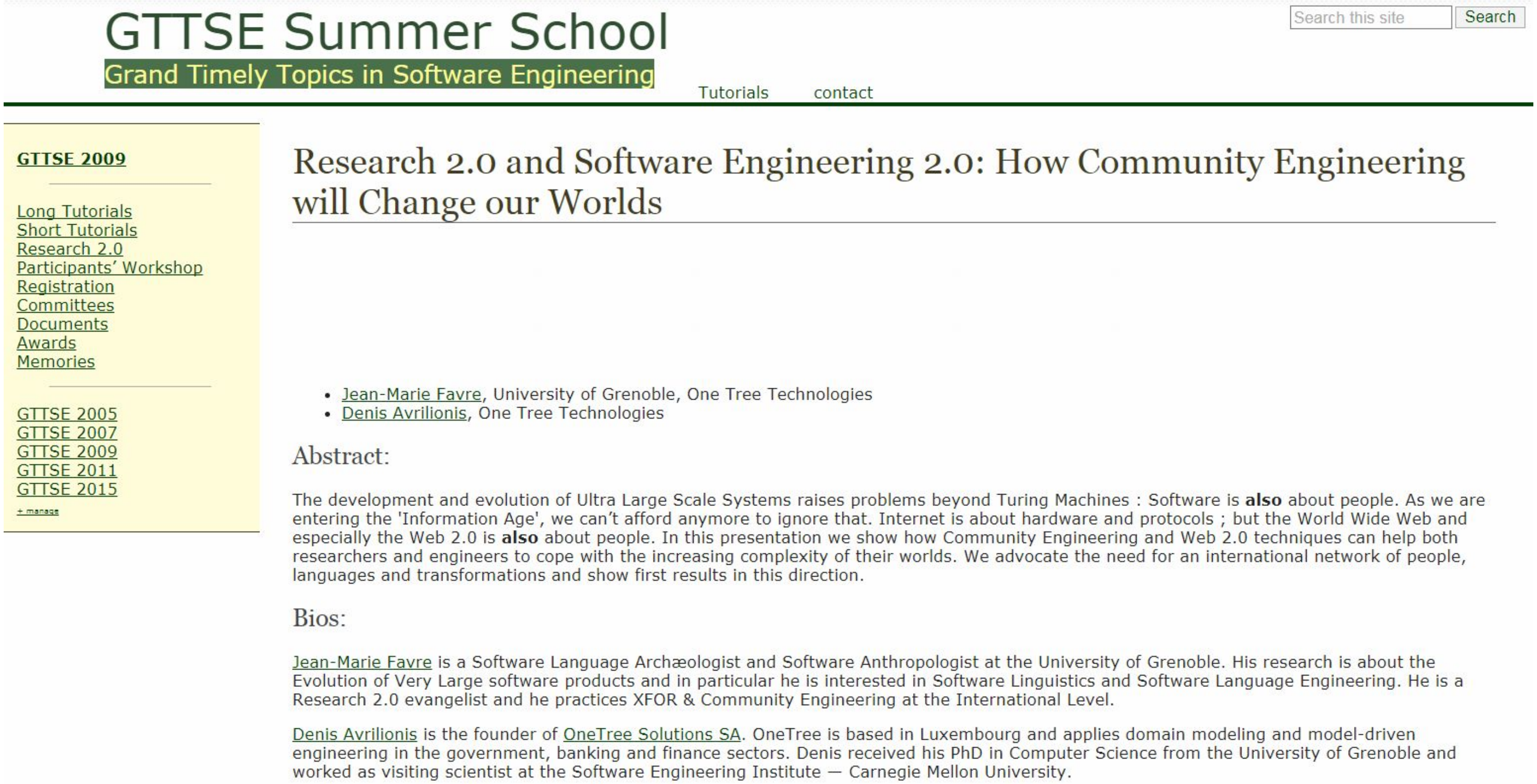
Deliverables

- Models of the portals, online services, repositories, etc.
  - Use-case diagrams
  - Class diagrams
  - ...
- Samples of utilization
  - Bridges that access the resources
  - Mashups demonstrating integration
- Derived material for teaching

Resources

- Relationship graph of SoTeSoLa working groups and hackathons (generated by this simple code)
- Community portals: Stack Overflow
- Repository technology: GitHub, SourceForge, ...
- Chrestomathies: Rosetta Code, 101companies, ...

Roots of SLEBoK (July 2009)



GTTSE Summer School

Grand Timely Topics in Software Engineering

Research 2.0 and Software Engineering 2.0: How Community Engineering will Change our Worlds

- Jean-Marie Favre, University of Grenoble, One Tree Technologies
- Denis Avrilionis, One Tree Technologies

Abstract:

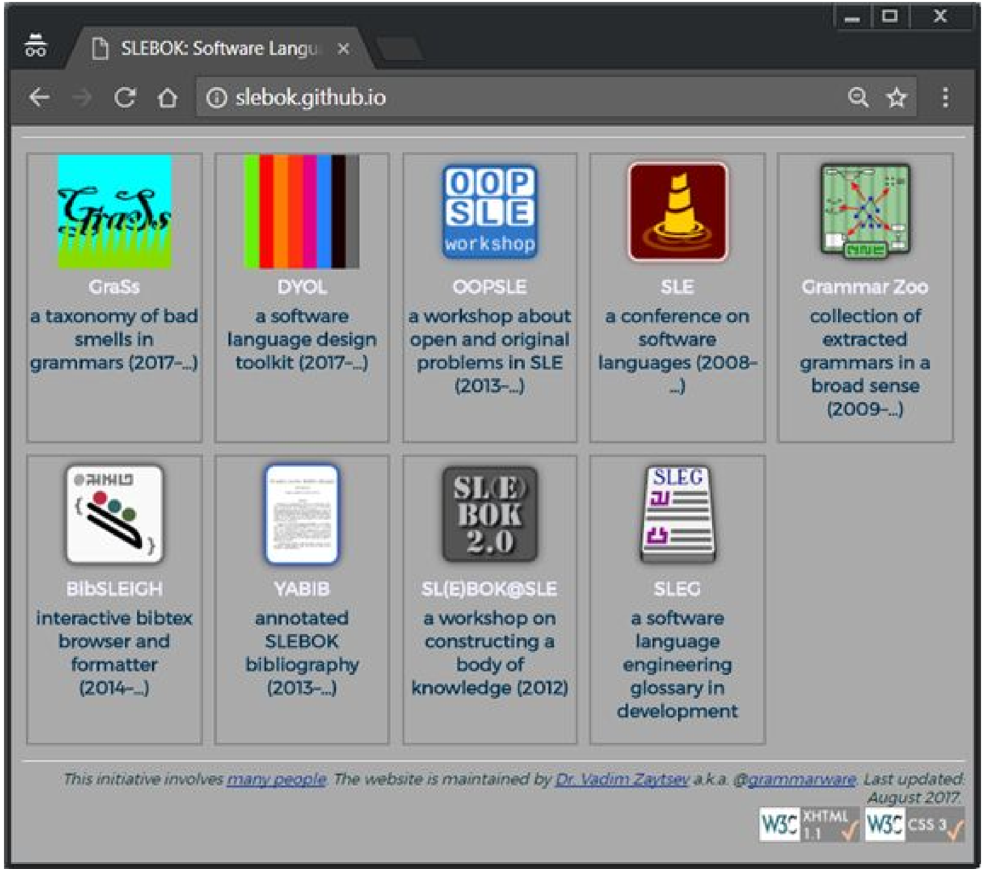
The development and evolution of Ultra Large Scale Systems raises problems beyond Turing Machines : Software is **also** about people. As we are entering the 'Information Age', we can't afford anymore to ignore that. Internet is about hardware and protocols ; but the World Wide Web and especially the Web 2.0 is **also** about people. In this presentation we show how Community Engineering and Web 2.0 techniques can help both researchers and engineers to cope with the increasing complexity of their worlds. We advocate the need for an international network of people, languages and transformations and show first results in this direction.

Bios:

Jean-Marie Favre is a Software Language Archæologist and Software Anthropologist at the University of Grenoble. His research is about the Evolution of Very Large software products and in particular he is interested in Software Linguistics and Software Language Engineering. He is a Research 2.0 evangelist and he practices XFOR & Community Engineering at the International Level.

Denis Avrilionis is the founder of OneTree Solutions SA. OneTree is based in Luxembourg and applies domain modeling and model-driven engineering in the government, banking and finance sectors. Denis received his PhD in Computer Science from the University of Grenoble and worked as visiting scientist at the Software Engineering Institute — Carnegie Mellon University.

Roots of SLEBoK (since 14 Feb 2013)



GraSs
a taxonomy of bad smells in grammars (2017–...)

DYOL
a software language design toolkit (2017–...)

OOPSLE
a workshop about open and original problems in SLE (2013–...)

SLE
a conference on software languages (2008–...)

Grammar Zoo
collection of extracted grammars in a broad sense (2009–...)

BibSLEIGH
interactive bibtex browser and formatter (2014–...)

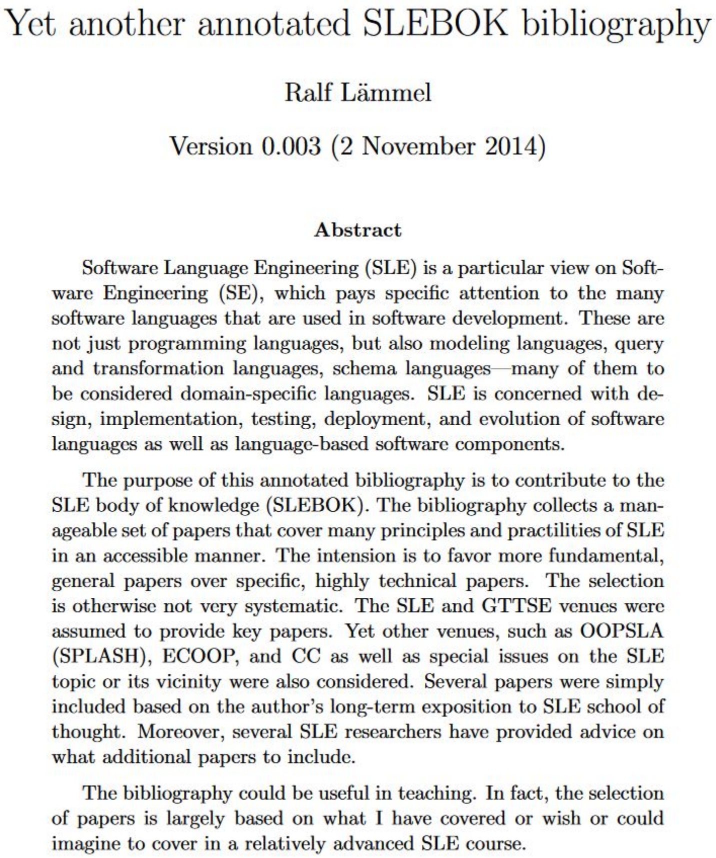
YABIB
annotated SLEBOK bibliography (2013–...)

SL(E)BOK@SLE
a workshop on constructing a body of knowledge (2012)

SLEG
a software language engineering glossary in development

This initiative involves many people. The website is maintained by Dr. Vadim Zaytsev a.k.a. @grammarware. Last updated: August 2017.

Roots of SLEBoK (16 Oct 2013)



Yet another annotated SLEBOK bibliography

Ralf Lämmel

Version 0.003 (2 November 2014)

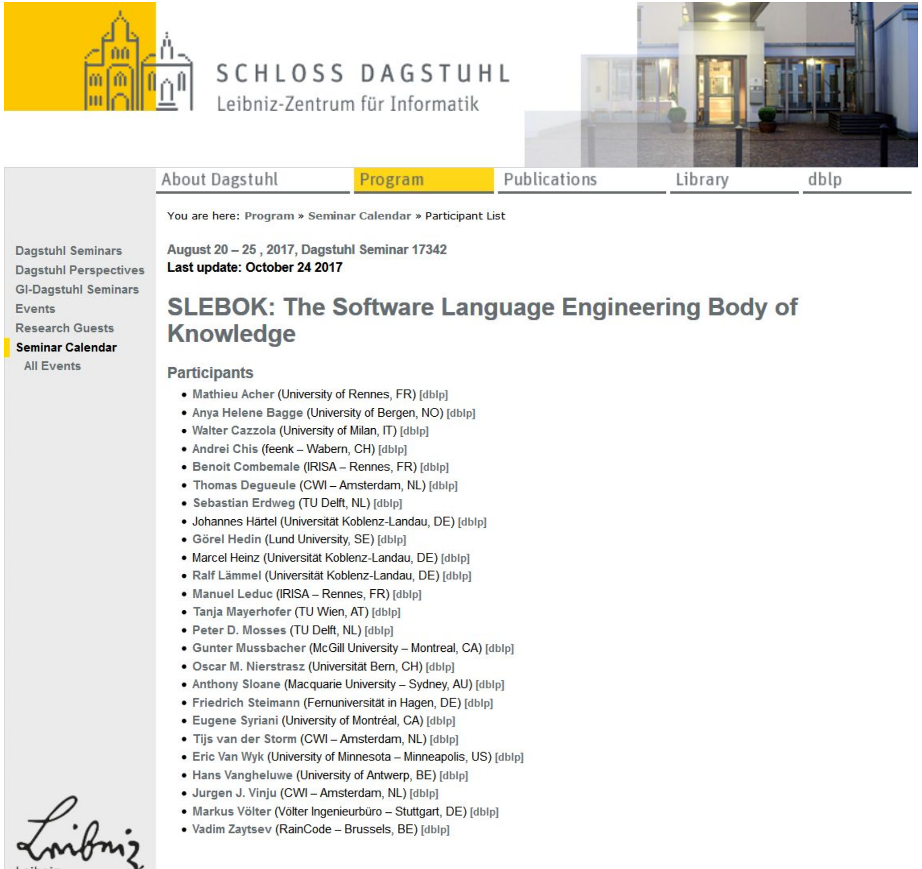
Abstract

Software Language Engineering (SLE) is a particular view on Software Engineering (SE), which pays specific attention to the many software languages that are used in software development. These are not just programming languages, but also modeling languages, query and transformation languages, schema languages—many of them to be considered domain-specific languages. SLE is concerned with design, implementation, testing, deployment, and evolution of software languages as well as language-based software components.

The purpose of this annotated bibliography is to contribute to the SLE body of knowledge (SLEBOK). The bibliography collects a manageable set of papers that cover many principles and practilities of SLE in an accessible manner. The intension is to favor more fundamental, general papers over specific, highly technical papers. The selection is otherwise not very systematic. The SLE and GTTSE venues were assumed to provide key papers. Yet other venues, such as OOPSLA (SPLASH), ECOOP, and CC as well as special issues on the SLE topic or its vicinity were also considered. Several papers were simply included based on the author's long-term exposition to SLE school of thought. Moreover, several SLE researchers have provided advice on what additional papers to include.

The bibliography could be useful in teaching. In fact, the selection of papers is largely based on what I have covered or wish or could imagine to cover in a relatively advanced SLE course.

Roots of SLEBoK (Aug 2017)



You are here: Program » Seminar Calendar » Participant List

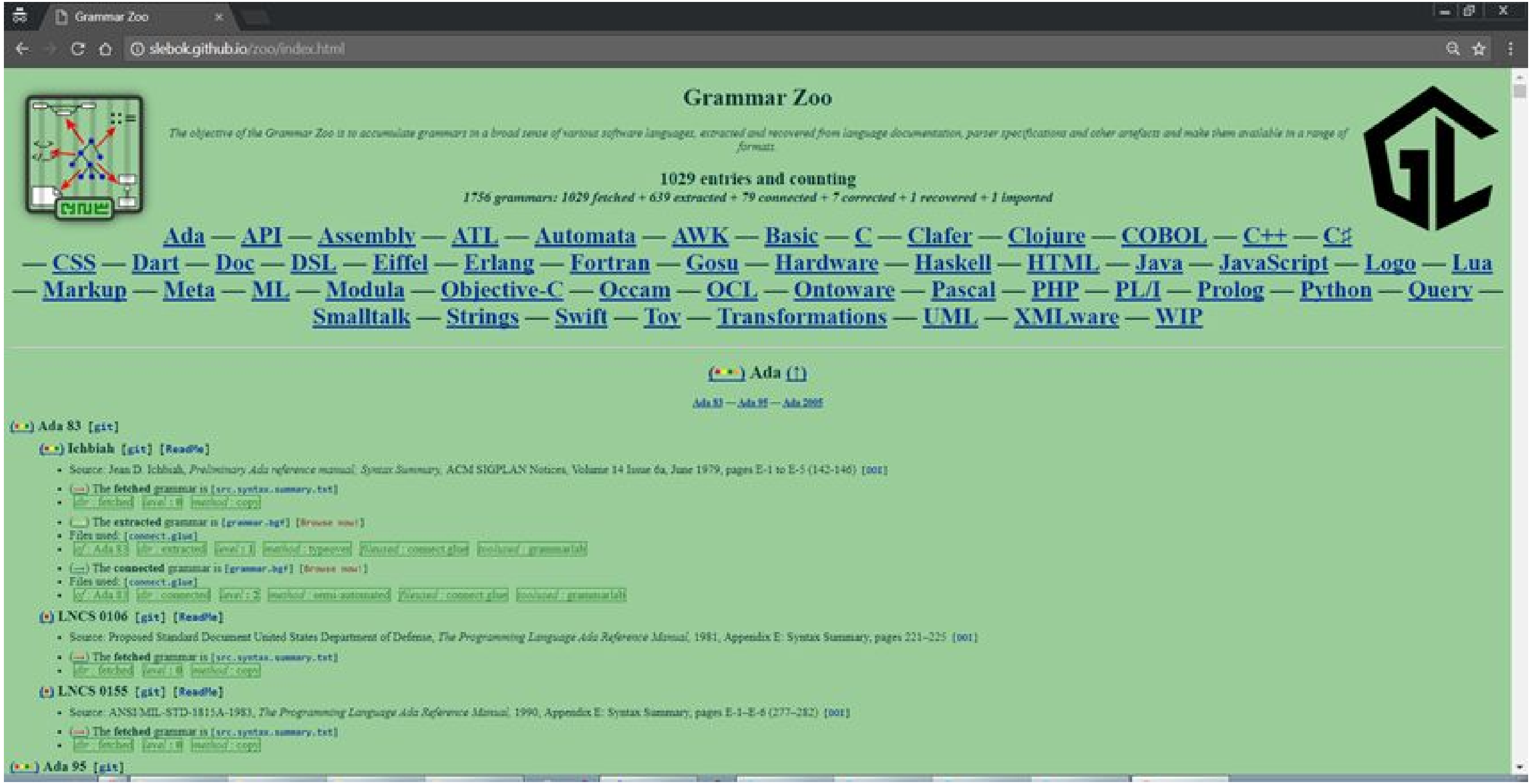
August 20 – 25 , 2017, Dagstuhl Seminar 17342
Last update: October 24 2017

SLEBOK: The Software Language Engineering Body of Knowledge

Participants

- Mathieu Acher (University of Rennes, FR) [dblp]
- Anya Helene Bagge (University of Bergen, NO) [dblp]
- Walter Cazzola (University of Milan, IT) [dblp]
- Andrei Chis (feenk – Wabern, CH) [dblp]
- Benoit Combemale (IRISA – Rennes, FR) [dblp]
- Thomas Degueule (CWI – Amsterdam, NL) [dblp]
- Sebastian Erdweg (TU Delft, NL) [dblp]
- Johannes Härtel (Universität Koblenz-Landau, DE) [dblp]
- Görel Hedin (Lund University, SE) [dblp]
- Marcel Heinz (Universität Koblenz-Landau, DE) [dblp]
- Ralf Lämmel (Universität Koblenz-Landau, DE) [dblp]
- Manuel Leduc (IRISA – Rennes, FR) [dblp]
- Tanja Mayerhofer (TU Wien, AT) [dblp]
- Peter D. Mosses (TU Delft, NL) [dblp]
- Gunter Mussbacher (McGill University – Montreal, CA) [dblp]
- Oscar M. Nierstrasz (Universität Bern, CH) [dblp]
- Anthony Sloane (Macquarie University – Sydney, AU) [dblp]
- Friedrich Steimann (Fernuniversität in Hagen, DE) [dblp]
- Eugene Syriani (University of Montréal, CA) [dblp]
- Tijs van der Storm (CWI – Amsterdam, NL) [dblp]
- Eric Van Wyk (University of Minnesota – Minneapolis, US) [dblp]
- Hans Vangheluwe (University of Antwerp, BE) [dblp]
- Jurgen J. Vinju (CWI – Amsterdam, NL) [dblp]
- Markus Völter (Völter Ingenieurbüro – Stuttgart, DE) [dblp]
- Vadim Zaytsev (RainCode – Brussels, BE) [dblp]

SLEBoK-style projects: Grammar Zoo (2009+)



Grammar Zoo

The objective of the Grammar Zoo is to accumulate grammars in a broad sense of various software languages, extracted and recovered from language documentation, parser specifications and other artefacts and make them available in a range of formats.

1029 entries and counting

1756 grammars: 1029 fetched + 639 extracted + 79 connected + 7 corrected + 1 recovered + 1 imported

Ada — API — Assembly — ATL — Automata — AWK — Basic — C — Clafer — Clojure — COBOL — C++ — C♯ — CSS — Dart — Doc — DSL — Eiffel — Erlang — Fortran — Gosu — Hardware — Haskell — HTML — Java — JavaScript — Logo — Lua — Markup — Meta — ML — Modula — Objective-C — Occam — OCL — Ontoware — Pascal — PHP — PL/I — Prolog — Python — Query — Smalltalk — Strings — Swift — Toy — Transformations — UML — XMLware — WIP

Ada (↑)

Ada 83 — Ada 95 — Ada 2005

Ada 83 [git]

Ichbiah [git] [ReadMe]

- Source: Jean D. Ichbiah, *Preliminary Ada reference manual. Syntax Summary.* ACM SIGPLAN Notices, Volume 14 Issue 6a, June 1979, pages E-1 to E-5 (142-146) [001]
- The **fetched** grammar is [src.syntax.summary.txt]
- dir : fetched | level : 0 | method : copy
- The **extracted** grammar is [grammar.bgf] [Browse now!]
- Files used: [connect.glue]
- of : Ada 83 | dir : extracted | level : 1 | method : typeover | fileused : connect.glue | toolused : grammarlab
- The **connected** grammar is [grammar.bgf] [Browse now!]
- Files used: [connect.glue]
- of : Ada 83 | dir : connected | level : 2 | method : semi-automated | fileused : connect.glue | toolused : grammarlab

LNCS 0106 [git] [ReadMe]

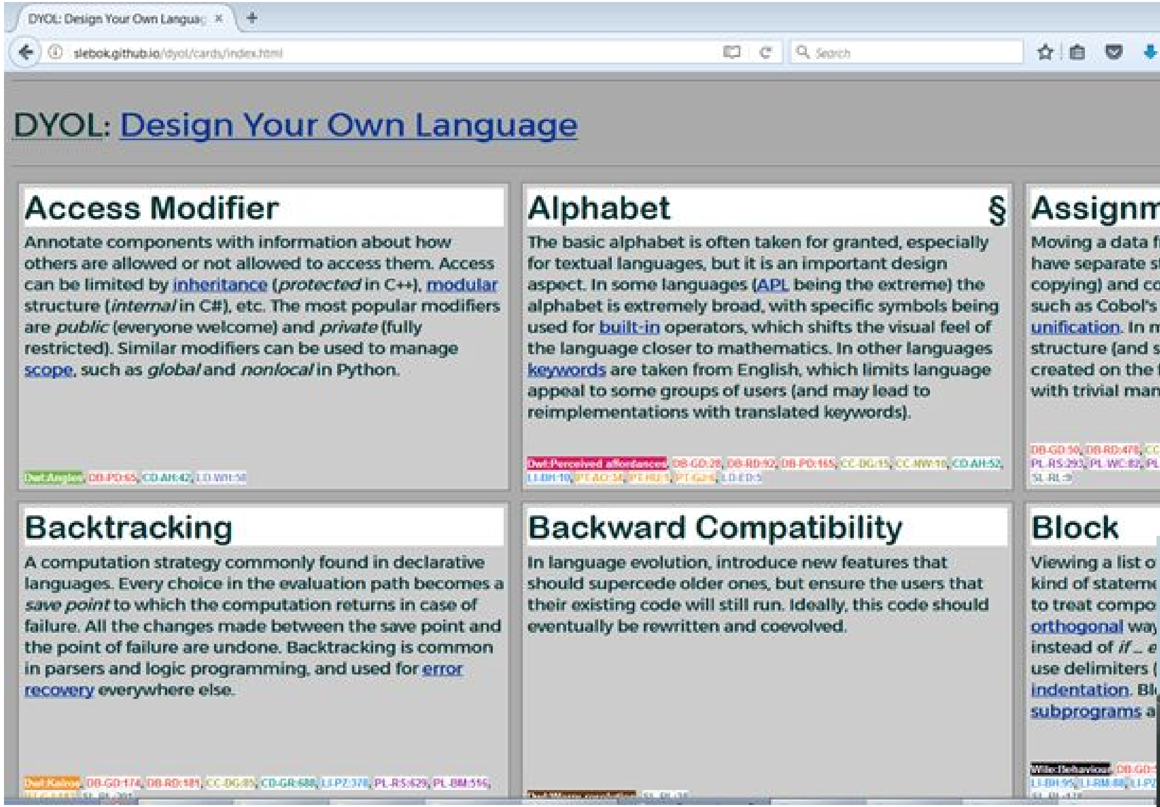
- Source: Proposed Standard Document United States Department of Defense, *The Programming Language Ada Reference Manual,* 1981, Appendix E: Syntax Summary, pages 221–225 [001]
- The **fetched** grammar is [src.syntax.summary.txt]
- dir : fetched | level : 0 | method : copy

LNCS 0155 [git] [ReadMe]

- Source: ANSI/MIL-STD-1815A-1983, *The Programming Language Ada Reference Manual,* 1990, Appendix E: Syntax Summary, pages E-1–E-6 (277–282) [001]
- The **fetched** grammar is [src.syntax.summary.txt]
- dir : fetched | level : 0 | method : copy

Ada 95 [git]

SLEBoK-style projects: DYOL (2017+)



DYOL: Design Your Own Languag
slebok.github.io/dyol/cards/index.html

DYOL: Design Your Own Language

Access Modifier

Annotate components with information about how others are allowed or not allowed to access them. Access can be limited by inheritance (*protected* in C++), modular structure (*internal* in C#), etc. The most popular modifiers are *public* (everyone welcome) and *private* (fully restricted). Similar modifiers can be used to manage scope, such as *global* and *nonlocal* in Python.

Dwl:Angles, DB-PD:65, CD-AH:42, LD-WH:58

Alphabet §

The basic alphabet is often taken for granted, especially for textual languages, but it is an important design aspect. In some languages (APL being the extreme) the alphabet is extremely broad, with specific symbols being used for built-in operators, which shifts the visual feel of the language closer to mathematics. In other languages keywords are taken from English, which limits language appeal to some groups of users (and may lead to reimplementations with translated keywords).

Dwl:Perceived affordances, DB-GD:28, DB-RD:92, DB-PD:165, CC-DG:15, CC-HW:10, CD-AH:52, LI-BH:10, PT-AO:34, PT-HU:1, PT-GJ:6, LD-ED:5

Assignment

Moving a data from one place to another. Some 4GLs have separate statements for straightforward (byte-copying) and composite (pattern-matching) assignments such as Cobol's *MOVE CORRESPONDING* which requires unification. In modern languages the source data structure (and sometimes the target one) can often be created on the fly. Many languages combine assignment with trivial manipulation (such as +=).

DB-GD:50, DB-RD:478, CC-WG:23, CD-AH:620, CD-SM:36, CD-GR:270, LI-RM:87, LI-PZ:291, PL-RS:293, PL-WC:82, PL-BM:105, PT-AO:206, LD-ED:3, LD-JW:20, LD-WH:54, SL-AS:190, SL-RL:9

Backtracking

A computation strategy commonly found in declarative languages. Every choice in the evaluation path becomes a *save point* to which the computation returns in case of failure. All the changes made between the save point and the point of failure are undone. Backtracking is common in parsers and logic programming, and used for error recovery everywhere else.

Dwl:Kairos, DB-GD:174, DB-RD:181, CC-DG:85, CD-GR:688, LI-PZ:378, PL-RS:629, PL-BM:516,

Backward Compatibility

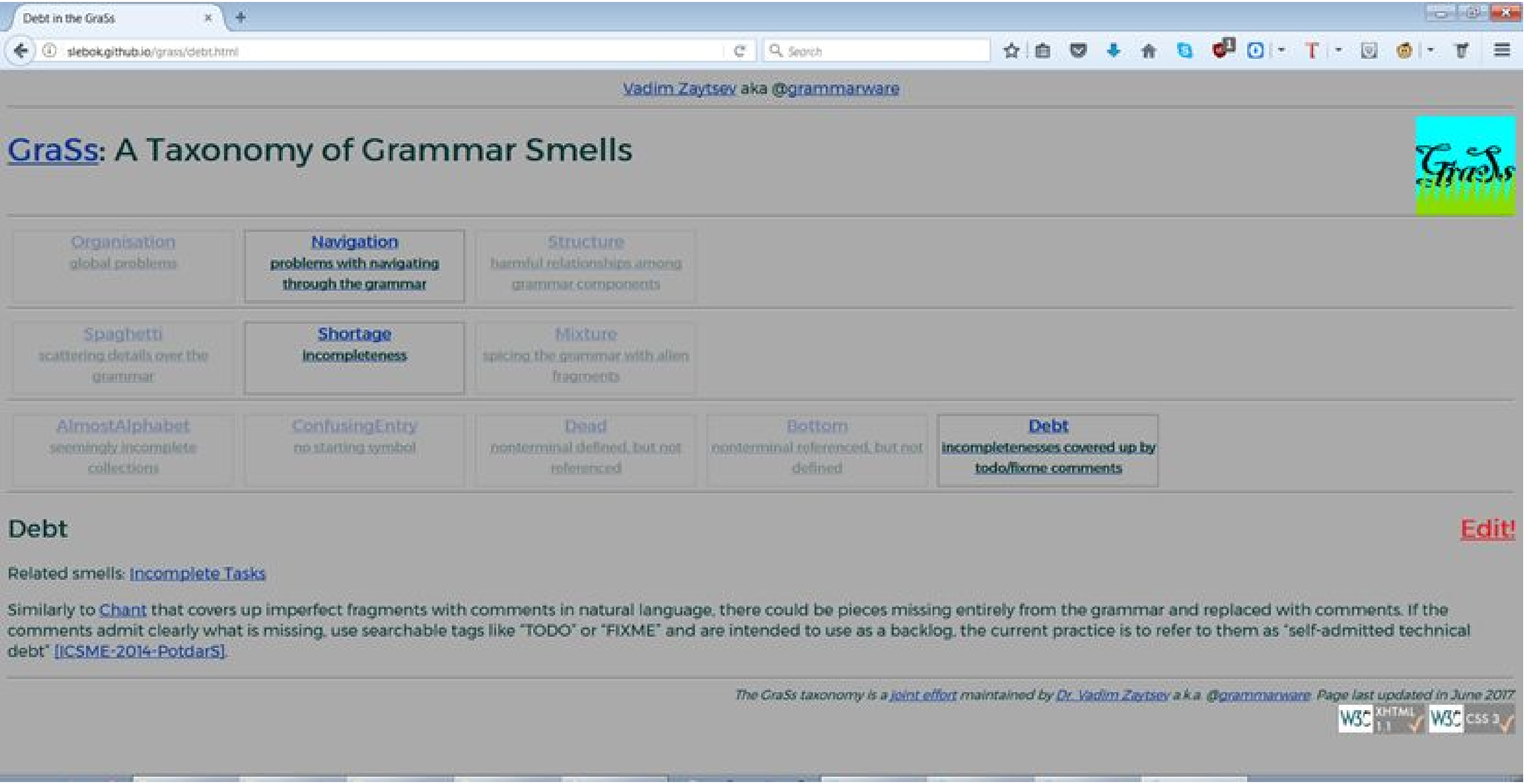
In language evolution, introduce new features that should supercede older ones, but ensure the users that their existing code will still run. Ideally, this code should eventually be rewritten and coevolved.

Block §

Viewing a list o
kind of stateme
to treat compo
orthogonal way
instead of *if ... e*
use delimiters (
indentation. Bl
subprograms a

Wiki:Behaviour, DB-GD:
LI-BH:95, LI-RM:88, LI-PZ

SLEBoK-style projects: GraSs (2017+)



Vadim Zaytsev aka @grammarware

GraSs: A Taxonomy of Grammar Smells

Organisation global problems	Navigation problems with navigating through the grammar	Structure harmful relationships among grammar components
Spaghetti scattering details over the grammar	Shortage incompleteness	Mixture splicing the grammar with alien fragments

AlmostAlphabet seemingly incomplete collections	ConfusingEntry no starting symbol	Dead nonterminal defined, but not referenced	Bottom nonterminal referenced, but not defined	Debt incompletenesses covered up by todo/fixme comments

Debt

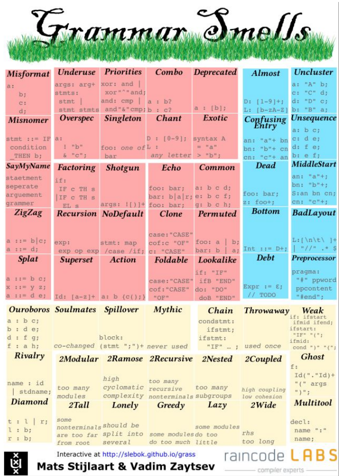
Edit!

Related smells: Incomplete Tasks

Similarly to Chant that covers up imperfect fragments with comments in natural language, there could be pieces missing entirely from the grammar and replaced with comments. If the comments admit clearly what is missing, use searchable tags like "TODO" or "FIXME" and are intended to use as a backlog, the current practice is to refer to them as "self-admitted technical debt" [ICSME-2014-PotdarS].

The GraSs taxonomy is a joint effort maintained by Dr. Vadim Zaytsev a.k.a. @grammarware. Page last updated in June 2017.

W3C XHTML 1.1
W3C CSS 3



Grammar Smells

<table>
<tr><td>Misformat
a:
b;
c:
d;</td><td>Underuse
args: arg+
stmts:
stmt |
stmt stmts</td><td>Priorities
xor: and |
xor"^"and;
and: cmp |
and"&"cmp;</td><td>Combo
a : b?
b : c?</td><td>Deprecated
a : [b];</td><td>Almost
D: [1-9]+;
L: [b-zA-Z]</td><td>Uncluster
a: "A" b;
c: "C" d;
d: "D" c;
b: "B" a;</td></tr>
<tr><td>Misnomer
stmt ::= IF
condition
_THEN b;</td><td>Overspec
a:
! "b"
& "c";</td><td>Singleton
foo: one of
bar</td><td>Chant
D : [0-9];
L :
any letter</td><td>Exotic
syntax A
= "a"
> "b";</td><td>Confusing Entry
an: "a"+ bn
bn: "b"+ cn
cn: "c"+ an</td><td>Unsequence
a: b c;
c: d e;
d: f e;
b: e f;</td></tr>
<tr><td>SayMyName
staetment
seperate
arguement
grammer</td><td>Factoring
if:
IF c TH s
|IF c TH s
EL s</td><td>Shotgun
args: ![)]+</td><td>Echo
foo: bar;
bar: b|a|r;
foo: bar;</td><td>Common
a: b c d;
e: b c f;
g: b c h;</td><td>Dead
foo: bar;
z: foo+;</td><td>MiddleStart
an: "a"+;
bn: "b"+;
S:an bn cn;
cn: "c"+;</td></tr>
<tr><td>ZigZag
a ::= b|c;
a ::= d;</td><td>Recursion
exp:
exp op exp</td><td>NoDefault
stmt: map
/case /if;</td><td>Clone
case:"CASE"
cof:c "OF"
c: "CASE"</td><td>Permuted
foo: a | b;
bar: b | a;</td><td>Bottom
Int ::= D+;</td><td>BadLayout
L:[\n\t\ ]+
| "//" .* $</td></tr>
<tr><td>Splat
a ::= b c;
x ::= y z;
a ::= d e;</td><td>Superset
Id: [a-z]+</td><td>Action
a: b {c();}</td><td>Foldable
case:"CASE"
cof:"CASE"<br>"OF"</td><td>Lookalike
if: "IF"
ifB "END"
do: "DO"
doB "END"</td><td>Debt
Expr := ε;
// TODO</td><td>Preprocessor
pragma:
"#" ppword
ppcontent
"#end";</td></tr>
<tr><td>Ouroboros
a : b c;
b : d e;
d : f g;
f : a h;</td><td>Soulmates
co-changed</td><td>Spillover
block:
(stmt ";")+</td><td>Mythic
never used</td><td>Chain
condstmt:
ifstmt;
ifstmt:
"IF" … ;</td><td>Throwaway
used once</td><td>Weak
if: ifstart
ifmid ifend;
ifstart:
"IF" "(";
ifmid:
cond ")" "{";</td></tr>
<tr><td>Rivalry
name : id
| stdname;</td><td>2Modular
too many
modules</td><td>2Ramose
high
cyclomatic
complexity</td><td>2Recursive
too many
recursive
nonterminals</td><td>2Nested
too many
subgroups</td><td>2Coupled
high coupling
low cohesion</td><td>Ghost
f:
Id("."Id)+
"(" args
")";</td></tr>
<tr><td>Diamond
t : l | r;
l : b;
r : b;</td><td>2Tall
some
nonterminals
are too far
from root</td><td>Lonely
should be
split into
several</td><td>Greedy
some modules
do too much</td><td>Lazy
some modules
do too
little</td><td>2Wide
rhs
too long</td><td>Multitool
decl:
name ":"
name;</td></tr>
</table>

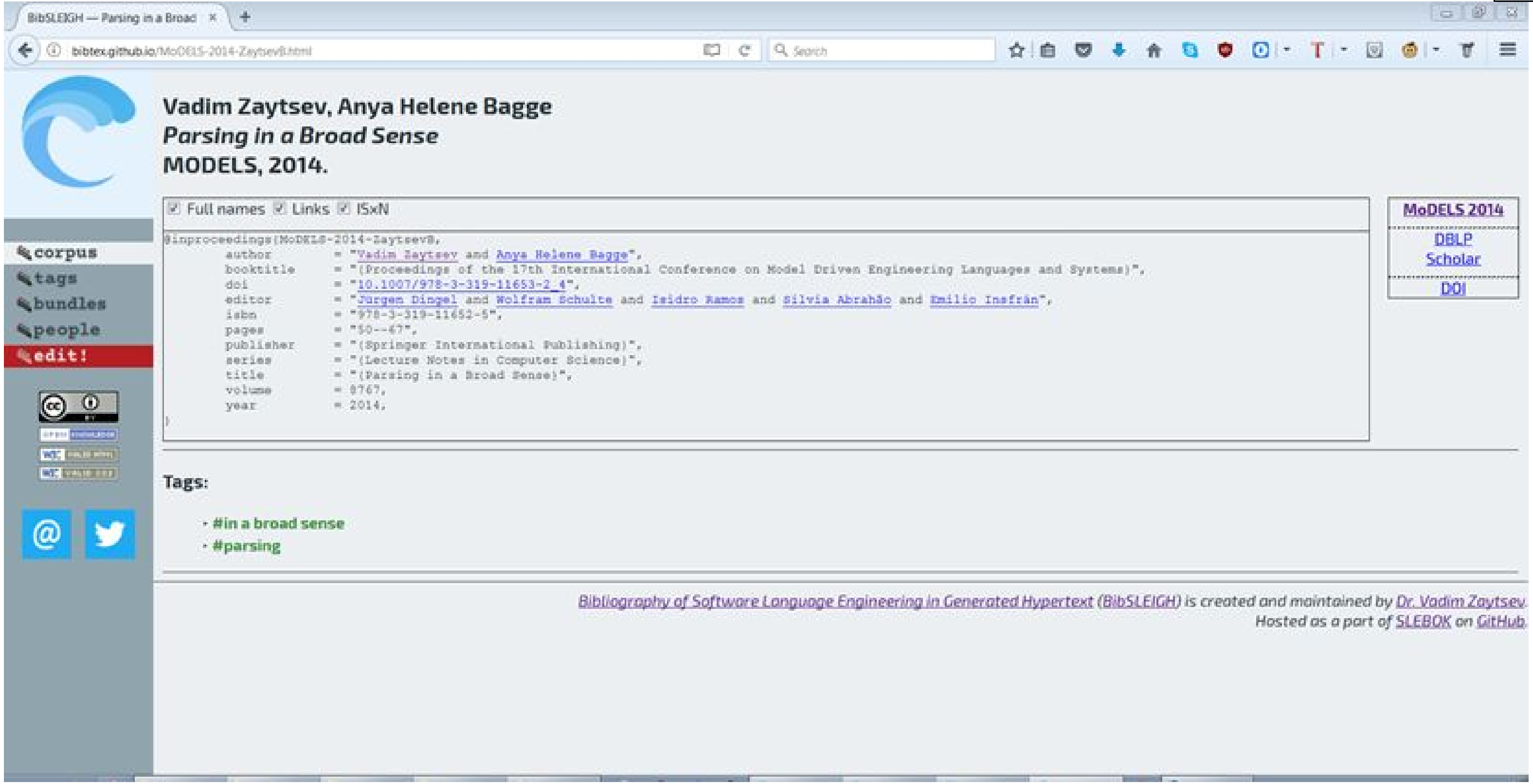
Interactive at http://slebok.github.io/grass

Mats Stijlaart & Vadim Zaytsev

raincode LABS

compiler experts

SLEBoK-style projects: BibSLEIGH (2014+)



BibSLEIGH — Parsing in a Broad
bibtex.github.io/MoDELS-2014-ZaytsevB.html
Search
corpus
tags
bundles
people
edit!

Vadim Zaytsev, Anya Helene Bagge
Parsing in a Broad Sense
MODELS, 2014.

☑ Full names ☑ Links ☑ ISxN

```
@inproceedings{MoDELS-2014-ZaytsevB,
        author          = "Vadim Zaytsev and Anya Helene Bagge",
        booktitle       = "{Proceedings of the 17th International Conference on Model Driven Engineering Languages and Systems}",
        doi             = "10.1007/978-3-319-11653-2_4",
        editor          = "Jürgen Dingel and Wolfram Schulte and Isidro Ramos and Silvia Abrahão and Emilio Insfrán",
        isbn            = "978-3-319-11652-5",
        pages           = "50--67",
        publisher       = "{Springer International Publishing}",
        series          = "{Lecture Notes in Computer Science}",
        title           = "{Parsing in a Broad Sense}",
        volume          = 8767,
        year            = 2014,
}
```

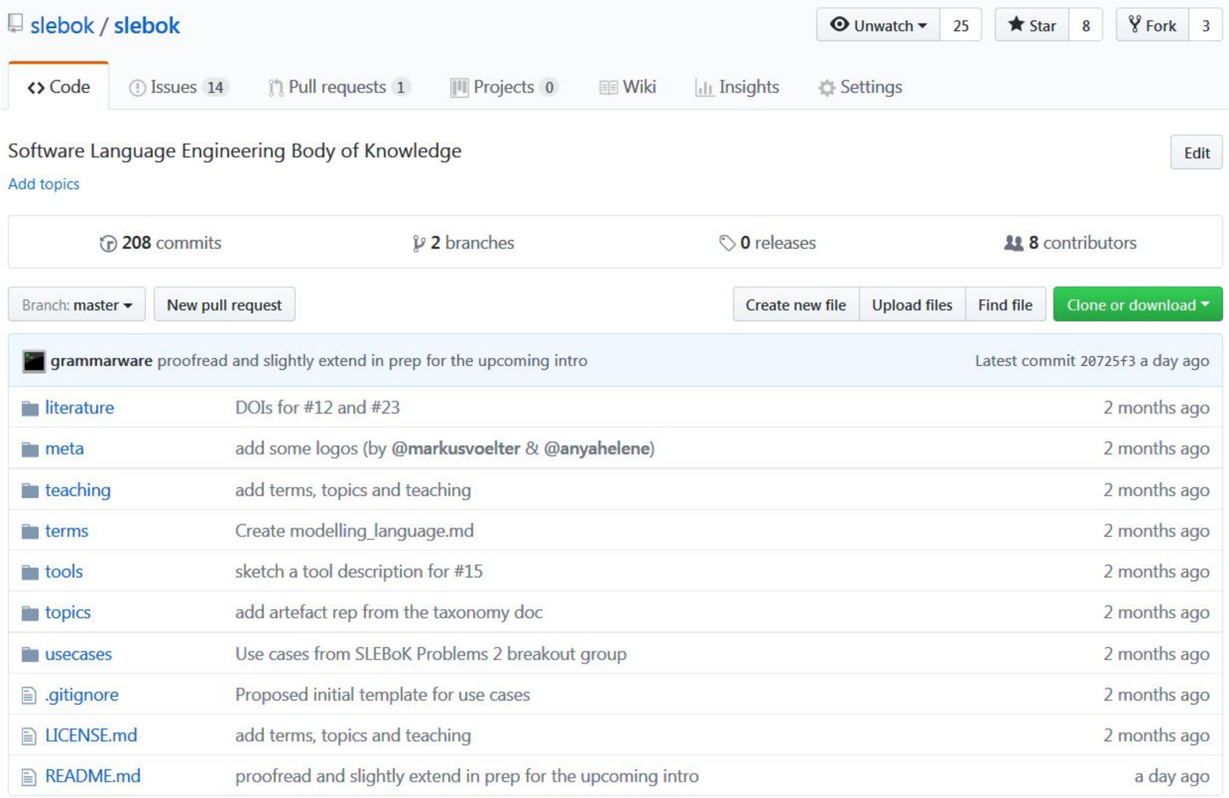
MoDELS 2014
DBLP
Scholar
DOI

Tags:

- #in a broad sense
- #parsing

Bibliography of Software Language Engineering in Generated Hypertext (BibSLEIGH) is created and maintained by Dr. Vadim Zaytsev.
Hosted as a part of SLEBOK on GitHub.

The new place for SLEBoK



Software Language Engineering Body of Knowledge

grammarware proofread and slightly extend in prep for the upcoming intro — Latest commit 20725f3 a day ago

literature	DOIs for #12 and #23	2 months ago
meta	add some logos (by @markusvoelter & @anyahelene)	2 months ago
teaching	add terms, topics and teaching	2 months ago
terms	Create modelling_language.md	2 months ago
tools	sketch a tool description for #15	2 months ago
topics	add artefact rep from the taxonomy doc	2 months ago
usecases	Use cases from SLEBoK Problems 2 breakout group	2 months ago
.gitignore	Proposed initial template for use cases	2 months ago
LICENSE.md	add terms, topics and teaching	2 months ago
README.md	proofread and slightly extend in prep for the upcoming intro	a day ago

https://github.com/slebok/slebok

Already contributing!

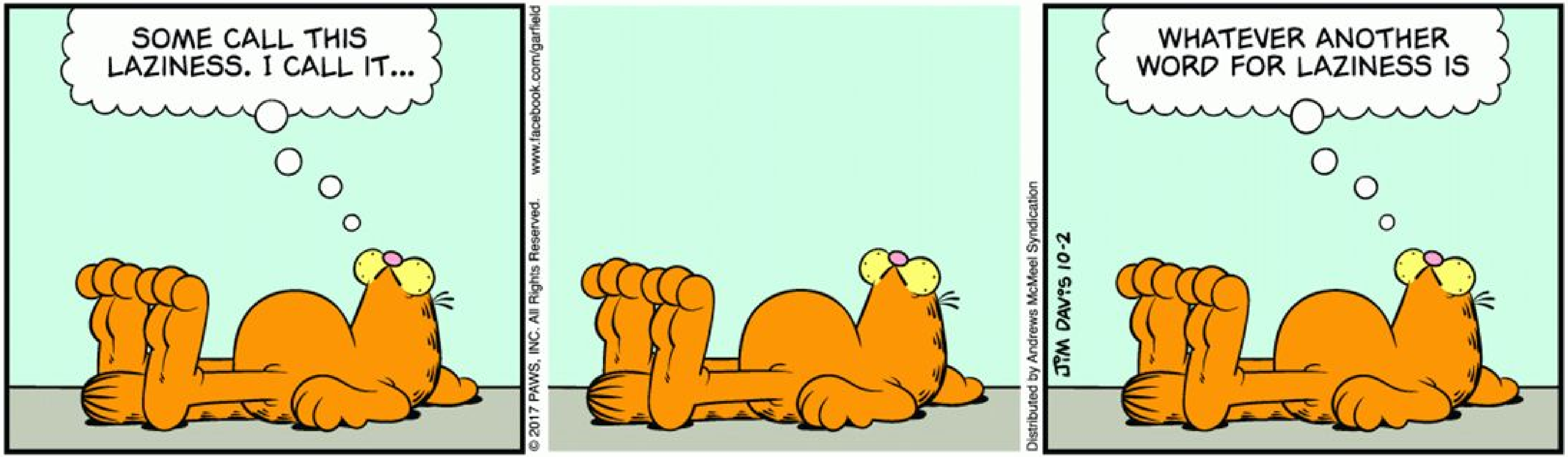
- Anya Helene Bagge
- Benoît Combemale
- Görel Hedin
- Manuel Leduc
- Marcel Heinz
- Oscar Nierstrasz
- Friedrich Steimann
- Tijs van der Storm
- Tony Sloane
- Vadim Zaytsev

```
git log | grep 'Author: ' | sort | uniq
```

Active SLEBoK initiatives

- Modular and reusable language design and implementation
  - Peter Mosses
- Practical guide to parsing
  - Jurgen Vinju
- Glossary, taxonomy, ontology of SLE terms [link]
  - Anya Helene Bagge
- Attribute grammars [link]
  - Tony Sloane
- SLE curricula [link]
  - Ralf Lämmel
- Use cases for SLE [link]
  - Oscar Nierstrasz

Get in touch!


SOME CALL THIS LAZINESS. I CALL IT...
WHATEVER ANOTHER WORD FOR LAZINESS IS
JIM DAVIS 10-2

© 2017 PAWS, INC. All Rights Reserved. www.facebook.com/garfield

Distributed by Andrews McMeel Syndication

https://garfield.com/comic/2017/10/02