Compiler Construction
Introduction

Guido Wachsmuth

TU Delft
Delft University of Technology

Challenge the future

Course IN4303, 2013/14
Compiler Construction

Overview

today's lecture

I
software languages

III
compiler construction

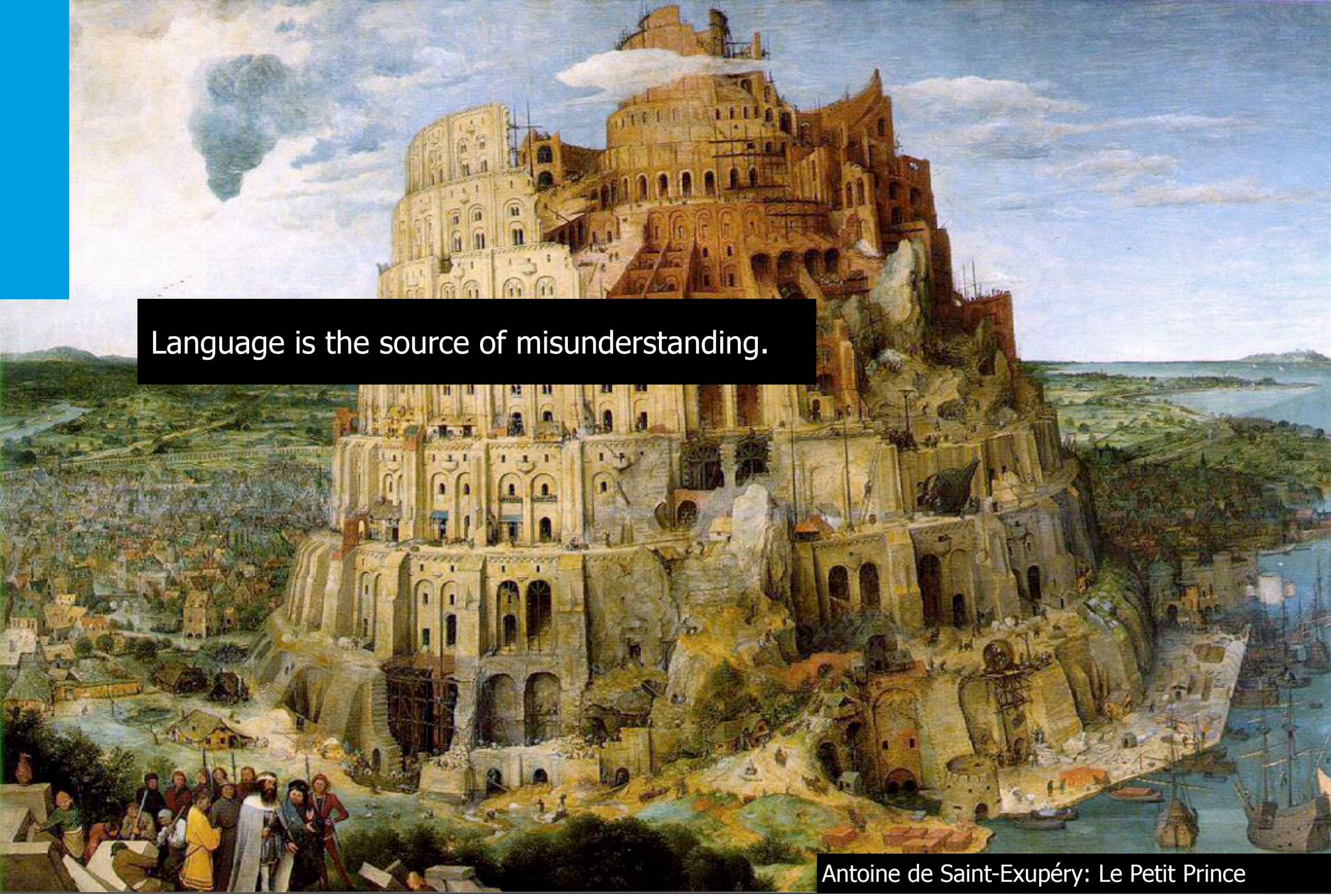
II
modern compilers

IV
organisation

I

software languages


language



Language is the source of misunderstanding.

Antoine de Saint-Exupéry: Le Petit Prince

Language is a

Edward Sapir: Language. An Introduction to the Study of Speech.

Language is a

purely human and non-instinctive method

Edward Sapir: Language. An Introduction to the Study of Speech.

Language is a

purely human and non-instinctive method

of communicating

Edward Sapir: Language. An Introduction to the Study of Speech.

Language is a

purely human and non-instinctive method

of communicating

ideas, emotions, and desires

Edward Sapir: Language. An Introduction to the Study of Speech.

Language is a

purely human and non-instinctive method

of communicating

ideas, emotions, and desires

by means of a system

Edward Sapir: Language. An Introduction to the Study of Speech.

Language is a

purely human and non-instinctive method

of communicating

ideas, emotions, and desires

by means of a system

of voluntarily produced symbols.

Edward Sapir: Language. An Introduction to the Study of Speech.

Language
properties

Language
properties

arbitrary

Language
properties

arbitrary

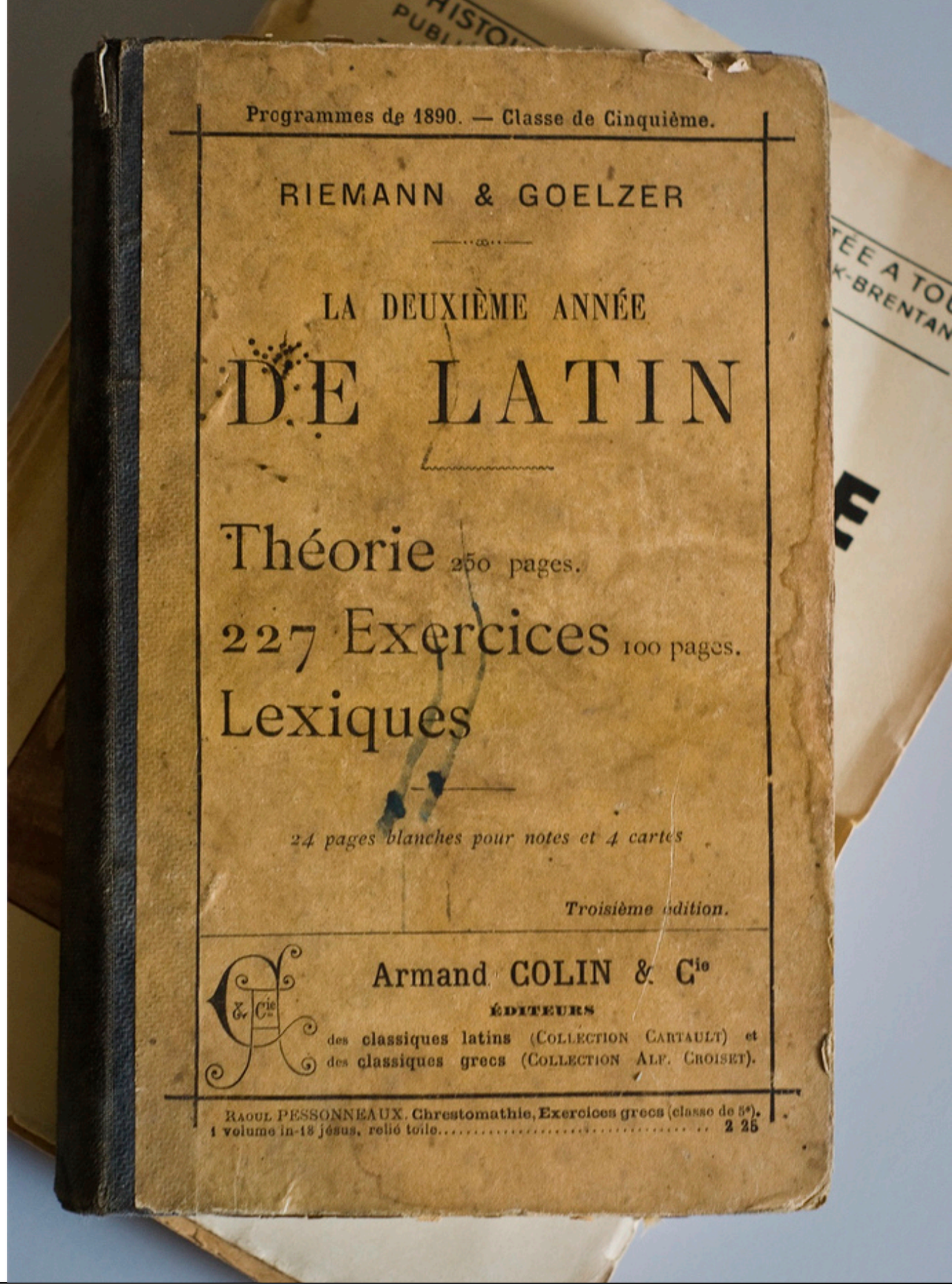
symbolic

Language
properties

arbitrary

symbolic

systematic



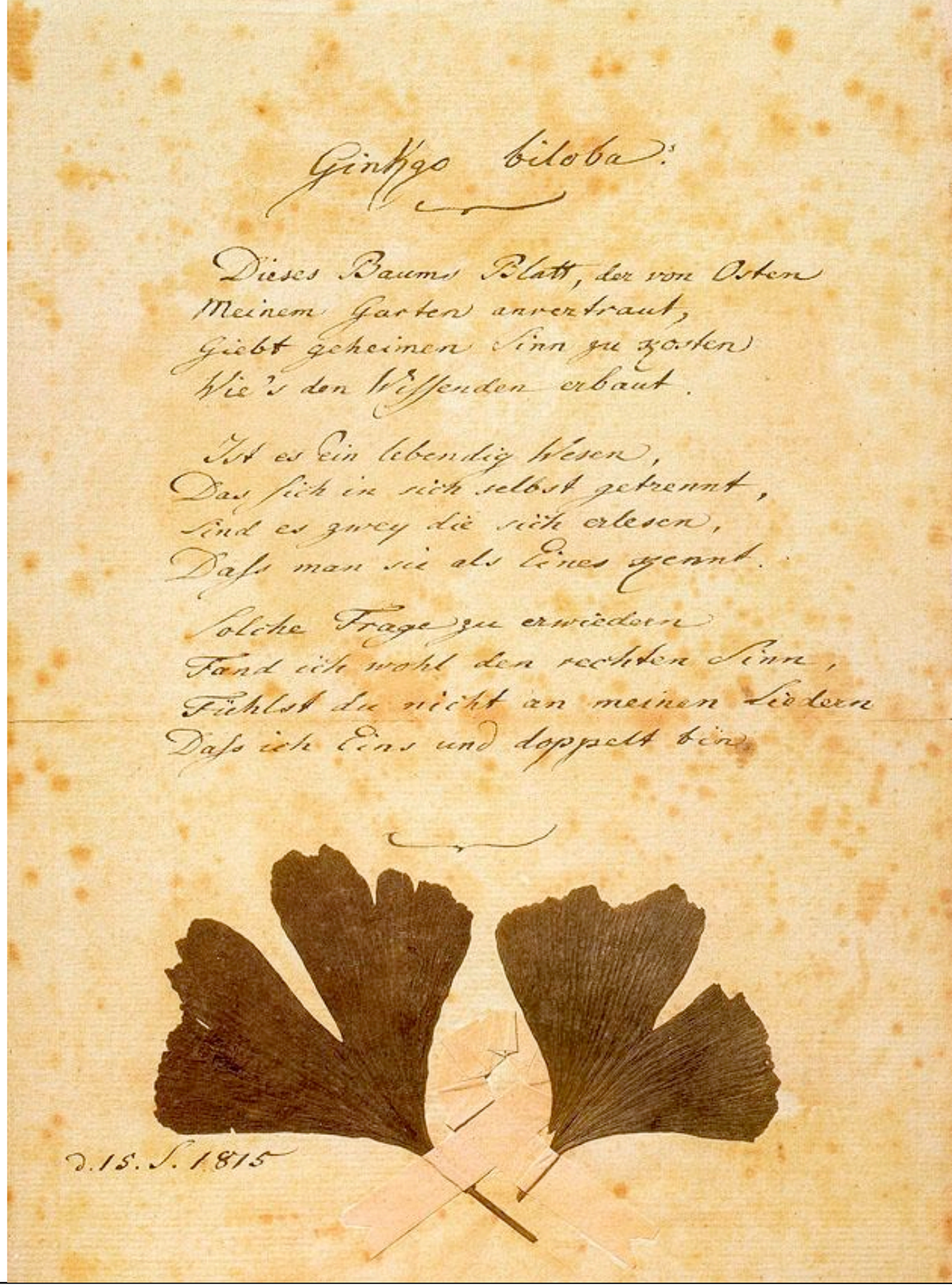
Language
properties

arbitrary

symbolic

systematic

productive



Language
properties

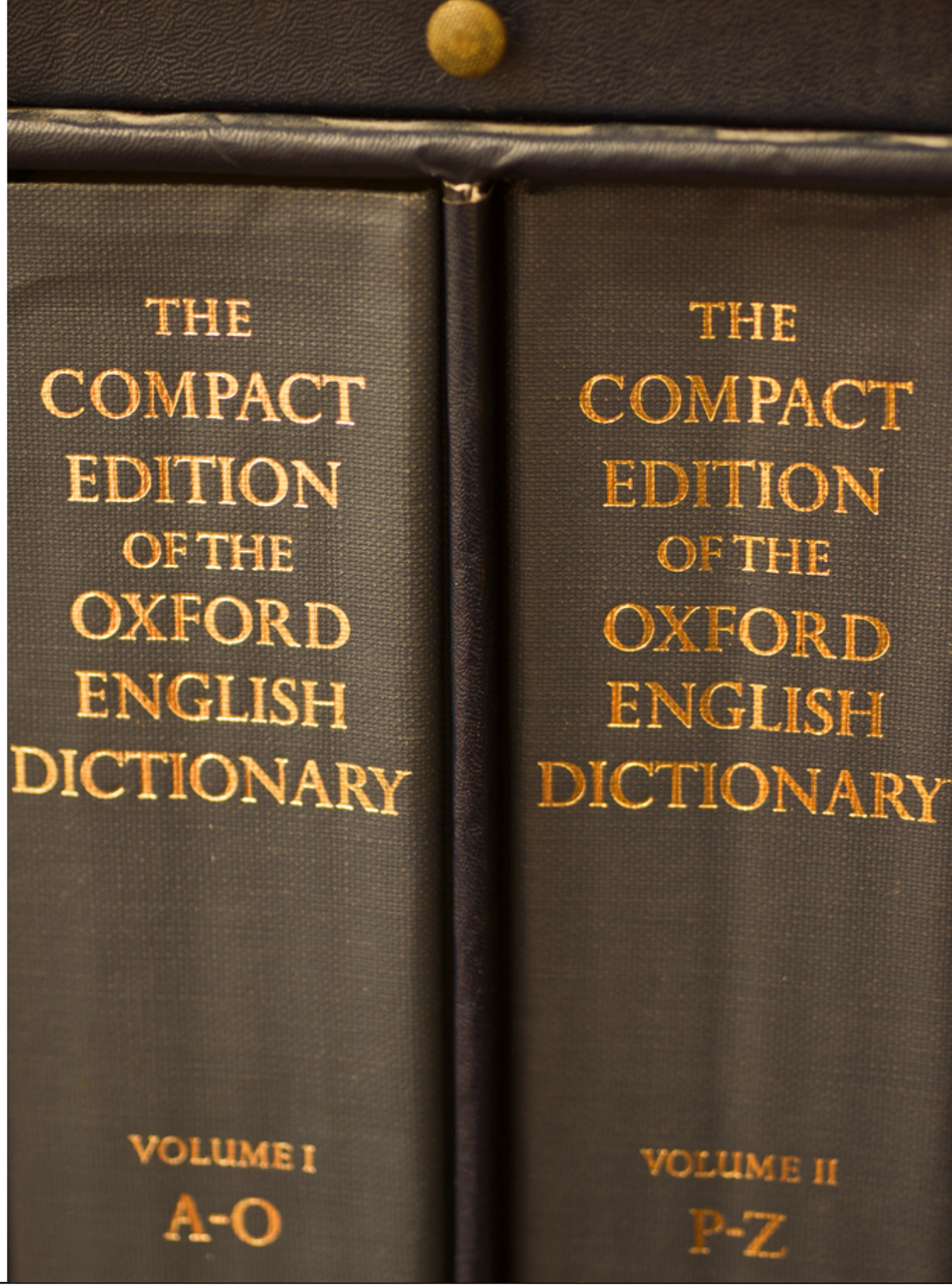
arbitrary

symbolic

systematic

productive

non-instinctive



Language
properties

arbitrary

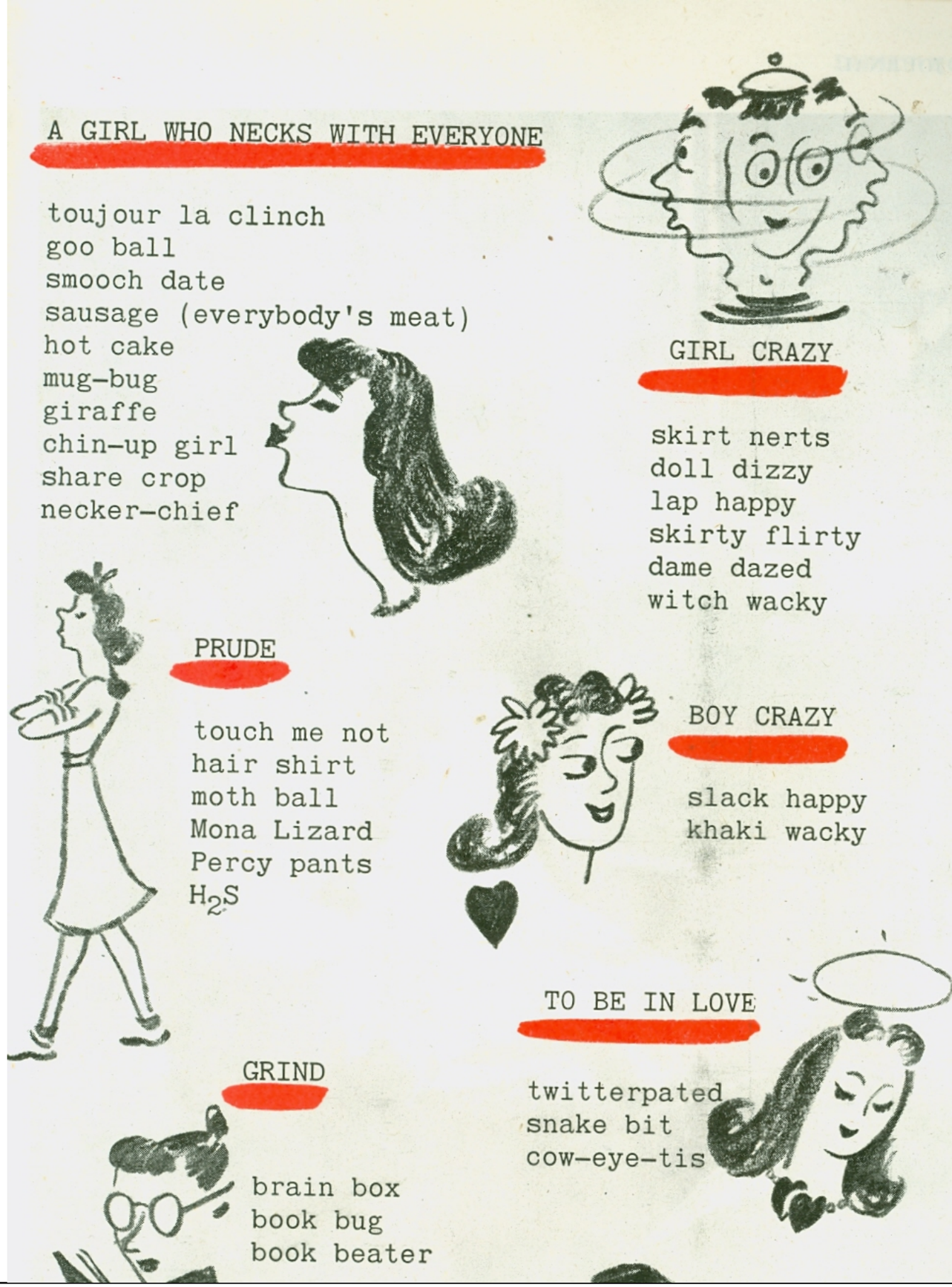
symbolic

systematic

productive

non-instinctive

conventional



Language
properties

arbitrary

symbolic

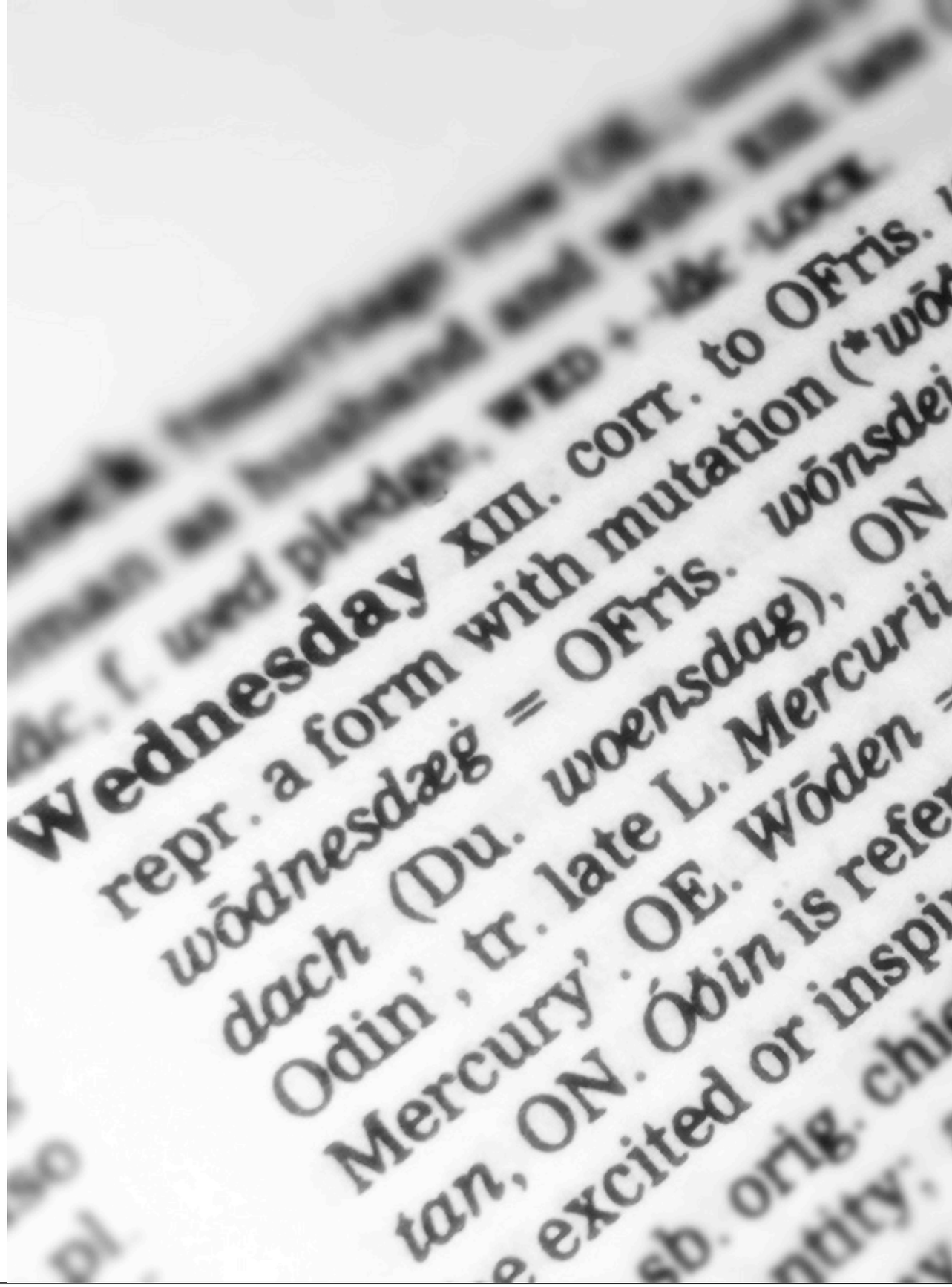
systematic

productive

non-instinctive

conventional

modifiable




metalanguage facility

Study of Language

linguistics and friends

philosophy

Study of Language

linguistics and friends

philosophy

linguistics

- lexicology
- grammar
  - morphology
  - syntax
  - phonology
- semantics

Study of Language

linguistics and friends

philosophy

linguistics

- lexicology
- grammar
  - morphology
  - syntax
  - phonology
- semantics

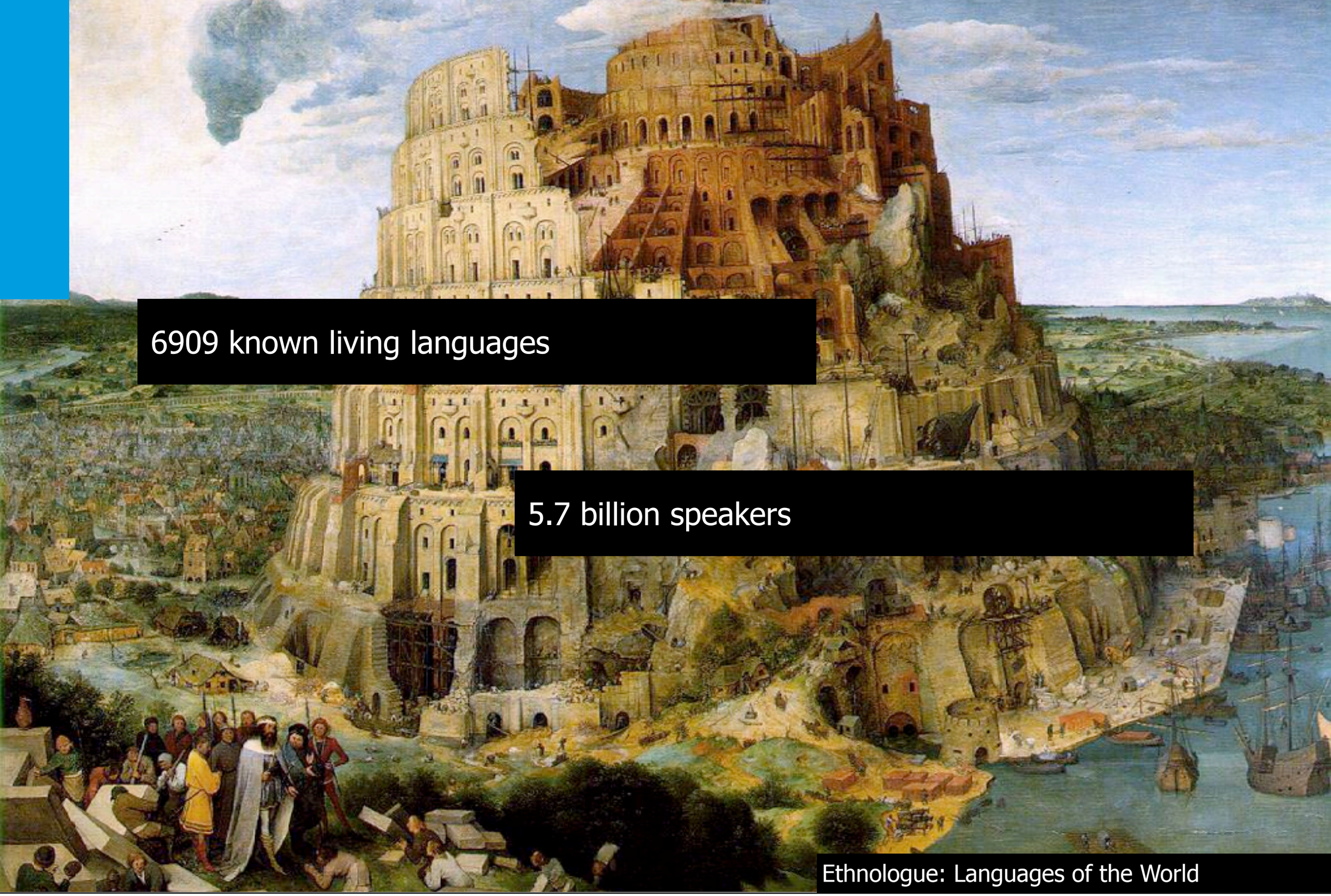
interdisciplinary

- applied linguistics
- computational linguistics
- historical linguistics
- neurolinguistics
- psycholinguistics
- sociolinguistics

What is a language?

A language is a dialect with an army and navy.

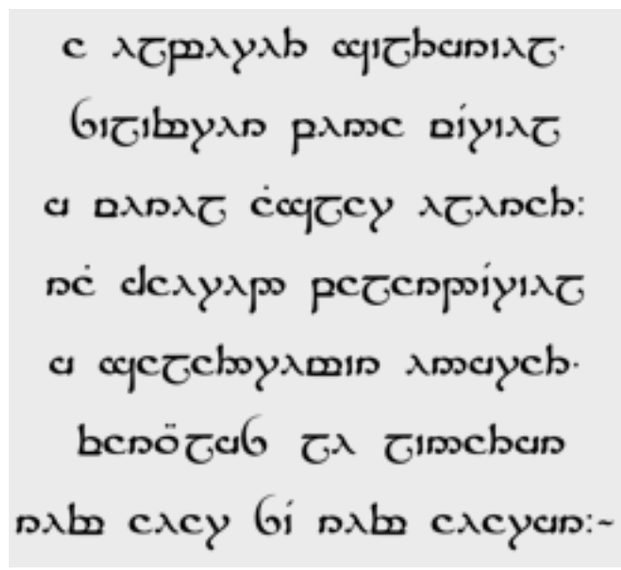
Max Weinreich: YIVO Bletter 25(1)

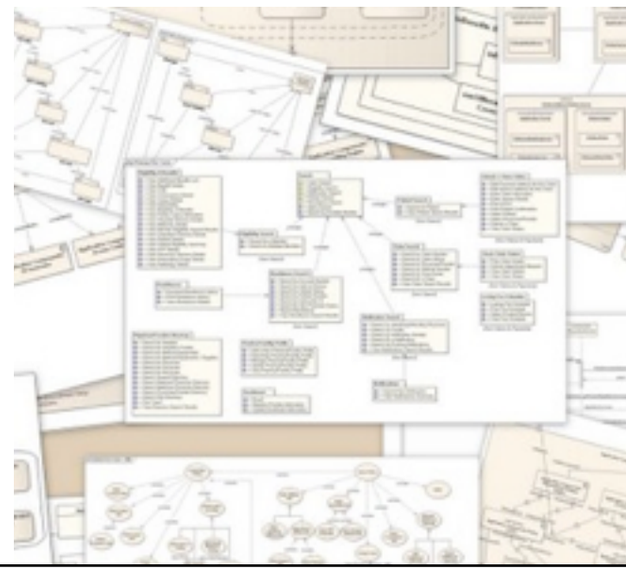

6909 known living languages
5.7 billion speakers
Ethnologue: Languages of the World

Language Control

towards artificial languages

Sanskrit Duden Simplified English



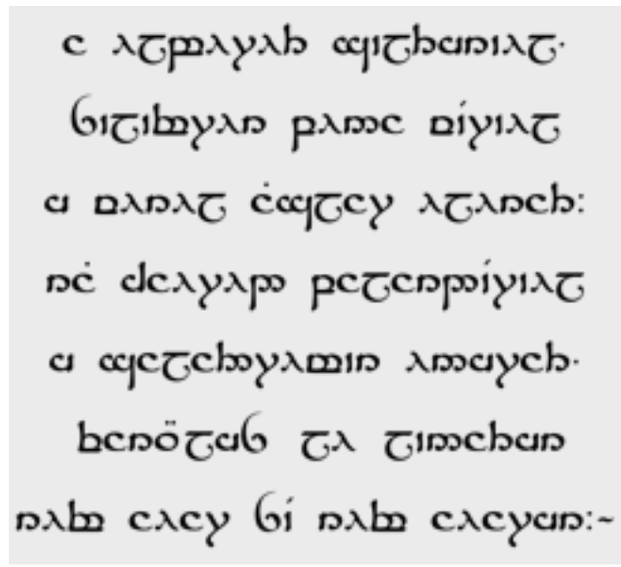


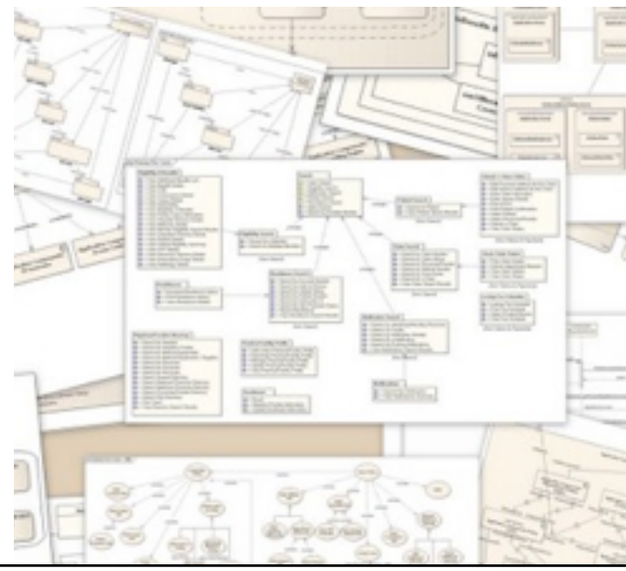
Language Control

towards artificial languages

Sanskrit Duden Simplified English

Esperanto Ido Interlingua





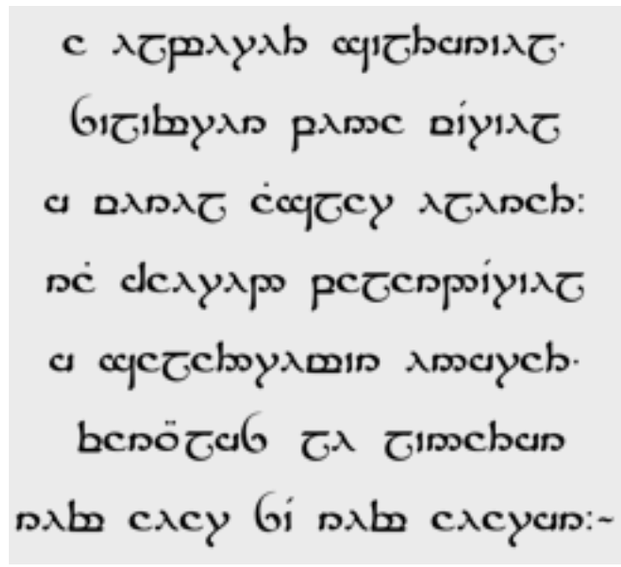
Language Control

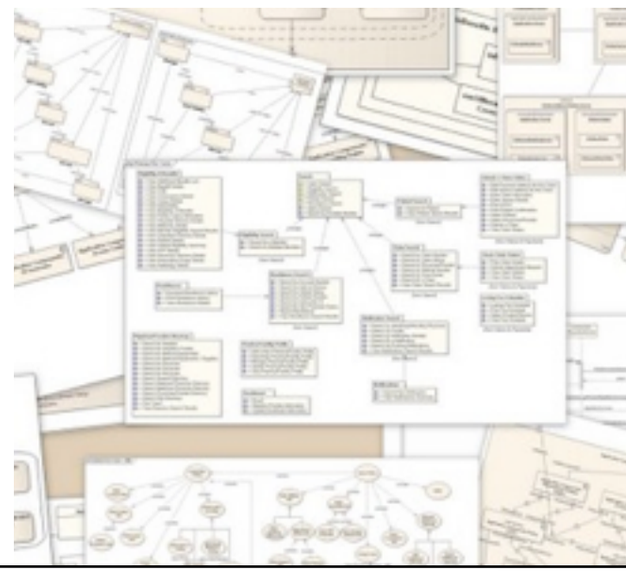
towards artificial languages

Sanskrit Duden Simplified English

Esperanto Ido Interlingua

Quenya Klingon Na’vi





Language Control

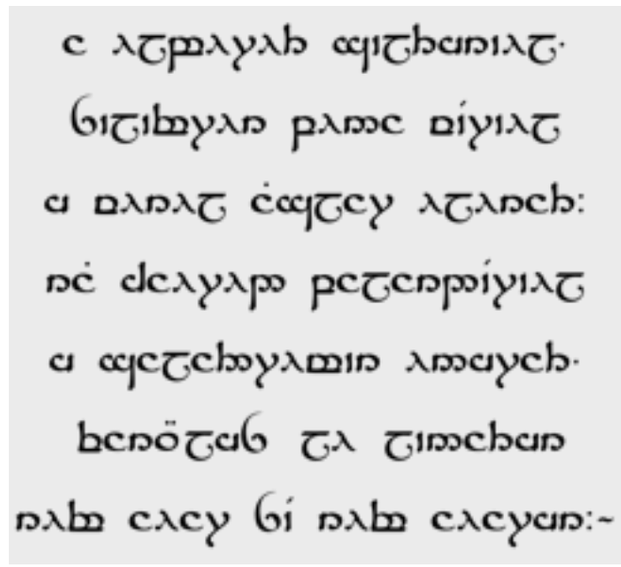
towards artificial languages

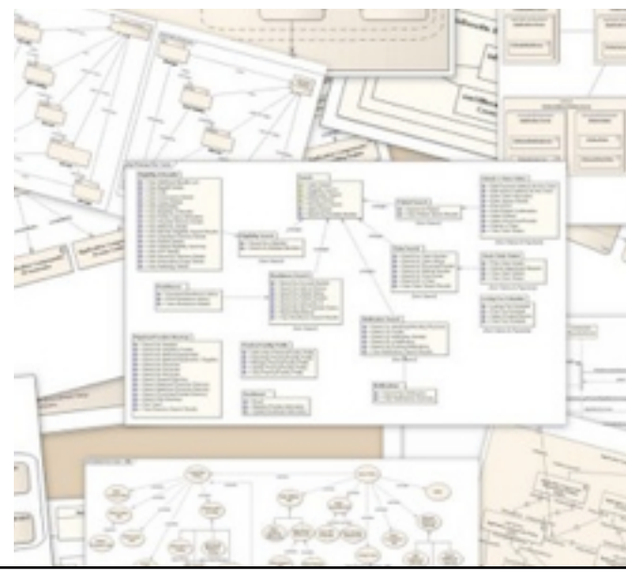
Sanskrit Duden Simplified English

Esperanto Ido Interlingua

Quenya Klingon Na'vi

International Algebraic Language





Language Control

towards artificial languages

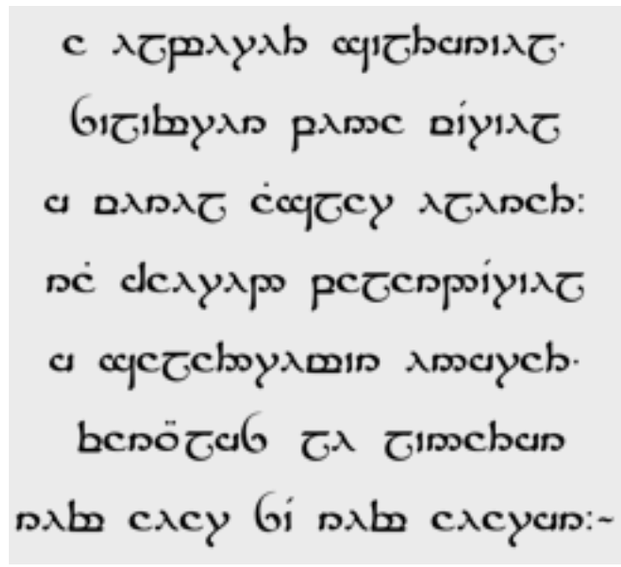
Sanskrit Duden Simplified English

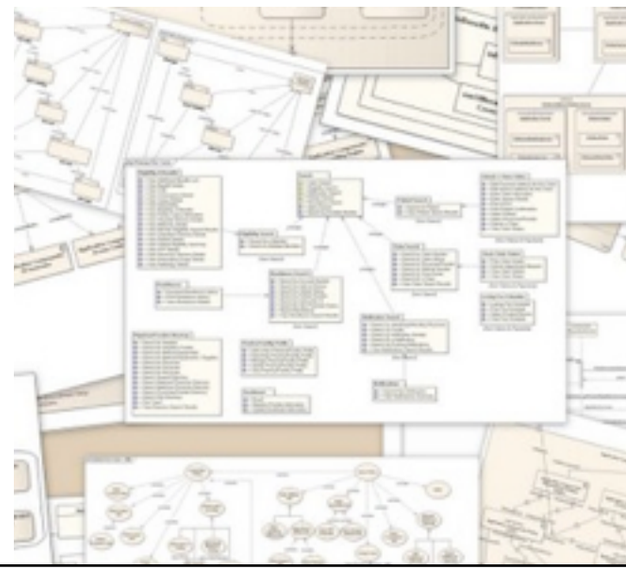
Esperanto Ido Interlingua

Quenya Klingon Na'vi

International Algebraic Language

Unified Modeling Language





To provide a means of communicating numerical methods and other procedures between people.

John W. Backus: The Syntax and Semantics of the Proposed International Algebraic Language of Zürich ACM-GAMM Conference.

To provide a means of communicating numerical methods and other procedures between

people.

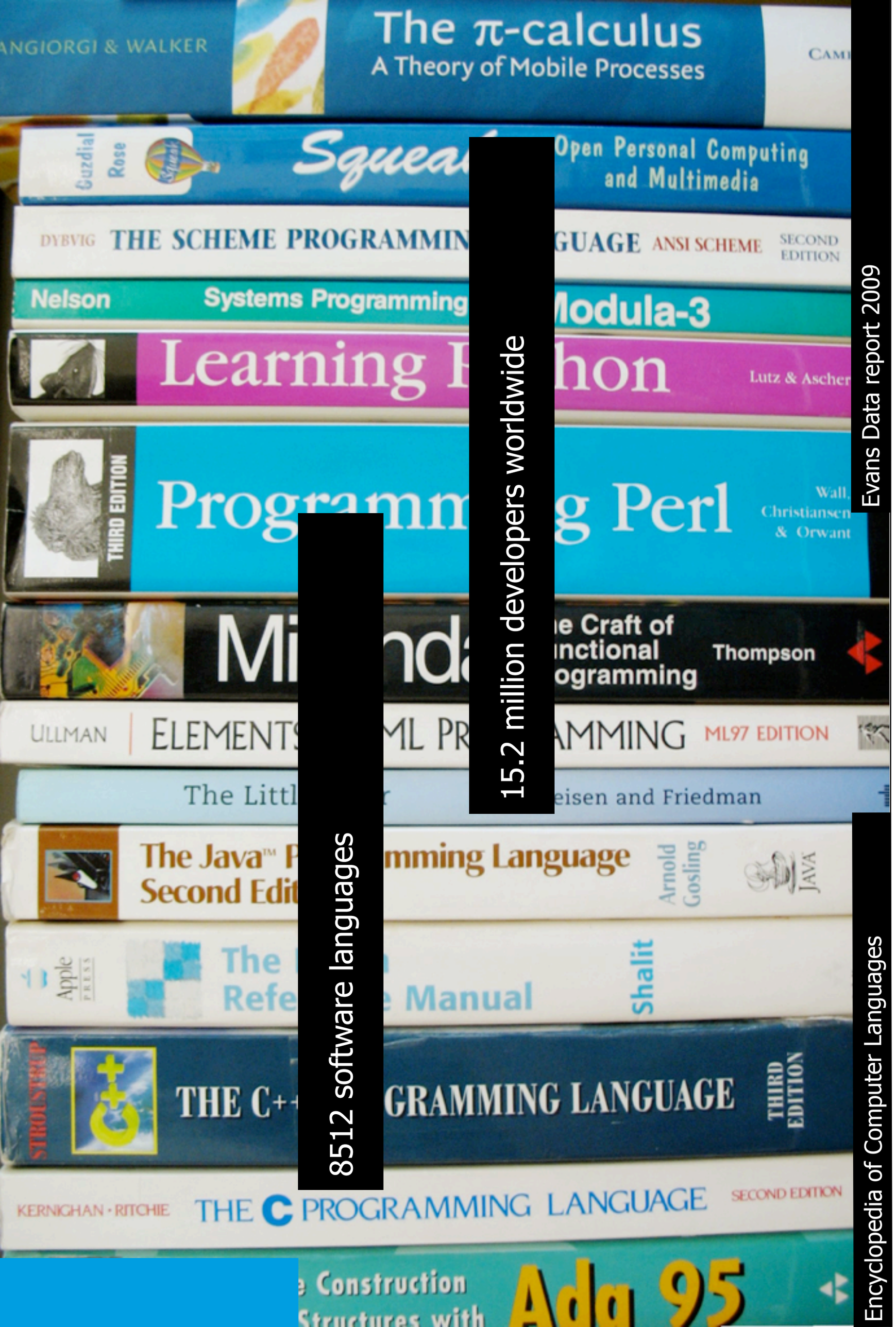
To provide a means for realising a stated process

on a variety of machines …

John W. Backus: The Syntax and Semantics of the Proposed International Algebraic Language of Zürich ACM-GAMM Conference.

Software Languages

languages to engineer software


The π-calculus
A Theory of Mobile Processes
Open Personal Computing and Multimedia
THE SCHEME PROGRAMMIN
GUAGE ANSI SCHEME SECOND EDITION
Nelson Systems Programming
Modula-3
Learning
hon
Lutz & Ascher
THIRD EDITION
Programm
g Perl
Wall, Christiansen & Orwant
Mi nda
e Craft of
nctional
ogramming
Thompson
ULLMAN
ELEMENTS
ML PR
AMMING
ML97 EDITION
The Littl
eisen and Friedman
The Java™ P
Second Edit
mming Language
Arnold Gosling
JAVA
Apple PRESS
The
Refe
e Manual
Shalit
STROUSTRUP
THE C++
GRAMMING LANGUAGE
THIRD EDITION
KERNIGHAN · RITCHIE
THE C PROGRAMMING LANGUAGE
SECOND EDITION
e Construction
Structures with
Ada 95
8512 software languages
15.2 million developers worldwide
Evans Data report 2009
Encyclopedia of Computer Languages

TU Delft

Study of Language

linguistics and friends

philosophy

linguistics

- lexicology
- grammar
  - morphology
  - syntax
  - phonology
- semantics

interdisciplinary

- applied linguistics
- computational linguistics
- historical linguistics
- neurolinguistics
- psycholinguistics
- sociolinguistics

Study of Language

linguistics and friends

philosophy

linguistics

- lexicology
- grammar
  - morphology
  - syntax
  - phonology
- semantics

interdisciplinary

- applied linguistics
- computational linguistics
- historical linguistics
- neurolinguistics
- psycholinguistics
- sociolinguistics

computer science

- grammar
- semantics

Tiger
the lecture language

```
/* factorial function */

let

   var x := 0

   function fact(n : int) : int =
      if n < 1 then 1 else (n * fact(n - 1))

in

   for i := 1 to 3 do (
      x := x + fact(i);
      printint(x);
      print(" ")
   )

end
```

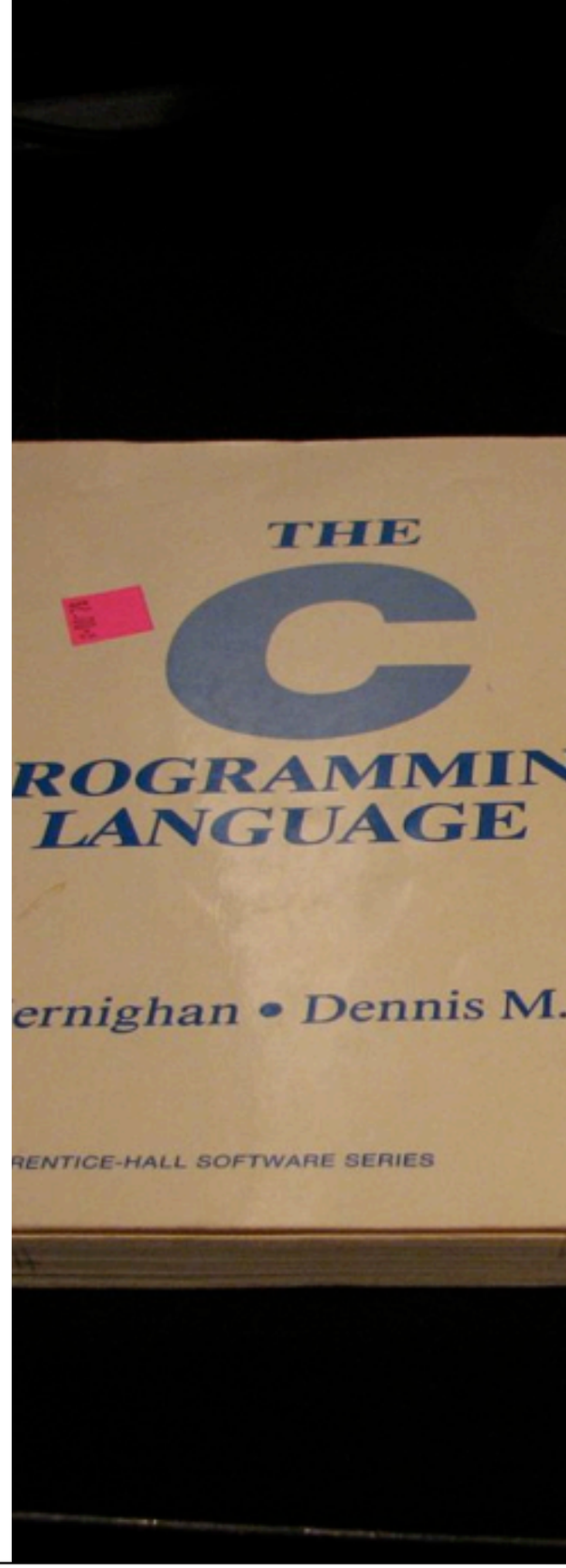
C
another lecture language

```c
#include <stio.h>

/* factorial function */

int fac(int num) {
    if (num < 1)
        return 1;
    else
        return num * fac(num - 1);
}

int main() {
    printf("%d! = %d\n", 10, fac(10));
    return 0;
}
```



MiniJava

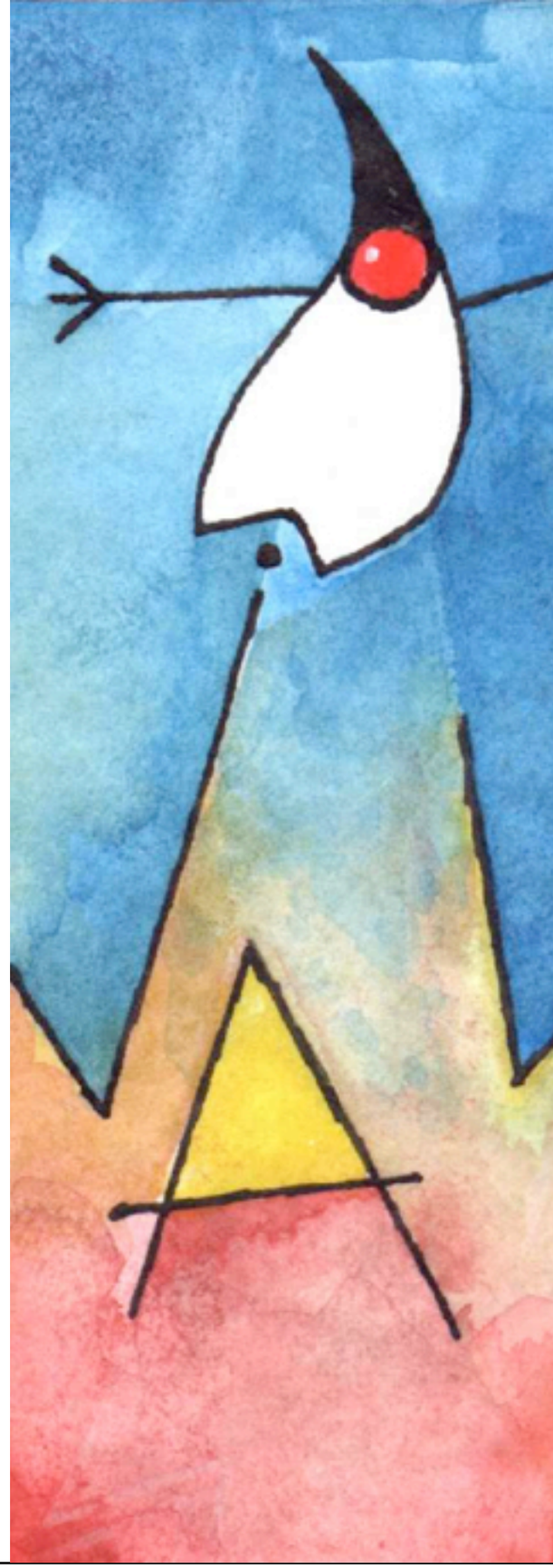
the lab language

```
class Main {

    public static void main(String[] args) {
        System.out.println(new Fac().fac(10));
    }
}

class Fac {

    public int fac(int num) {
        int num_aux;
        if (num < 1)
            num_aux = 1;
        else
            num_aux = num * this.fac(num - 1);
        return num_aux;
    }
}
```



II

modern compilers

To provide a means of communicating numerical methods and other procedures between

people.

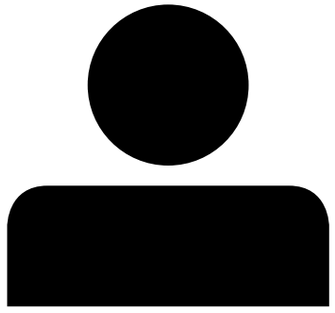
To provide a means for realising a stated process

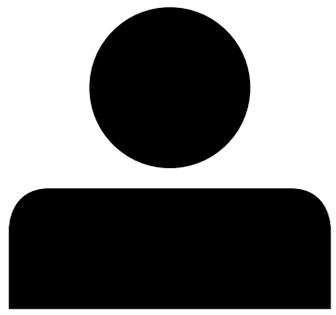
on a variety of machines …

John W. Backus: The Syntax and Semantics of the Proposed International Algebraic Language of Zürich ACM-GAMM Conference.

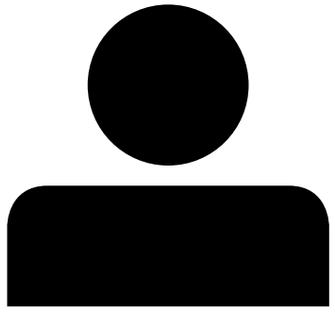
Communication

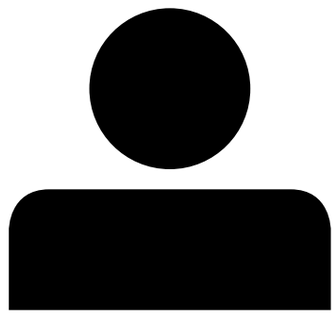
between people





Communication
between people

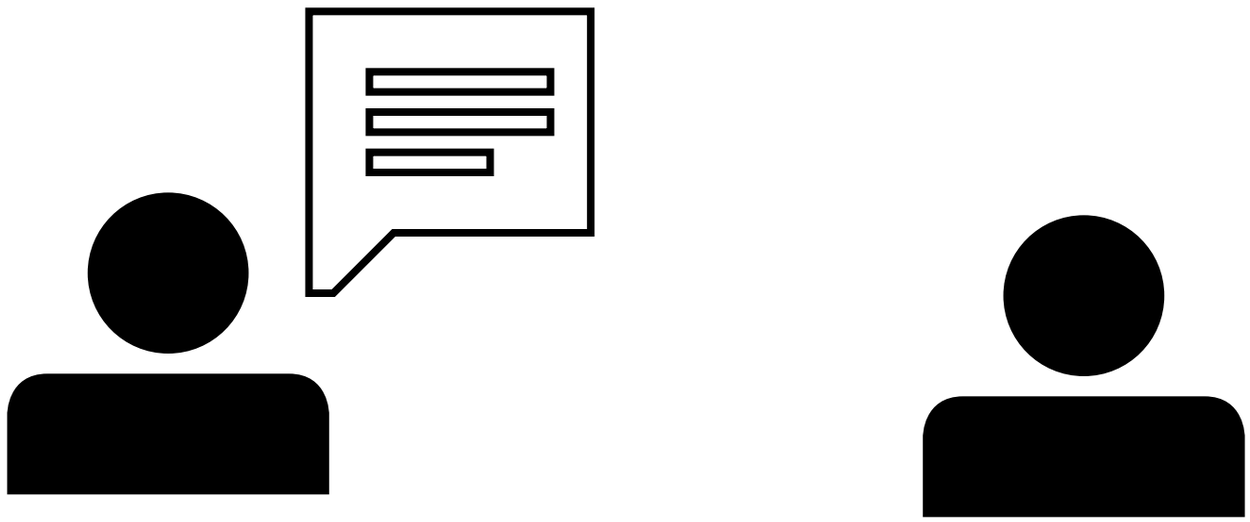




natural language

Communication

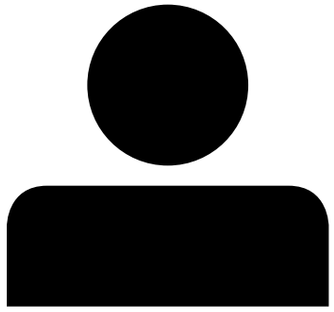
between people

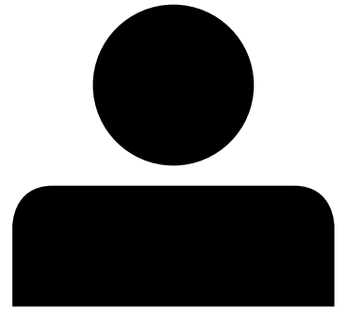


natural language

Communication

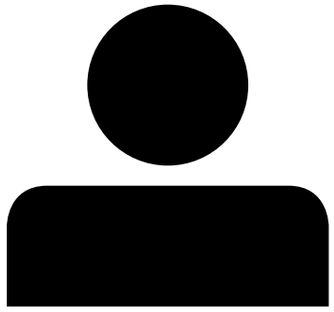
translation



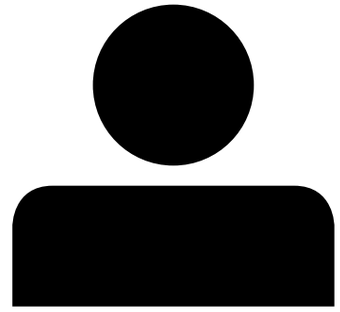


Communication

translation



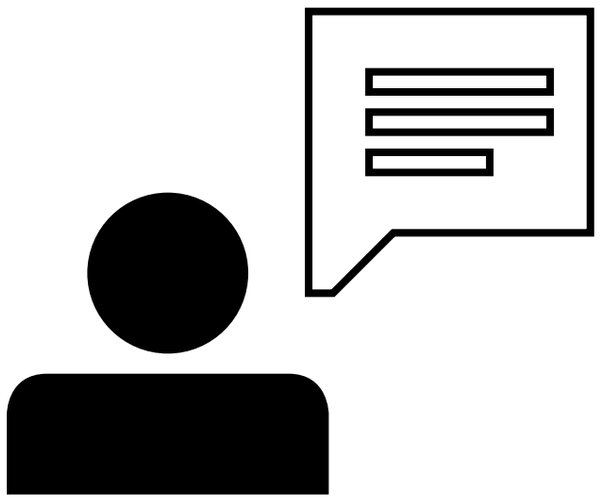
natural language



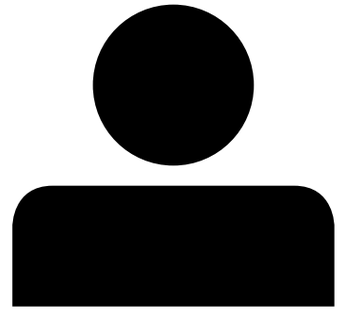
natural language

Communication

translation



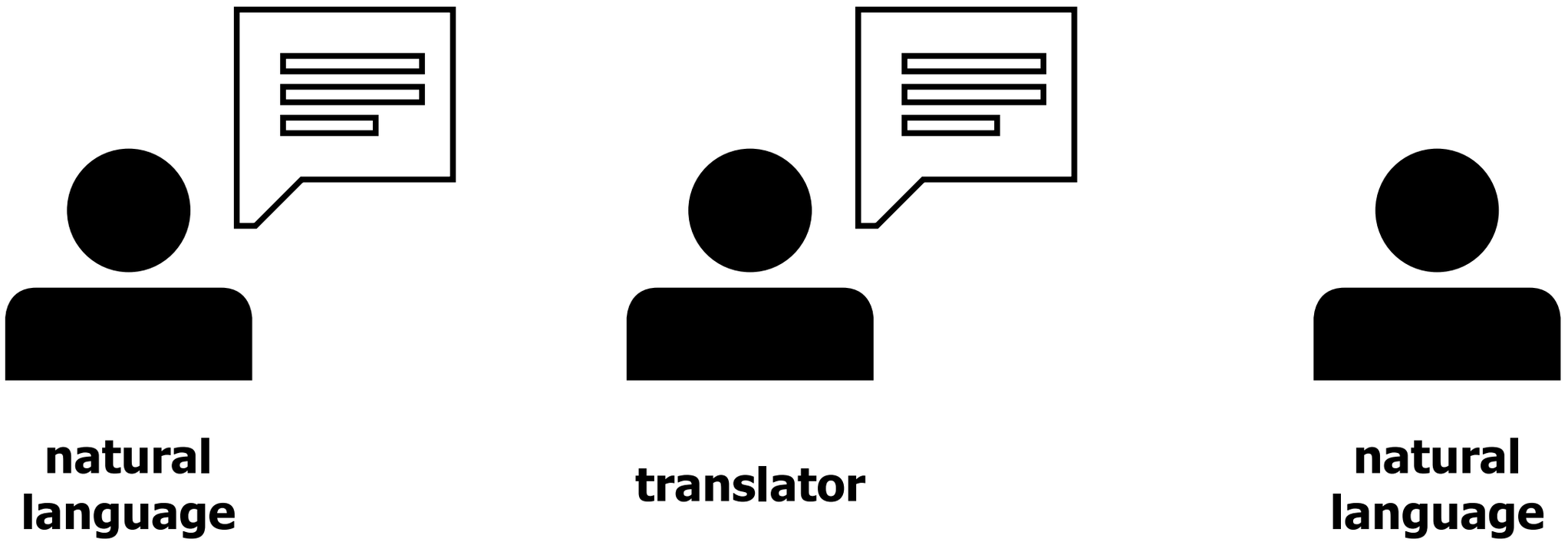
natural language



natural language

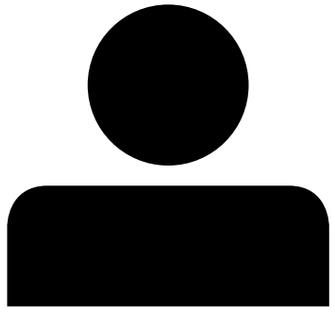
Communication

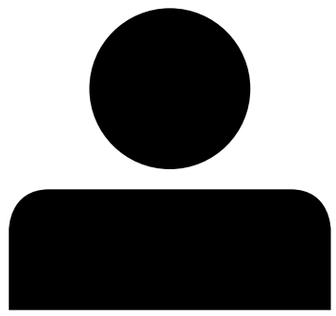
translation



Communication

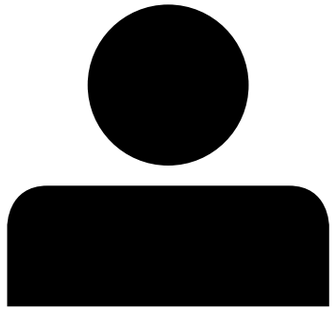
software language

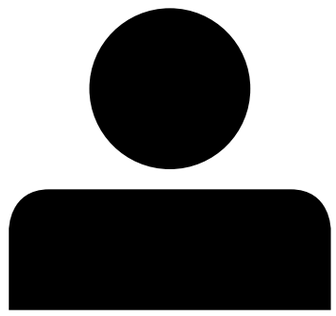




Communication

software language





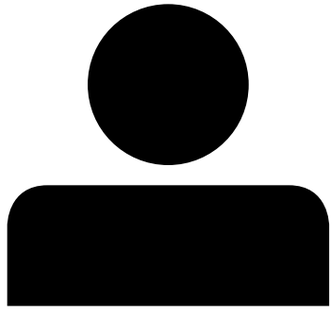
software language

Communication

software language

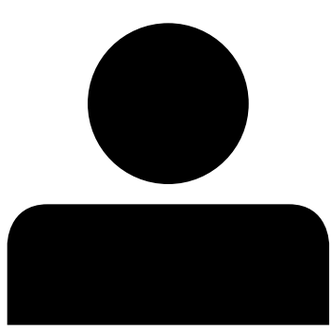
Communication

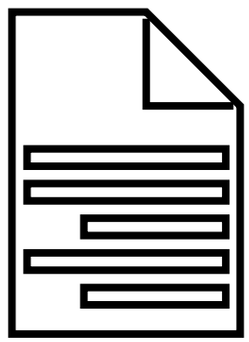
compilation

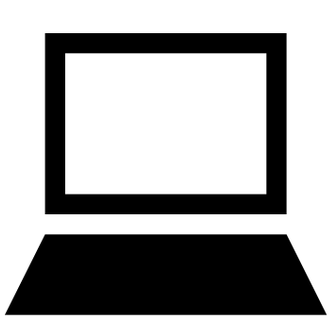


Communication

compilation



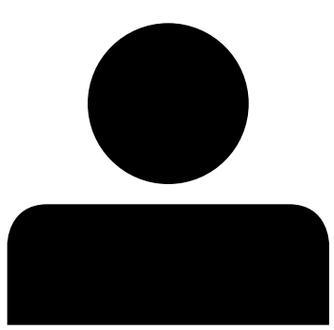


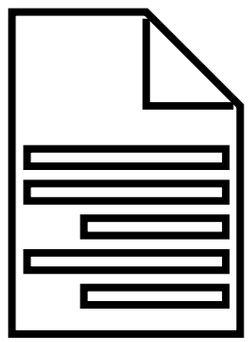


software language

Communication

compilation



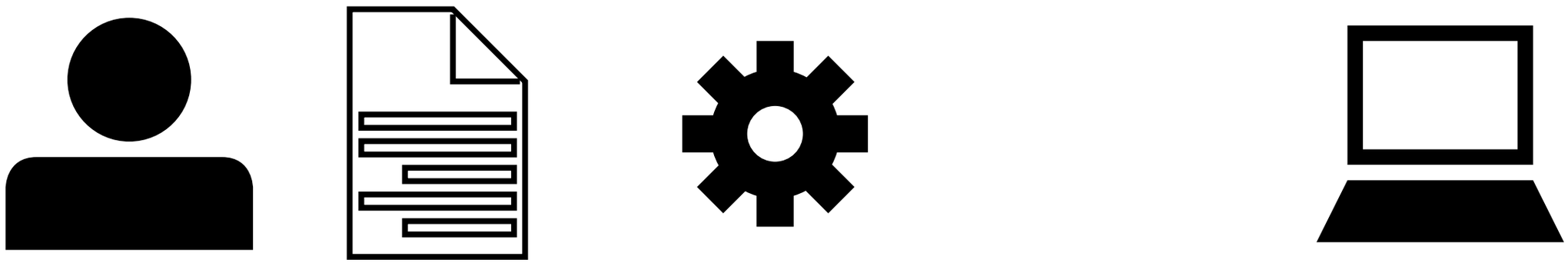


software language

machine language

Communication

compilation



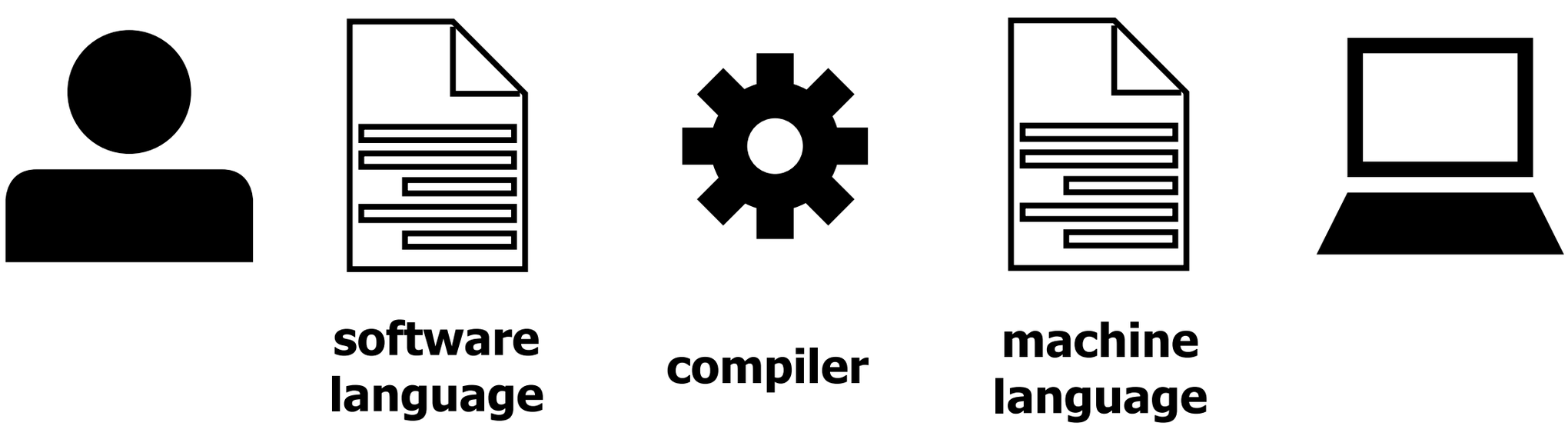
software language

compiler

machine language

Communication

compilation



Traditional Compilers

example

Traditional Compilers

example

ls

Course.java

Traditional Compilers

example

ls

Course.java

javac -verbose Course.java

[parsing started Course.java]
[parsing completed 8ms]
[loading java/lang/Object.class(java/lang:Object.class)]
[checking university.Course]
[wrote Course.class]
[total 411ms]

Traditional Compilers
example

ls

Course.java

javac -verbose Course.java

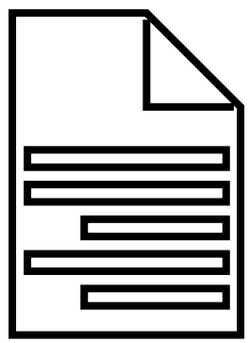
[parsing started Course.java]
[parsing completed 8ms]
[loading java/lang/Object.class(java/lang:Object.class)]
[checking university.Course]
[wrote Course.class]
[total 411ms]

ls

Course.class Course.java

Traditional Compilers

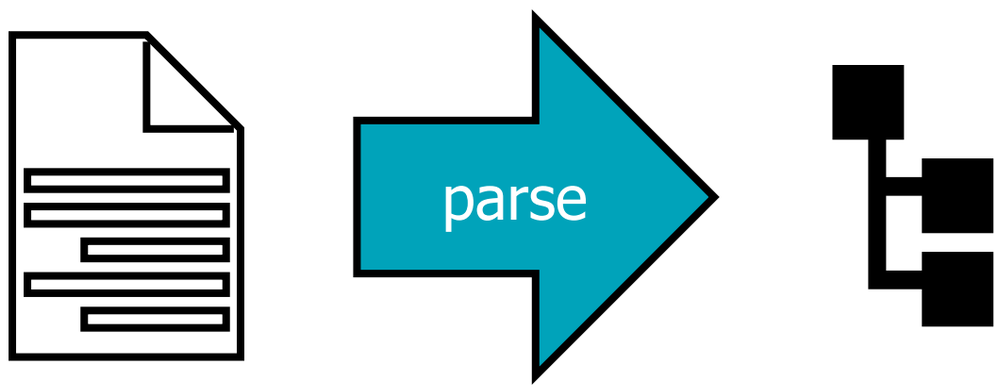
architecture



Source Code

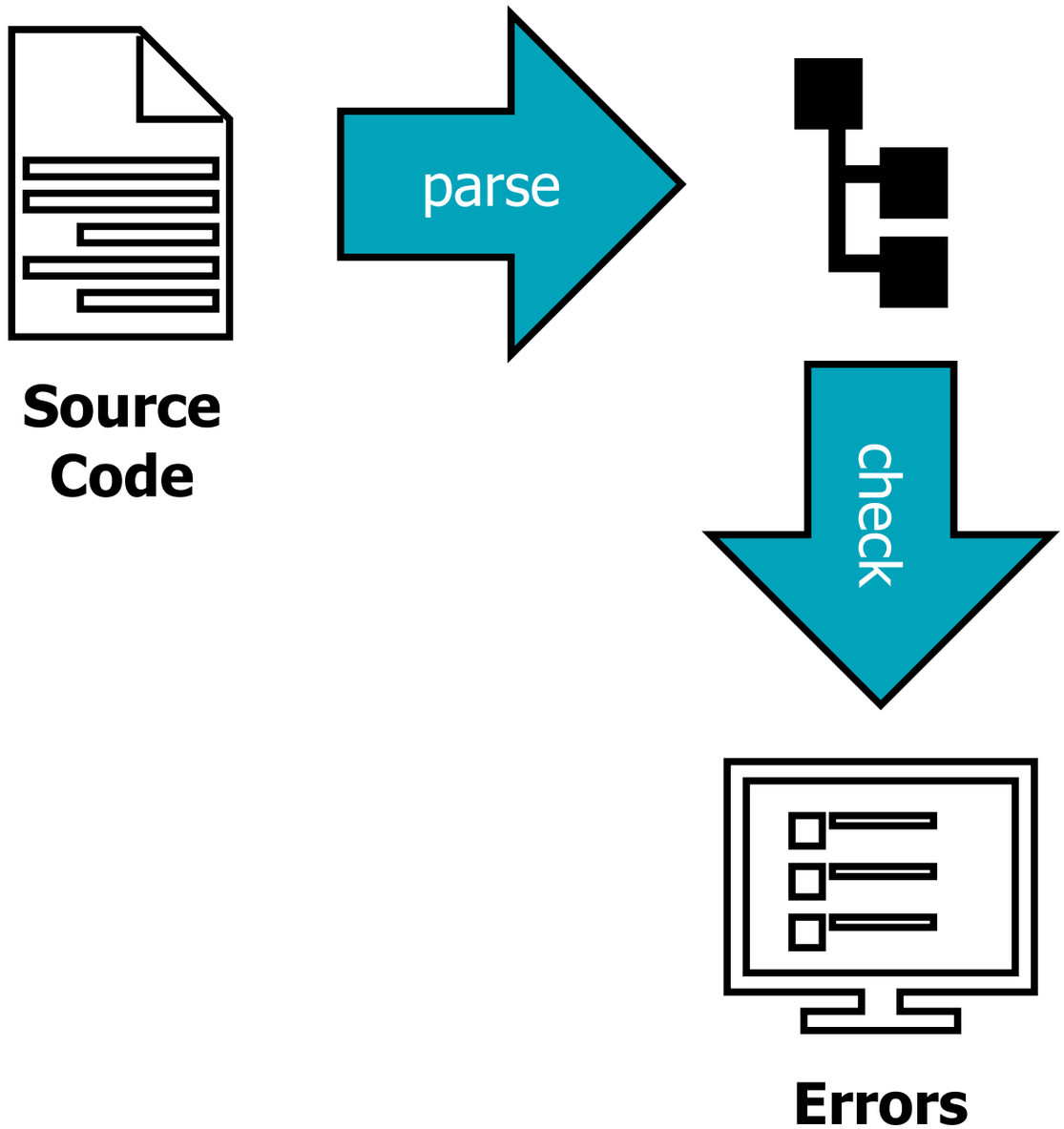
Traditional Compilers

architecture



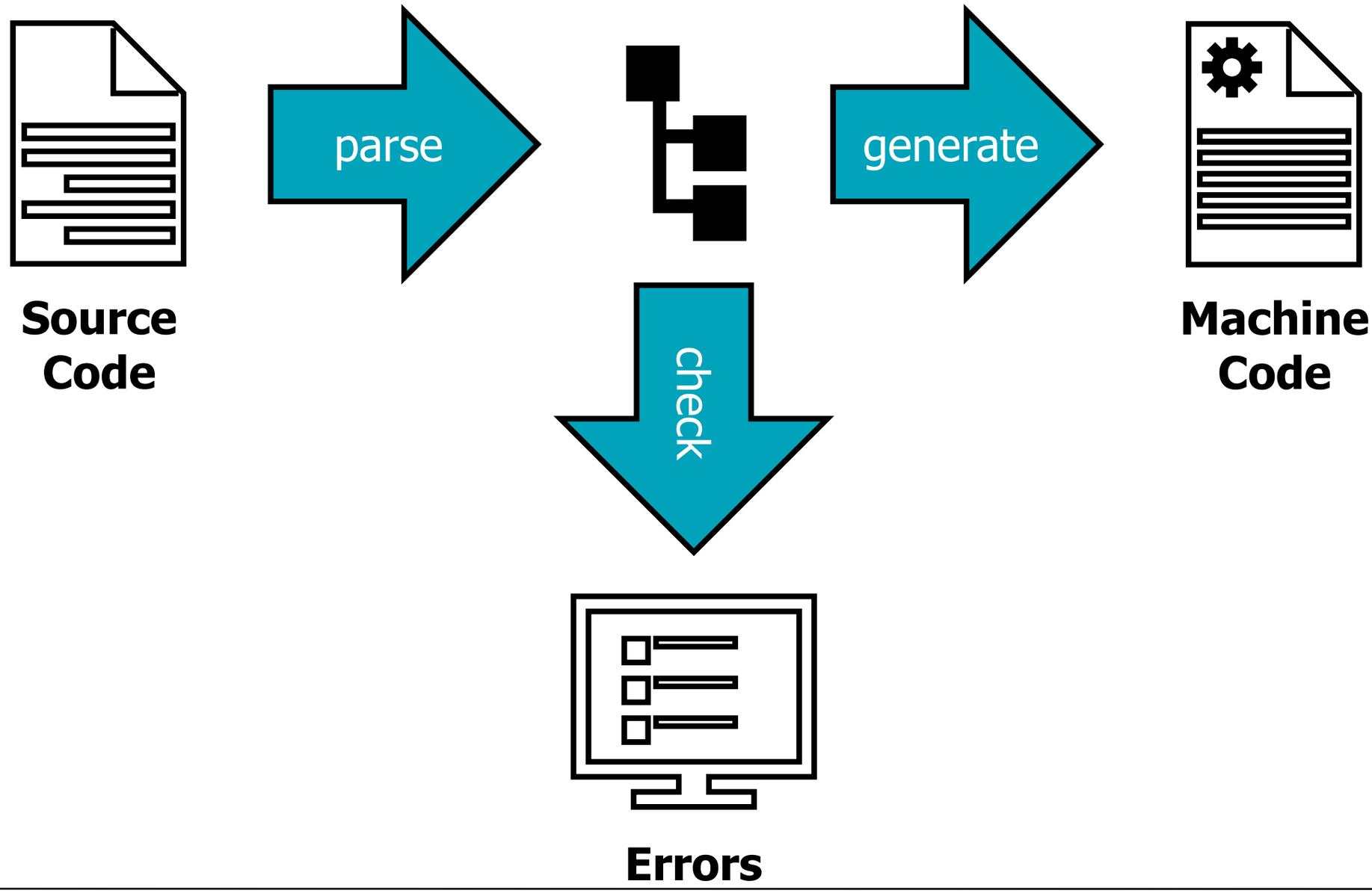
Traditional Compilers

architecture



Traditional Compilers

architecture



Modern Compilers in IDEs

example

Modern Compilers in IDEs

features

syntactic editor services

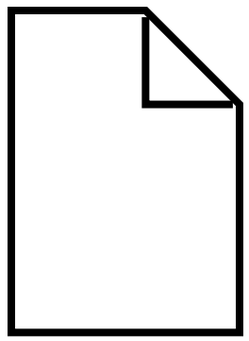
- syntax checking
- syntax highlighting
- outline view
- code folding
- bracket matching

semantic editor services

- error checking
- reference resolving
- hover help
- code completion
- refactoring

Modern Compilers in IDEs

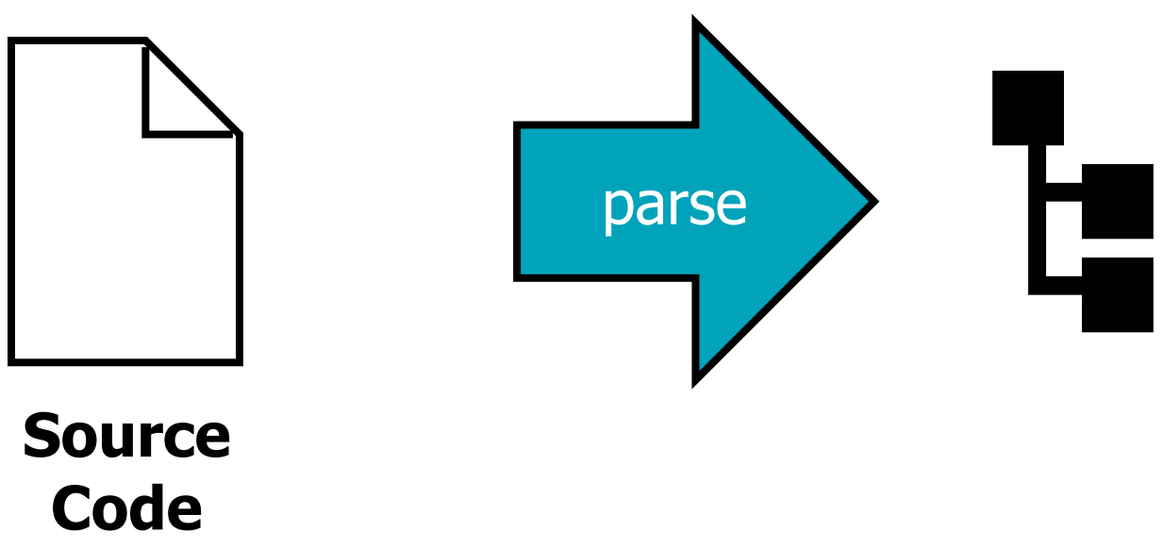
architecture



Source Code

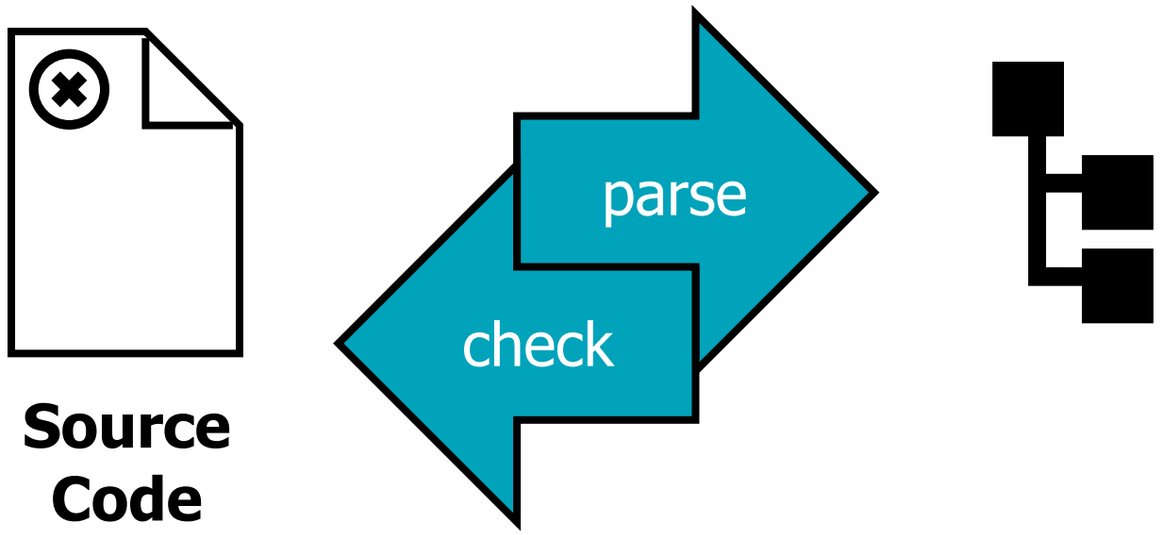
Modern Compilers in IDEs

architecture



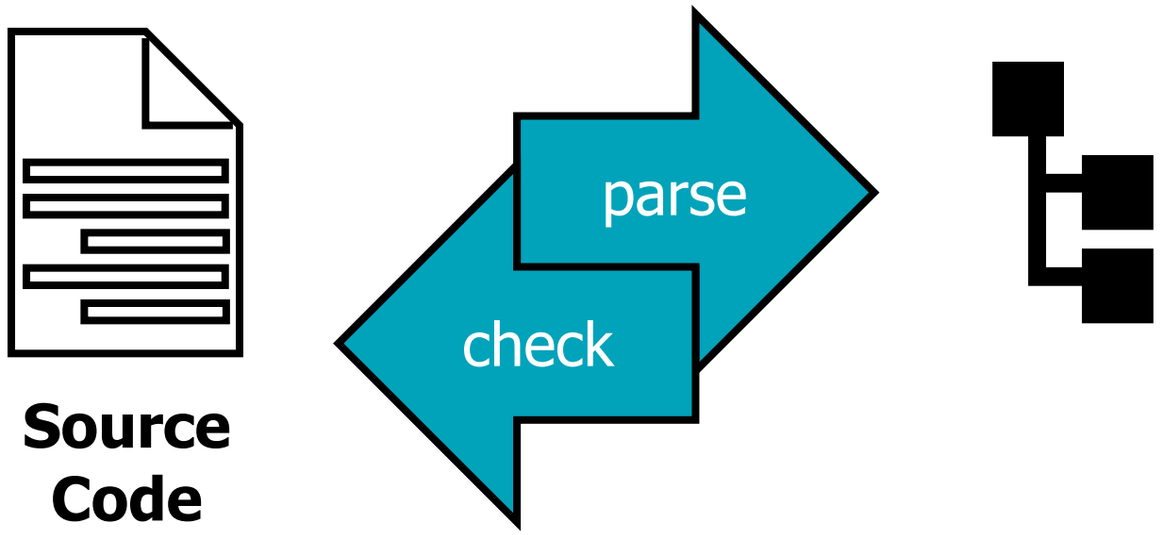
Modern Compilers in IDEs

architecture



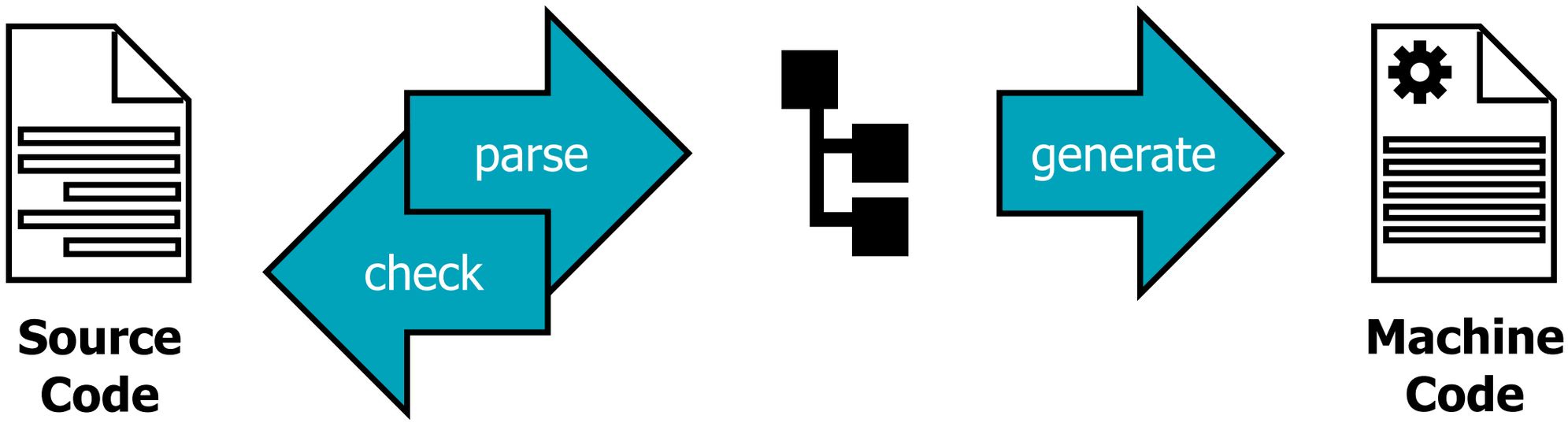
Modern Compilers in IDEs

architecture



Modern Compilers in IDEs

architecture



Language Processors
beyond compilation

syntax

- scanners
- parsers
- pretty-printers
- syntax-directed editors

semantics

- type checker
- analysis tools
- optimisers
- refactoring tools
- renovation tools
- interpreters
- code generators
- debuggers

Software Languages

languages to engineer **software**

pieces of **software** themselves

coffee break

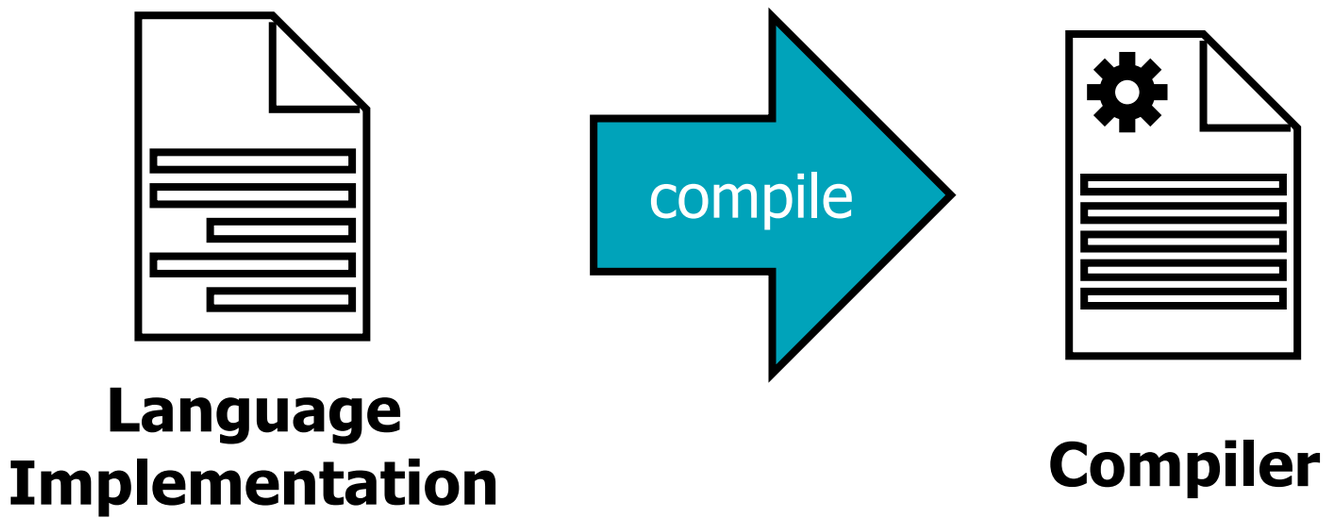
III

compiler construction

metalanguage facility

Traditional Compiler Compilers

manual implementation

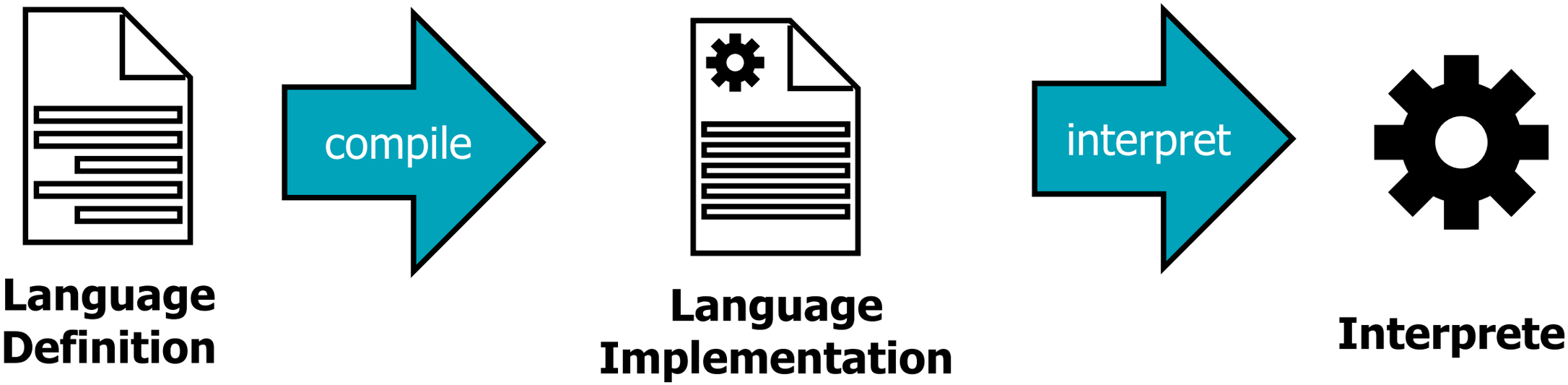


Traditional Compiler Compilers

compilation + compilation

Traditional Compiler Compilers

compilation + interpretation



Language Definition
components

syntax definition

- concrete syntax
- abstract syntax

static semantics

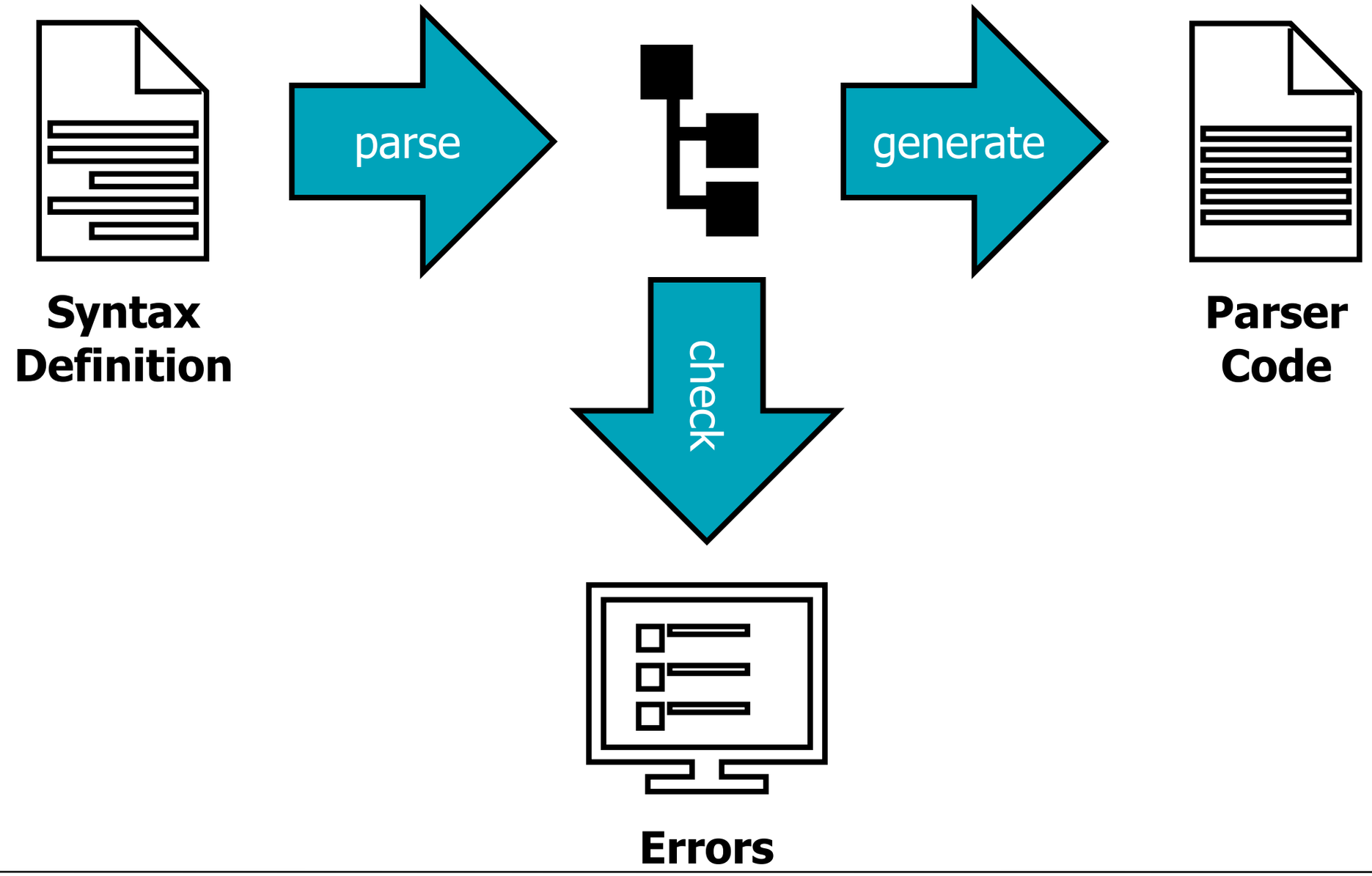
- name bindings
- type analysis

dynamic semantics

- translation
- interpretation

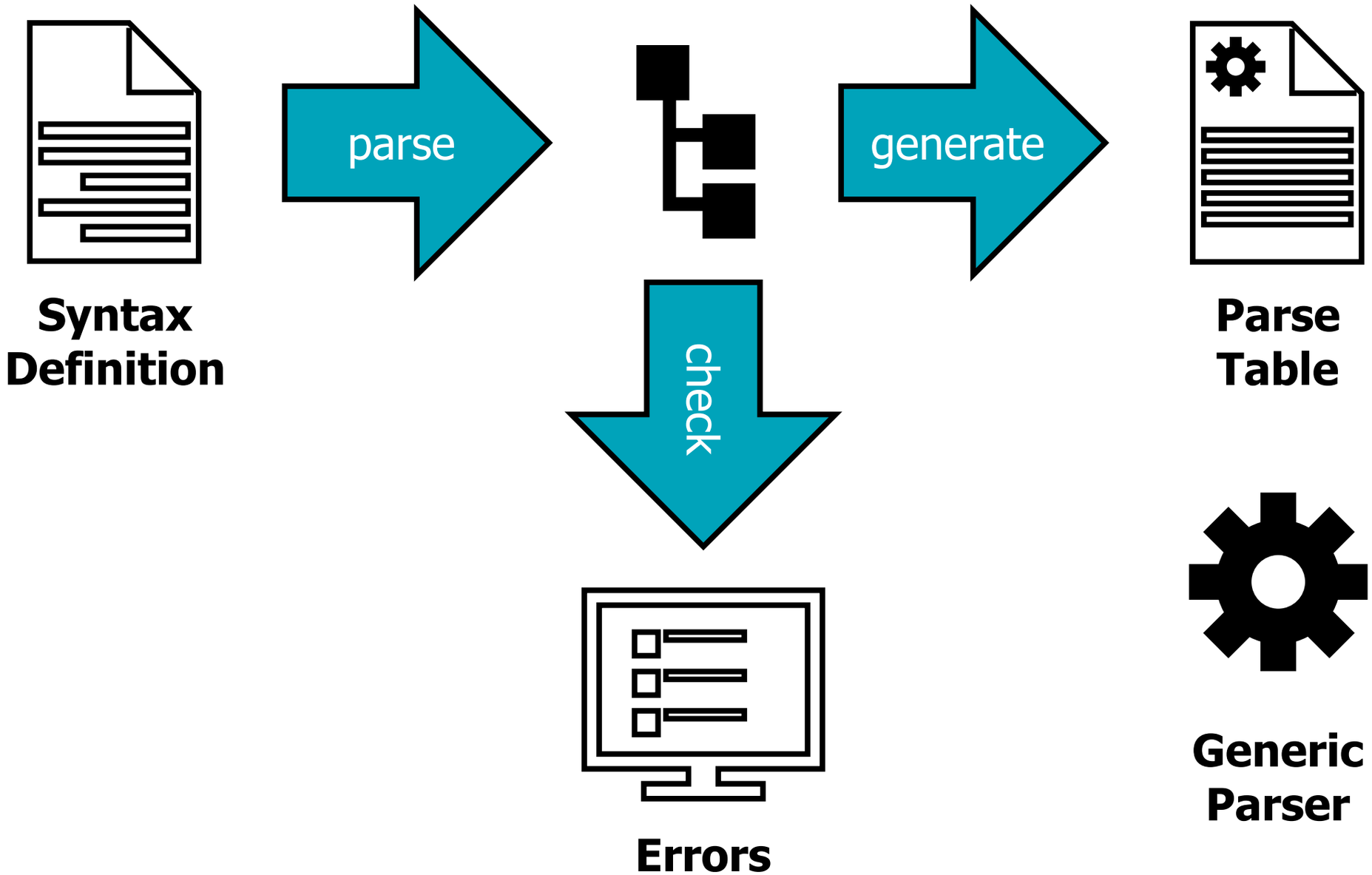
Traditional Compiler Compilers

compilation + compilation

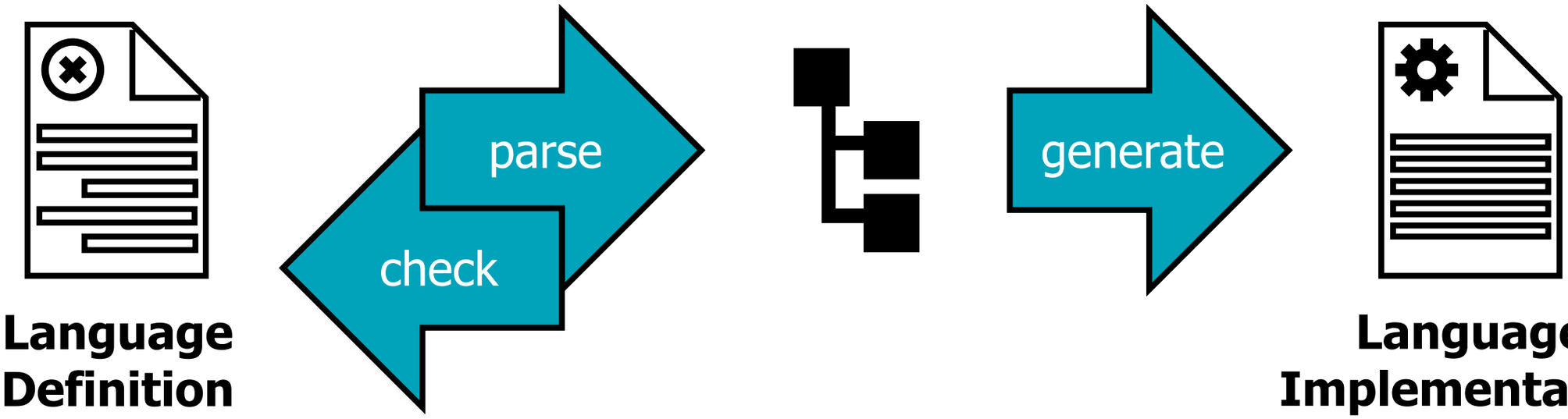


Traditional Compiler Compilers

compilation + interpretation



Language Workbenches
architecture



Spoofax

the lecture & lab language workbench

based on IMP framework by IBM Research

syntax definition

- declarative meta-languages
- modular syntax definitions

semantics

- declarative transformation languages
- operating on ASTs
- rewrite rules, strategies

editor services

- declarative configuration languages

IV

organisation

Organisation

overview

Blackboard

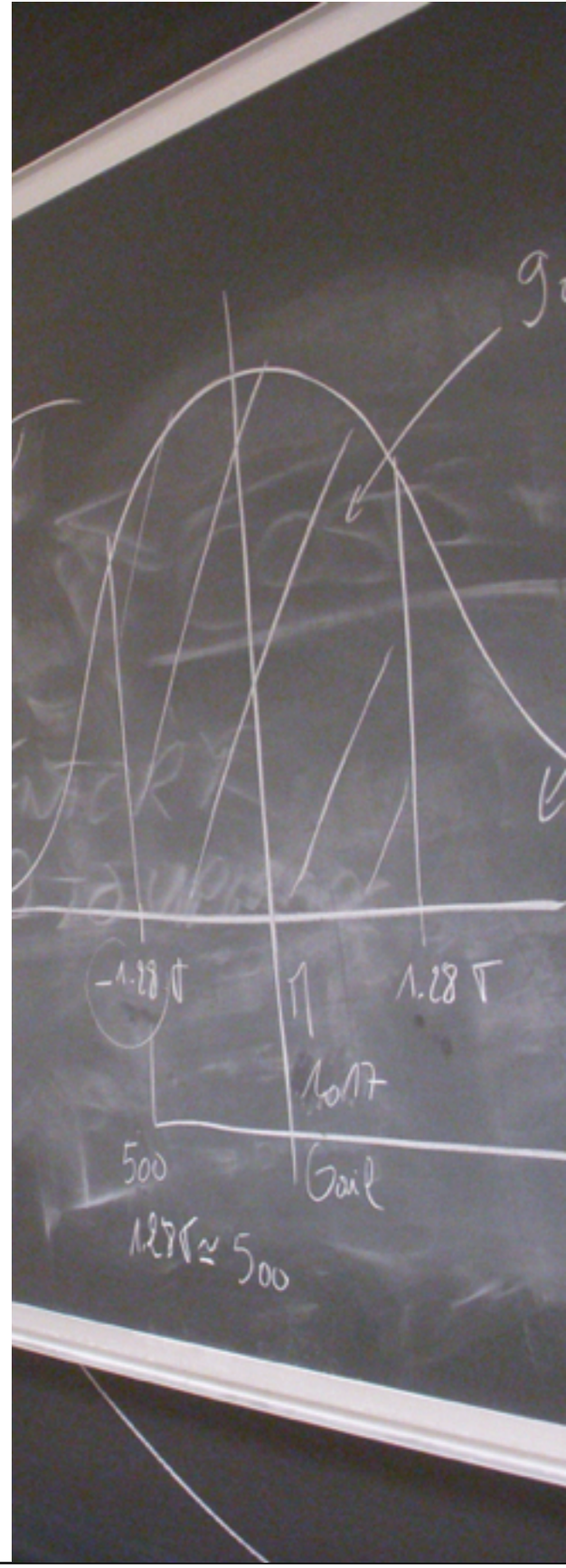
http://blackboard.tudelft.nl

course

- 15 lectures
- 14 lab sessions

assessment

- weekly lab assignments
- written exam Jan 28



Organisation

contact

lectures & lab sessions

office

HB 08.040, Mekelweg 4

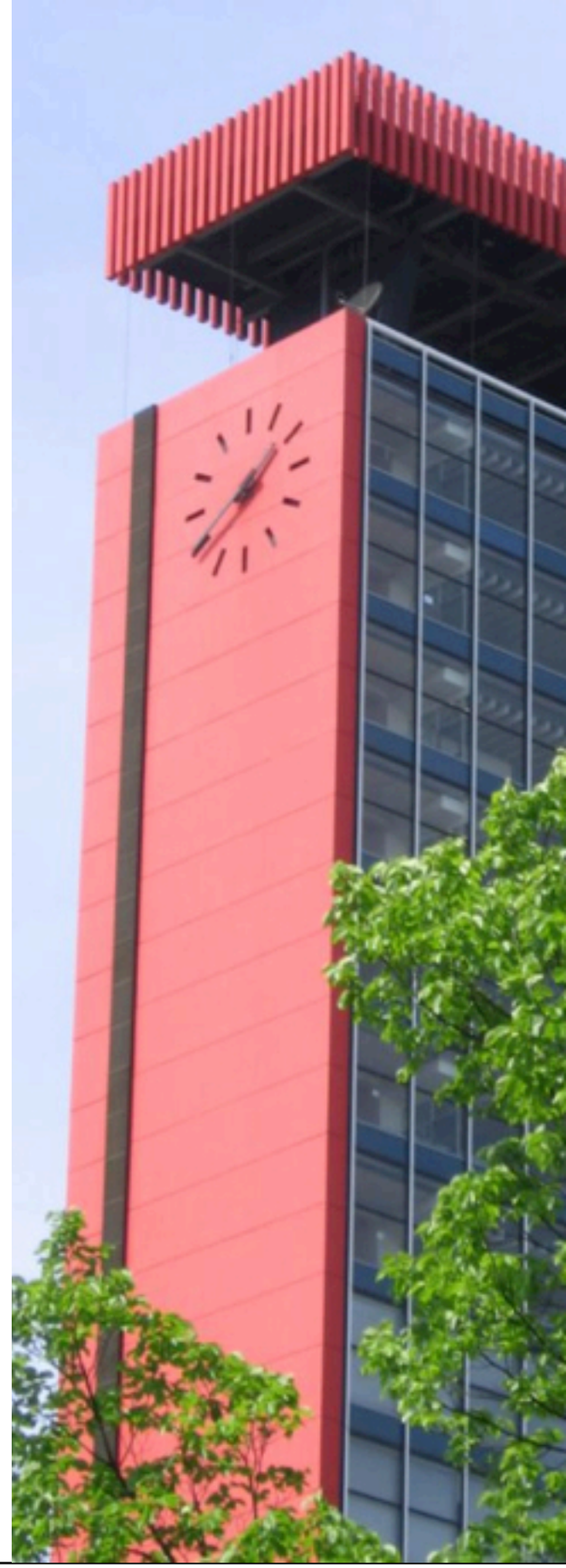
e-mail

G.H.Wachsmuth@tudelft.nl

twitter

http://twitter.com/IN4303

http://twitter.com/guwac



Organisation
lectures

lectures

- 1 + 8 + 5 + 1 lectures
- starting Sep 9

parts

- introduction (2 lectures)
- declarative language definition (6)
- implementing imperative and object-oriented languages (4)
- compiler components and their generators (3)

Organisation
assignments

practical work

- 1 + 6 + 7 lab sessions
- starting Sep 12

milestones

- editor (3 labs)
- name + type analysis + error checking (5)
- code generator (5)

Organisation

resources

text book

Andrew W. Appel, Jens Palsberg: Modern Compiler Implementation in Java, 2nd Edition. Cambridge University Press, 2002

papers

http://researchr.org/bibliography/compiler-construction

software

http://eclipse.org

http://spoofax.org

http://strategoxt.org

V

summary

Summary
lessons learned

Summary
lessons learned

What are software languages?

- computer-processable artificial languages used in software eng.
- pieces of software

Summary
lessons learned

What are software languages?

- computer-processable artificial languages used in software eng.
- pieces of software

What is a compiler?

- piece of software translating high-level code to low-level code
- traditional compilers vs. compilers in IDEs

Summary
lessons learned

What are software languages?

- computer-processable artificial languages used in software eng.
- pieces of software

What is a compiler?

- piece of software translating high-level code to low-level code
- traditional compilers vs. compilers in IDEs

How can we construct compilers?

- manual implementation vs. generation
- language workbenches

Summary
lessons learned

What are software languages?

- computer-processable artificial languages used in software eng.
- pieces of software

What is a compiler?

- piece of software translating high-level code to low-level code
- traditional compilers vs. compilers in IDEs

How can we construct compilers?

- manual implementation vs. generation
- language workbenches

How is this course organised?

Literature

learn more

Literature

learn more

language

Edward Sapir. Language. An Introduction to the Study of Speech. 1921

M. Paul Lewis, editor. Ethnologue: Languages of the World. 2009

Literature

learn more

language

Edward Sapir. Language. An Introduction to the Study of Speech. 1921

M. Paul Lewis, editor. Ethnologue: Languages of the World. 2009

MiniJava and Tiger

Andrew W. Appel, Jens Palsberg: Modern Compiler Implementation in Java. Cambridge University Press

Andrew W. Appel: Modern Compiler Implementation in ML.

Literature

learn more

language

Edward Sapir. Language. An Introduction to the Study of Speech. 1921

M. Paul Lewis, editor. Ethnologue: Languages of the World. 2009

MiniJava and Tiger

Andrew W. Appel, Jens Palsberg: Modern Compiler Implementation in Java. Cambridge University Press

Andrew W. Appel: Modern Compiler Implementation in ML.

Spoofax

Lennart C. L. Kats, Eelco Visser. The Spoofax Language Workbench. Rules for Declarative Specification of Languages and IDEs. OOPSLA 2010

Outlook
coming next

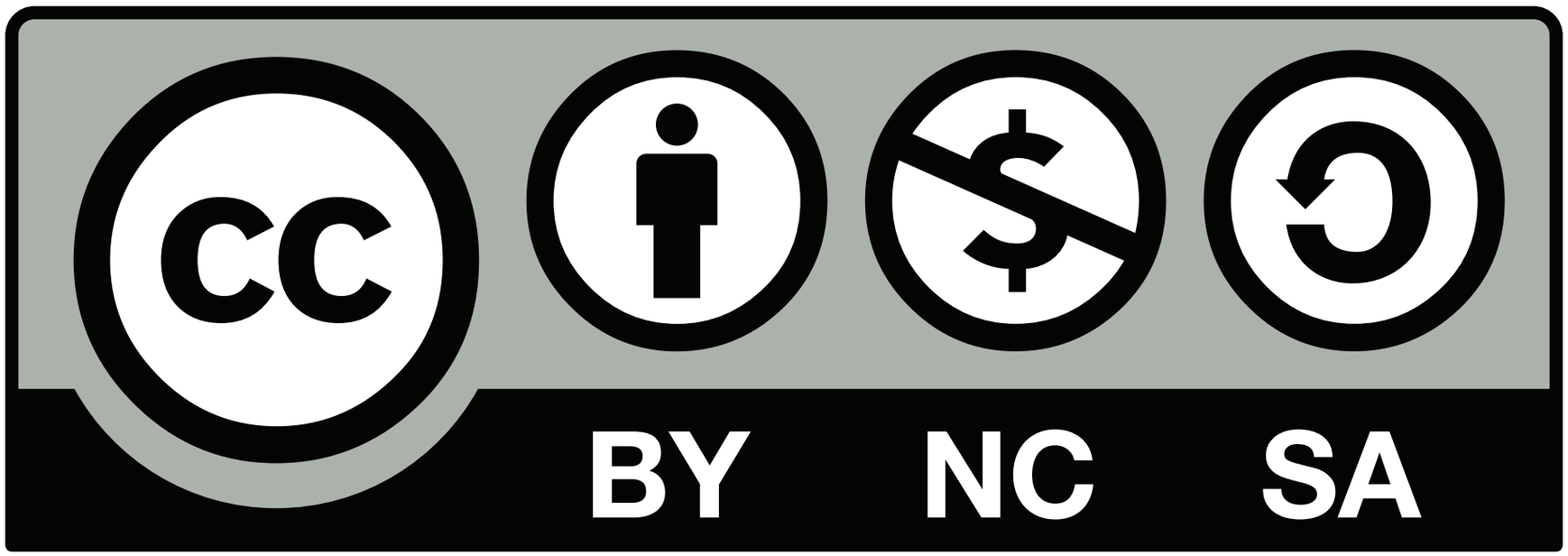
imperative & OO programming languages

declarative language definition

- Lecture 1: Grammars and Trees
- Lecture 2: SDF and ATerms
- Lecture 3: Term Rewriting
- Lecture 4: Static Analysis and Error Checking
- Lecture 5: Code Generation

Lab Sep 12

- get used to Eclipse, Spoofax, and MiniJava
- explore an Spoofax project


BY NC SA

Pictures
attribution & copyrights

Slide 1: Kröller-Müller Museum by Susanne Tofern, some rights reserved

Slide 4: Timeout ETB Wohnbau baskets by Michael Gohl, all rights reserved

Slides 5/12: Tower of Babel by Pieter Bruegel the Elder, public domain

Slide 7:
Fashionable melange of English words (1) by trialsanderrors, public domain
Fashionable melange of English words (2) by trialsanderrors, public domain
Latin Grammar by Anthony Nelzin, some rights reserved
Ginkgo Biloba by Johann Wolfgang von Goethe, public domain
Oxford English Dictionary by Camilla Hoel, some rights reserved
Sub-deb slang by genibee, some rights reserved
Wednesday by Michael Fawcett, some rights reserved

Slide 8/32: Thesaurus by Enoch Lau, some rights reserved

Slide 11:
The captured Swiftsure, Seven Oaks, Loyal George and Convertine brought through Goeree Gat by Willem van de Velde the Younger, public domain

Slide 13:
Duden 1891 by Merker Berlin, public domain
Elvish hymn by Sémhur, some rights reserved
UML Diagrams by Kishorekumar 62, some rights reserved

Pictures

attribution & copyrights

Slide 16: Programming language textbooks by K.lee, public domain

Slide 18: Tiger by Bernard Landgraf, some rights reserved

Slide 19: The C Programming Language by Bill Bradford, some rights reserved

Slide 20: Italian Java book cover by unknown artist, some rights reserved

Slides 23-26, 28, 31, 37-39, 41-43: PICOL icons by Melih Bilgil, some rights reserved

Slide 34: Tchibo by Dominica Williamson, some rights reserved

Slide 46: Blackboard by Wikistand, public domain

Slide 47: EWI building by Alper Çuğun, some rights reserved

Slides 48, 49: Students, TU Delft, Media Solutions, all rights reserved

Slide 54: The wanderer above the sea of fog by Caspar David Friedrich, public domain