IARIA
International Academy, Research, and Industry Association

SoftNet 2020
Porto
October 18 –22, 2020

CHALLENGES OF DISTANT LEARNING IMPLEMENTATION OF A PROGRAMMING CLASS

IARIA SOFTNET 2020 KEYNOTE

Evgeny Pyshkin

University of Aizu

Abstract

In the situation of societal lockdowns of 2020, educational institutions faced new challenges in organizing teaching and learning processes so that to adopt them to extensive use of remote teaching models. Currently we observe an increased academic discourse on efficient distant learning approaches. Though various practical solutions, successful practices, and supporting computer technology seem to be already in place, many additional efforts are required from academicians and tutors to resolve significant technical, managerial, methodological, and psychological issues of distant learning.

In this talk, we discuss our approach to programming class organization and workflow with a particular focus on its adoption to current situation requiring extensive (and even exclusive) use of distant learning tools, which have both great advantages and considerable limitations. We share a number of methodological and organizational solutions that may be used to improve software development instruction, where the suggested methods are not only focused on remote teaching tools, but may serve traditional face-to-face classes as well.

We particularly address such aspects as forms of teacher/learner collaboration and the project roles that teachers and students can perform during class sessions, interactivity issues, incremental design applied to the classroom needs, importance of visual models, integration of academic workflow with software testing, project management, code review and source control tools.

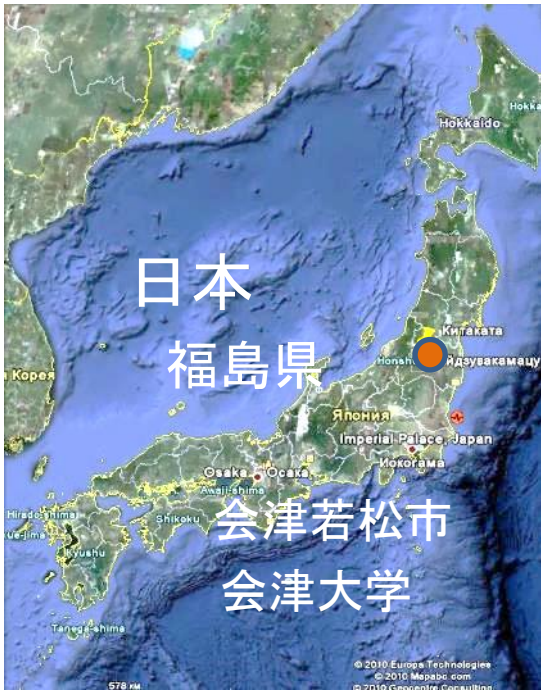
Speaker

Senior Associate Professor, Ph.D., Doctoral Maru-Go ***University of Aizu***

Career: Software Engineer; Assistant Professor; Associate Professor;
Senior Associate Professor ***Peter the Great St. Petersburg Polytechnic University***

https://u-aizu.ac.jp/~pyshe/

- Aizu-Wakamatsu – city in Tohoku Region, Japan
  - Population around 150000
  - Rice growing and sake production
  - Samurai city
  - University of Aizu – international university (40% of staff are foreigners)
  - University focused on computer technology and its applications



University of Aizu:
To Advanced Knowledge for Humanity

- Since 1993: First contemporary university in the region
- International Outlook **70.6** (**1st** in Japan)
- Citations **58.6** (**8th** in Japan)
- The only university in Japan offering bilingual programs in the undergraduate school.
- Graduate school is completely in English

THE WORLD UNIVERSITY RANKINGS

2021

601–800th	24th	201–250th
World University Rankings 2021	Japan University Rankings 2020	Young University Rankings 2020

https://u-aizu.ac.jp/

University of Aizu 会津大学
Japan National Ranking by Nikkei

主要101大学 5指標で比較する
教育・研究力ランキング

本誌では3回目となる、偏差値や人気に左右されない大学ランキング。今回は国際性をより重く見るため、各大学の「留学生受け入れ数」を指標の一つに加えた。ランキング対象と したのは、教員や高校生に人気のある101大学。この5指標で上位に入るのはどの大学か。自分が意外だと感じた大学の背景を探れば、大学を見る目がさらに養われるはずだ。

1~10位

順位	都道府県	設置	大学名	生徒に人気	教員が慕める	㋐学部数 ㋑学部学生数	ST比	㋐科研費配分額 ㋑海外論文数	論文のFWCI	国際共著論文率	留学生数	400社就職率(19年)	本誌評価値
1位	東京都	国	東京大学	9位	1位	10学部 1万4058人	7.3	166億8309万円 5万7261本	1.35	34.5%	4267人	—	393.7点
2位	京都府	国	京都大学	8位	2位	10学部 1万2992人	8.5	105億5475万円 3万9178本	1.35	32.7%	2732人	—	350.5点
3位	東京都	私	早稲田大学	1位	4位	13学部 4万267人	36.4	21億421万円 1万1378本	1.14	31.5%	7942人	37.2%	327.4点
4位	宮城県	国	東北大学	7位	5位	10学部 1万881人	6.0	75億805万円 3万455本	1.12	32.4%	2162人	29.9%	323.2点
5位	大阪府	国	大阪大学	16位	6位	11学部 1万5285人	9.6	81億3524万円 3万1697本	1.12	28.7%	2480人	35.6%	320.8点
6位	愛知県	国	名古屋大学	18位	17位	9学部 9724人	9.2	58億6056万円 2万3110本	1.22	30.7%	2462人	33.7%	318.5点
7位	福岡県	国	九州大学	11位	14位	12学部 1万1647人	9.1	57億7413万円 2万3664本	1.13	31.6%	2313人	26.7%	314.0点
8位	茨城県	国	筑波大学	24位	18位	9学群 9909人	9.3	32億3320万円 1万5867本	1.18	31.8%	2673人	18.5%	308.6点
9位	東京都	国	東京工業大学	33位	12位	6学院 4828人	9.8	37億3533万円 1万8741本	1.17	32.4%	1664人	57.4%	301.9点
10位	東京都	公	首都大学東京	56位	47位	7学部 6882人	13.5	8億2880万円 5674本	1.51	35.6%	656人	21.7%	297.2点

Advancing Schools 2019

11~25位 学生数約1000人の会津大学が北大に迫る

順位	都道府県	設置	大学名	生徒に人気	教員が慕める	㋐学部数 ㋑学部学生数	ST比	㋐科研費配分額 ㋑海外論文数	論文のFWCI	国際共著論文率	留学生数	400社就職率(19年)	本誌評価値
11位	北海道	国	北海道大学	23位	13位	12学部 1万1311人	8.8	46億5690万円 2万498本	1.01	30.5%	1570人	24.8%	295.8点
12位	福島県	公	会津大学	199位	53位	1学部 1036人	11.3	2920万円 1334本	1.13	49.2%	11人	10.3%	289.1点
13位	広島県	国	広島大学	30位	22位	12学部 1万695人	9.7	20億429万円 1万2295本	1.04	29.1%	1883人	16.3%	285.9点
14位	兵庫県	国	神戸大学	12位	21位	12学部 1万1596人	13.0	24億1100万円 1万1759本	1.18	26.4%	1399人	30.7%	284.0点
15位	千葉県	国	千葉大学	21位	25位	10学部 1万547人	10.7	17億3412万円 1万162本	1.05	26.1%	1861人	16.4%	280.5点
16位	長野県	国	信州大学	46位	36位	9学部 9077人	12.7	8億4010万円 6105本	1.25	33.4%	351人	14.1%	280.2点
17位	岡山県	国	岡山大学	28位	38位	11学部 1万157人	12.5	16億3017万円 9584本	1.15	29.4%	710人	15.2%	277.0点
18位	秋田県	公	国際教養大学	93位	8位	1学部 871人	14.0	1400万円 109本	0.79	50.5%	151人	44.5%	274.2点
19位	新潟県	国	新潟大学	58位	46位	10学部 1万255人	11.8	13億3040万円 6309本	1.16	26.6%	545人	10.6%	271.9点
20位	東京都	私	慶応義塾大学	3位	3位	10学部 2万8643人	25.1	24億9860万円 1万4988本	1.09	22.0%	2013人	—	270.1点
21位	大分県	私	立命館アジア太平洋大学	160位	23位	2学部 5481人	34.3	3490万円 285本	0.53	46.7%	2906人	10.4%	265.9点
22位	石川県	国	金沢大学	36位	31位	3学域 7862人	11.4	16億9630万円 7196本	1.00	24.5%	632人	13.7%	264.6点
23位	長崎県	国	長崎大学	47位	133位	10学部 7504人	10.3	10億7460万円 6239本	0.94	28.6%	549人	13.3%	264.0点
24位	東京都	私	立教大学	5位	20位	10学部 1万9380人	45.6	3億2700万円 1169本	1.21	39.5%	850人	24.9%	262.2点
25位	熊本県	国	熊本大学	49位	53位	7学部 7757人	12.4	13億3653万円 6451本	0.95	27.0%	522人	14.3%	261.7点

Charming Aizu








Speaker's Professional Background

- Human-centric computing
  - Rapid transformation from HCI to its own distinctive research agenda
  - Multiple links to digital transformation concepts
  - Affecting society and individuals
  - Inter- and even transdisciplinary nature of HCC
  - Towards more personalization and user collaboration

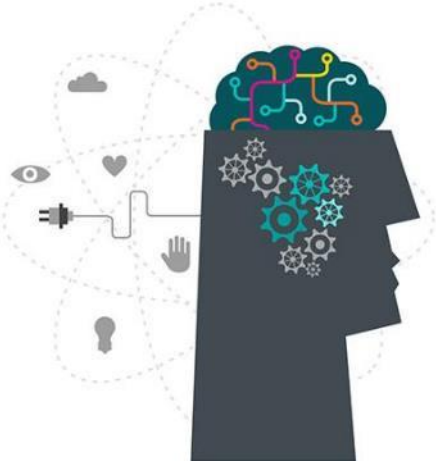
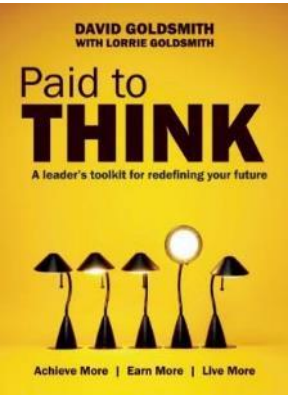


Image: http://icc.mtu.edu/hcc/

- Software Engineering
  - Methodology for software development education
  - Software testing
  - Software for computer-assisted language teaching
- Cross-cultural communication, technology disciplines in scope of humane sciences

Focus of this Talk

- Review of earlier works on programming teaching
  - Multi-aspect tasks in software education
  - Interdisciplinary aspects of software engineering
  - From engineering to liberal arts
- Programming class workflow
  - Why lecture and exercises are not enough
  - Network of connected activities and links to distant learning



- Affordances and constraints of distant learning
- Teaching and learning practices
  - Developing students' abilities to think effectively
  - Developing practical and soft skills

perhaps a primary goal of any kind of good education ☺

* David Goldsmith. 2012. Paid to Think: A Leader's Toolkit for Redefining Your Future. BenBella Books, Inc.

From Engineering to Liberal Arts: Revisiting a Case of Software Engineering Education *

- Considering software disciplines within the context of liberal arts is connected to significant changes in the learning models
- We anticipate more than only professional developers' skills from our students
  - They have to be able to work in a **collaborative environment**
  - Significance of organizational learning models favoring **public display, teamwork and professional discussion** significantly increases
- It is extremely important to find ways to create a collaboration environment where students can actively participate in the **co-learning** process together with their more experiences colleagues

Computer Science

- Arithmetic
- Geometry
- Astronomy
- Logic
- Grammar
- Rhetoric
- Music

*"Computer science draws upon perspectives from many disciplines and has a symbiotic relationship with the liberal arts disciplines, so it might be considered the ultimate of them" ***

* E. Pyshkin, "Liberal arts in a digitally transformed world: A case of software development education," CEE-SECR '17, https://doi.org/10.1145/3166094.3166117.

** H.M. Walker and C. Kelemen, "Computer science and the liberal arts: a philosophical examination," ACM Transactions on Computing Education (TOCE), Mar 1, 2010, vol. 10, no. 1, pp.2:1–2:10.

Bridge a Methodology Gap in Software Education

Attention to important particularities of software development process with respect to a software development course

Software changeability

- Much different from products of engineering

Software as a community product

- Contributing to open-source solutions requires specific skills and abilities

Many interdisciplinary activities

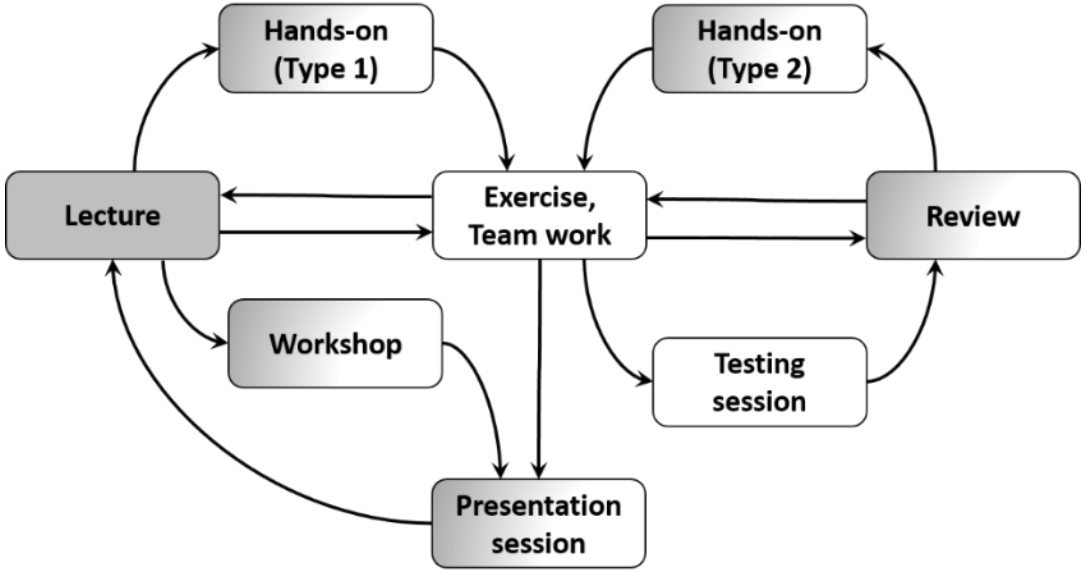
- Students have to get programming skills, but also to learn how to communicate with stakeholders, and how to cooperate in multidisciplinary teams

Programming is close to language study

- A software problem may have a variety of acceptable solutions

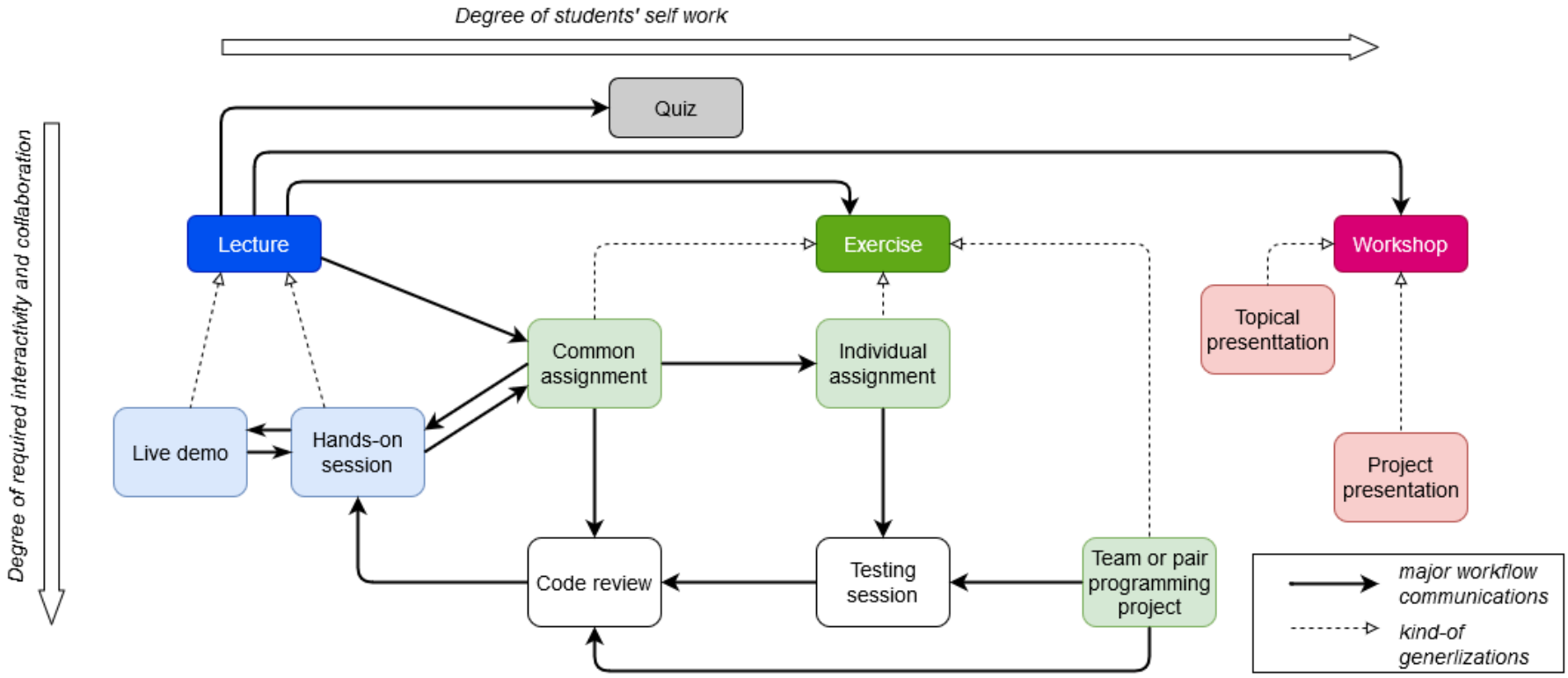
Use Case:
Learning Activities in a Programming Course

- Practices and lessons learned after teaching the connected undergraduate courses “Introduction to Programming” and “C Programming” in the University of Aizu (128 class hours in total)



- Diversity of activity forms
  - How computer science can learn from teaching forms and practices which exists in fine arts

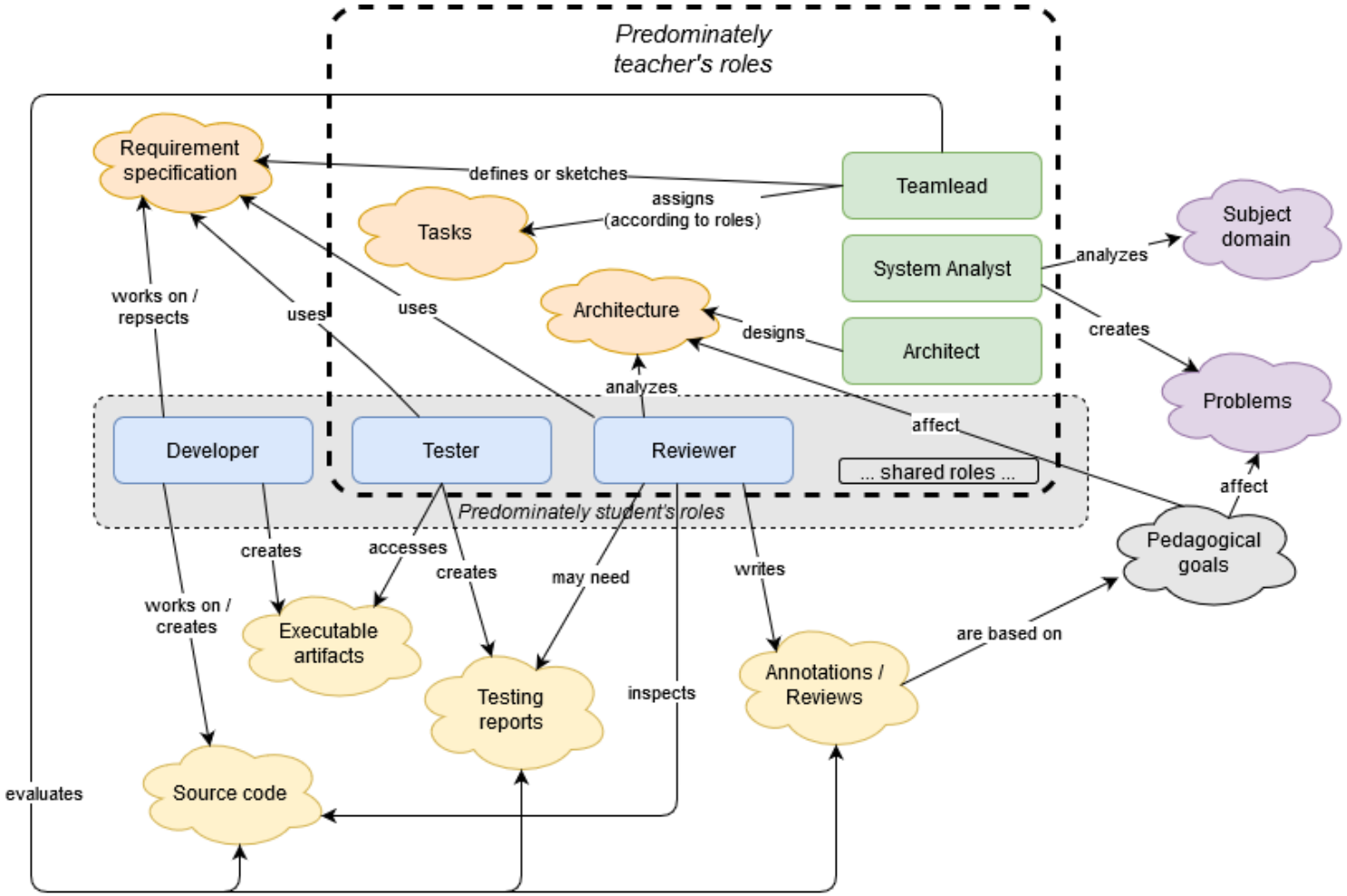
Refining Our Approach to Programming Class Organization and Workflow * **



* E. Pyshkin, "On Programming Classes under Constraints of Distant Learning," 2020 The 4th International Conference on Software and e-Business (ICSEB-2020), Dec 18-20, 2020, Osaka, Japan. To appear.

** M. Mozgovoy and E. Pyshkin, "Plagiarism Detection Systems for Programming Assignments: Practical Considerations," The 15th International Conference on Software Engineering Advances (ICSEA 2019), Oct 18-22, Porto, Portugal, IARIA, 2020. To appear.

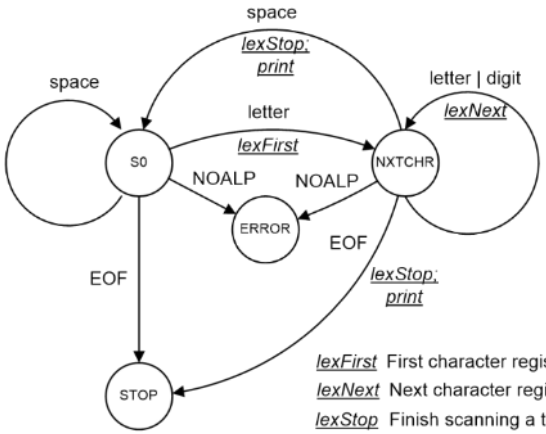
Roles that Teachers and Students Perform *

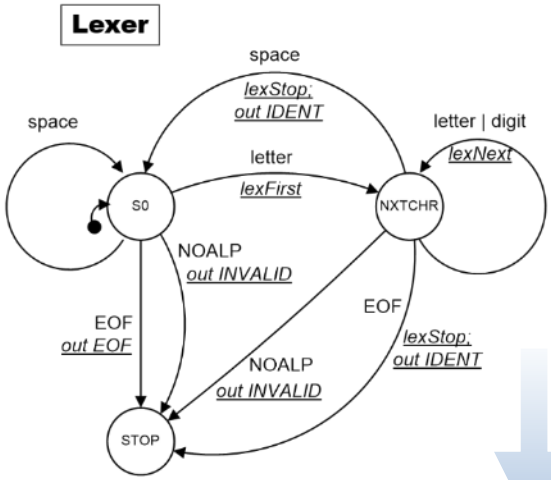


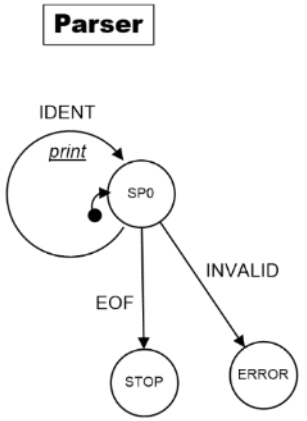
* E. Pyshkin, “On Programming Classes under Constraints of Distant Learning,” 2020 The 4th International Conference on Software and e-Business (ICSEB-2020), Dec 18-20, 2020, Osaka, Japan. To appear.

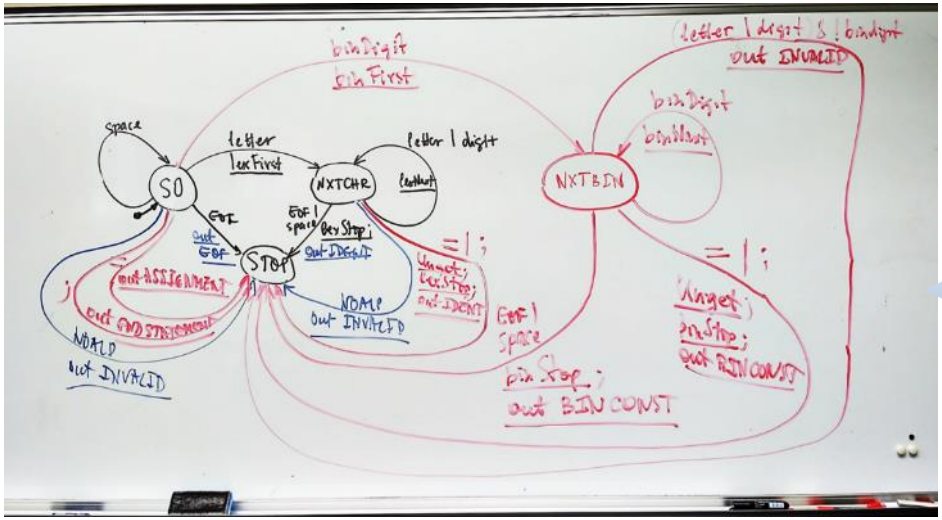
** Martin Cortazzi and Lixian Jin. 1999. Bridges to learning: Metaphors of teaching, learning and language. Researching and applying metaphor 149 (1999), 176.

Use Case: Hands-On Session Example









Exercise: Let's Extend Our Model

- Now our language supports three types of tokens:
  - Identifiers
  - Assignment =
  - End-of-expression ;
  - Binary values 100011101

  Lexer's responsibility
- Input file contains expressions like the following:
  - value = 100011101;

  Parser's responsibility
- The problem
  - Revise the state diagrams
  - Implement the revised finite state machines
    - Lexer recognizes tokens
    - Parser follows syntax rules
  - In process of program execution all the binary values should be printed as hexadecimal values

Bridging to More Complex Concepts: Multi-Aspect Tasks *

Example:

The syntax of a parenthesis-free expression may be defined by a context-free grammar $E = (V_N, V_T, P, S)$ where V_N is a finite set of nonterminal symbols, V_T is a finite set of terminal symbols, S is a start symbol, and P is a finite set of the grammar production rules.

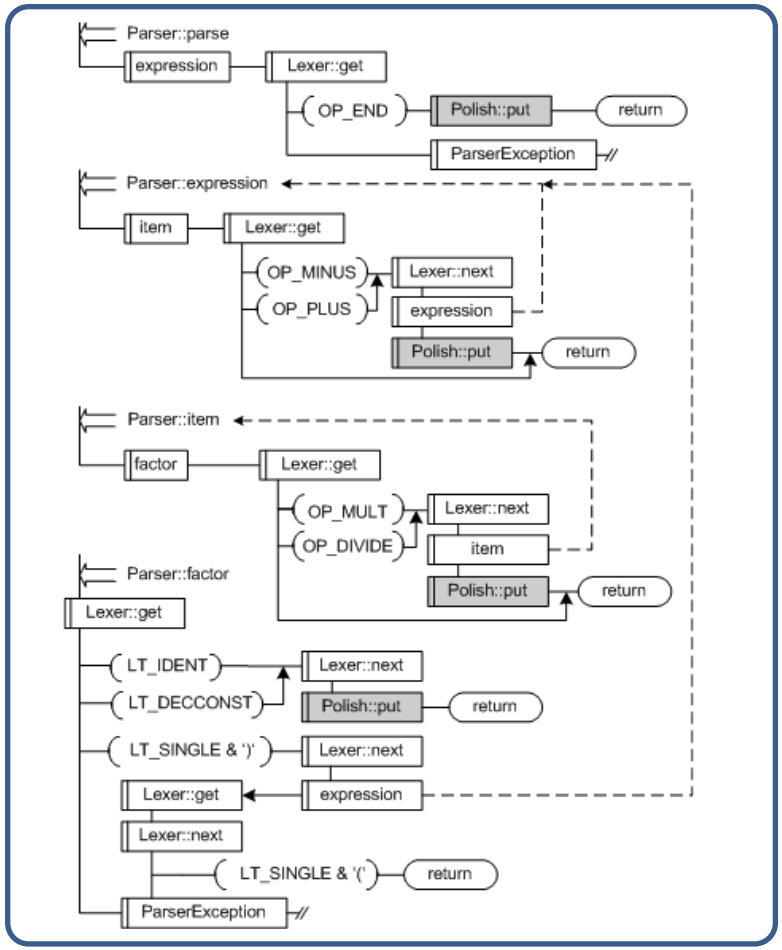
The first group of rules is to be used during the syntactic analysis stage

```
S ::= <expression> ";"
<expression> ::= <item>
<expression> ::= <item>{"+"|"-"}<expression>
<item> ::= <factor>
<item> ::= <factor>{"*"|"/"}<item>
<factor> ::= <identifier>
<factor> ::= <dec-const>
<factor> ::= "("<expression>")"
```

The second group is to be used by a lexer:

```
<identifier> ::=   <letter>[<letter>|<dec-digit>]...
<dec-const> ::= [<dec-digit>]...
<letter> ::=   {'_'|'A'|'B'|...|'Z'|'a'|'b'|...|'z'}
<dec-digit> ::= {'0'|'1'|'2'|'3'|'4'|'5'|'6'|'7'|'8'|'9'}
```

Visualizing the solution in the form of L-Net suggested by M. Lekarev**



* E. Pyshkin, "Multi-Aspect Tasks in Software Education: a Case of a Recursive Parser," IJACSIT, 3(3), 2014, 282–305.

** M. Lekarev, "Das graphische Verfahren der Software-Entwicklung für logisch komplizierte Anwendungen," In Fachhoshschule Hamburg Tech. Bericht., 1993.

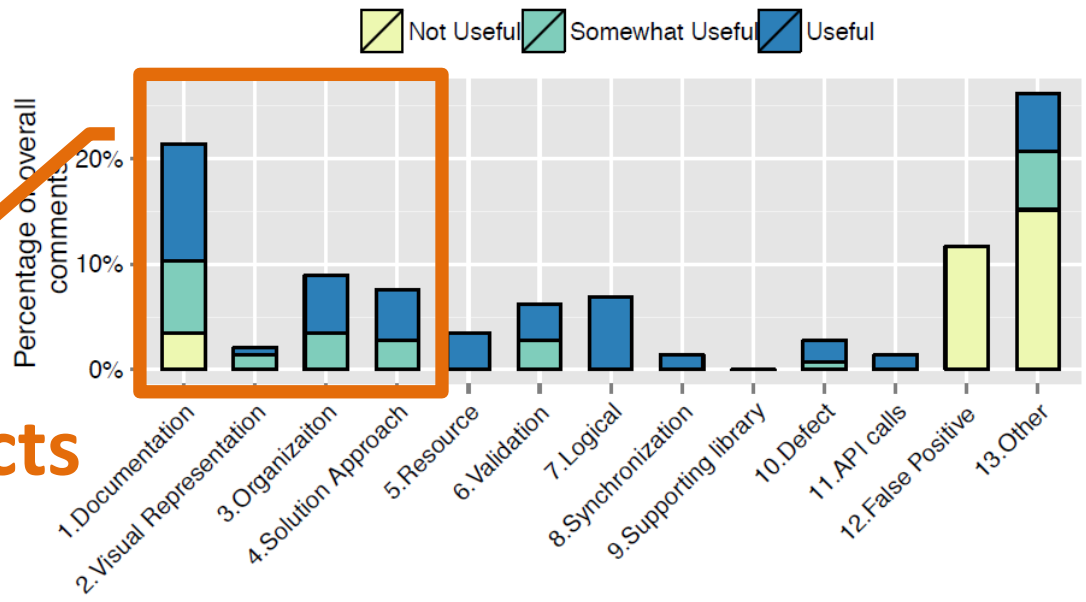
*** I. Sommerville, Software Engineering, 9th ed. Addison-Wesley, 2010.

Case Study: Parenthesis-free Expression Parsing as a Multi-Aspect Task

Solution stage	*Underlying models*	*Programming language focus*	*Visual models*	*In-depth focus*
Requirement definition	Recursive grammar	Unit tests planning	Syntactic diagrams	Left- and right- recursive grammars, grammar transformation
Symbol syntactic class definition	Sets, bitwise operations	Enumerations, symbol tables	Masked bitwise operations	Syntactic classes recognition in more complex languages
Lexical analysis	Finite state machines, hash search	Inheritance, dynamic binding, parametric polymorphism	Visual formalisms, state charts, class diagrams	Upcasting, downcasting, run time type information, regular expressions
Syntactic analysis	Recursive descent parser	Recursive functions	Syntax trees, class diagrams	Type switch, multiple dispatch
Code generation	Polish notation	Collections	Interfaces of collections	
Computation (execution)	Stack	Stack operations	Stack implementations, transforming recursion to iteration	

Code Review Practices in a Programming Class

- "Human-based" quality assurance practice
- Extends learning process and enforcing practicing "soft skills"
- Around 40% of useful review comments are related to **evolvability defects**



* Bosu A., Greiler M., and Bird C., "Characteristics of useful code reviews: An empirical study at Microsoft," 12th Working Conference on Mining Software Repositories, 2015

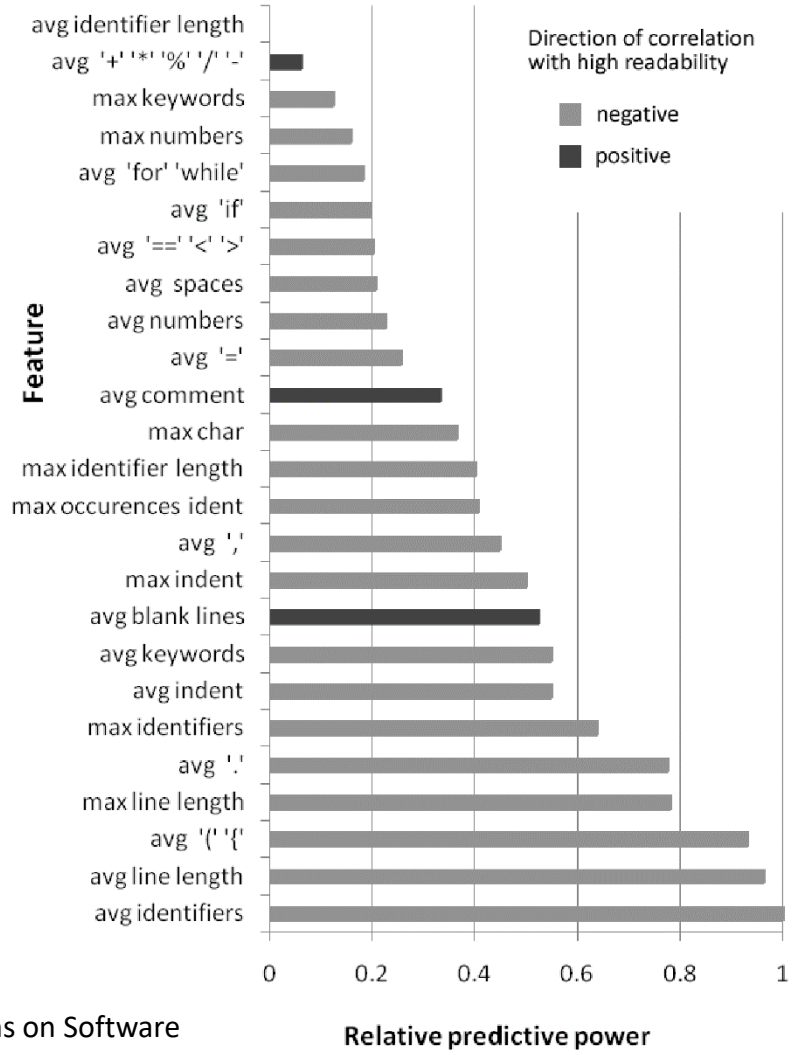
Readability Features: Example

Strong influence on readability

- Naming conventions
- Empty lines
- Number of identifiers and characters per line
- Specific indentation scheme

Moderate predictive readability power

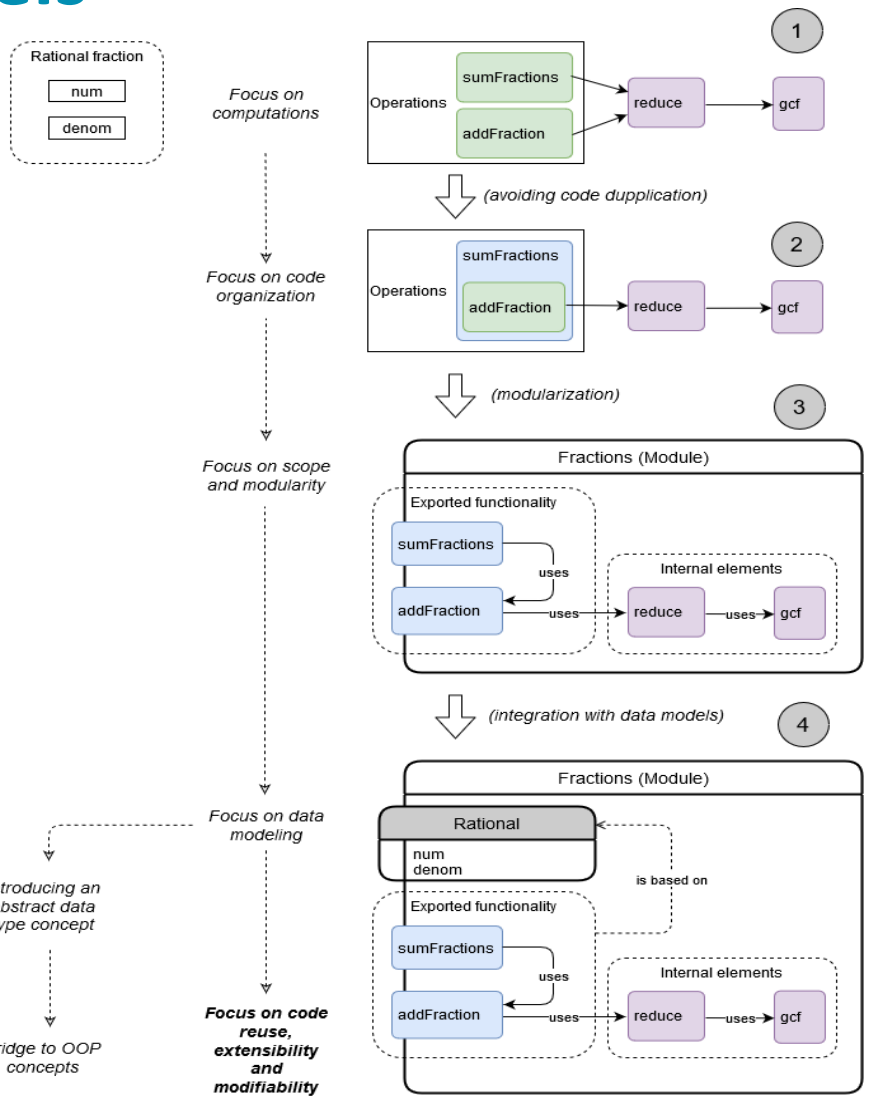
- Long identifier names
  - They are not directly connected to a concept of self-documented code
- Comments
  - While comments can enhance readability, they are typically used in code segments that started out less readable



* Buse R. and Weimer W.R., "Learning a metric for code readability," IEEE Transactions on Software Engineering 36, 4, 2010.

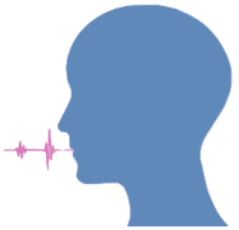
Incremental Design and Importance of Visual Models

- From the very first steps, it is important to introduce to students an approach to work on their practical assignments incrementally.
- Even classroom demos (should) be discussed in their possible evolvement
- *Example with Rational fractions (C Programming class):* **from imperative constructs to structural types and modularity**



Case Study and Discussion: Distant Learning Affordances and Constrains

- Pre-recorded videos
- Online meeting tools
- Learning management systems
- Version control systems
- Project management tools
- Code review tools
- Hybrid educational environments specifically focused on programming teaching
- Mind maps



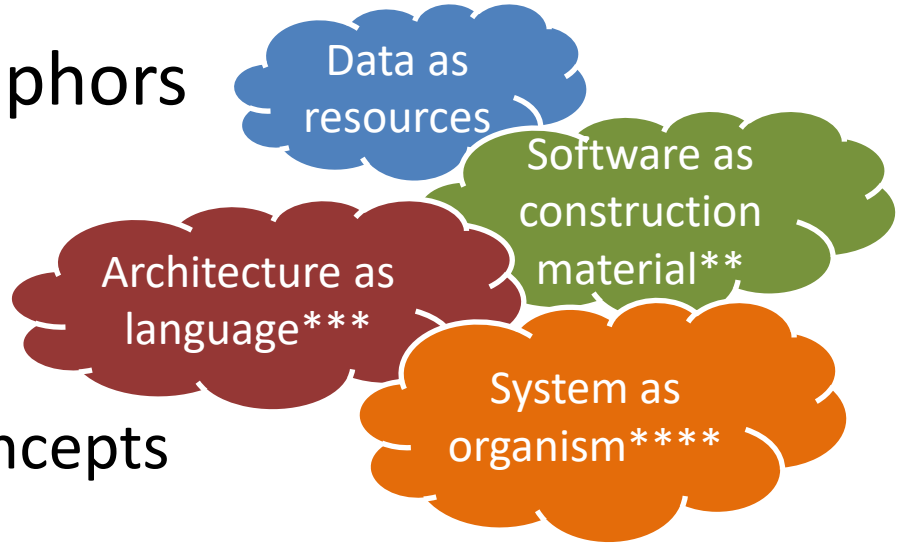
Challenges of Present Day Software in Frame of Education

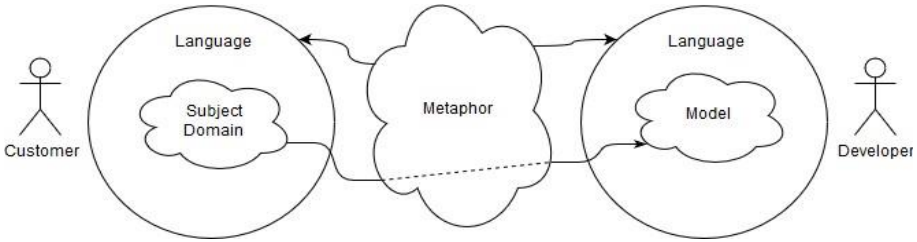
- ***"TRADITIONAL" SOFTWARE***
  - Logical and imperative
  - States and transitions
  - Limited parallelism
  - Data modeling and processing
  - Logic resolution

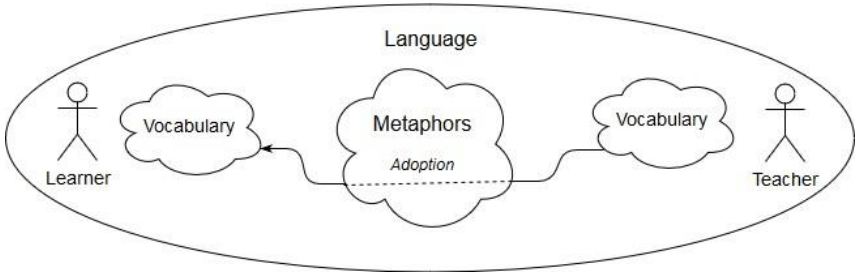
- ***AI BASED SOFTWARE***
  - Associative
  - Neural networks
  - Inherently parallel
  - Machine learning
  - Inference (prediction)

Revisiting Metaphors in (Software) Education

- Importance of good metaphors in educations
  - For mapping domains
  - For reifying abstract ideas
  - For conveying complex concepts
  - For bridging communities







* E. Pyshkin and J. Blake, A Metaphoric Bridge: Understanding Software Engineering Education through Literature and Fine Arts," Society. Communication. Education. Vol. 11, Issue 3. 2020. To appear.
** J. Carbonell, A. Sa´nchez-Esguevillas, and B. Carro, "The role of metaphors in the development of technologies. The case of the artificial intelligence," Futures, vol. 84, 2016, pp. 145–153.
*** K. Smolander, "Four metaphors of architecture in software organizations: finding out the meaning of architecture in practice," in Proceedings of International Symposium on Empirical Software Engineering. IEEE, 2002, pp. 211–221.
**** J. E. Kendall and K. E. Kendall, "Metaphors and methodologies: Living beyond the systems machine," MIS quarterly, 1993, pp. 149–171.

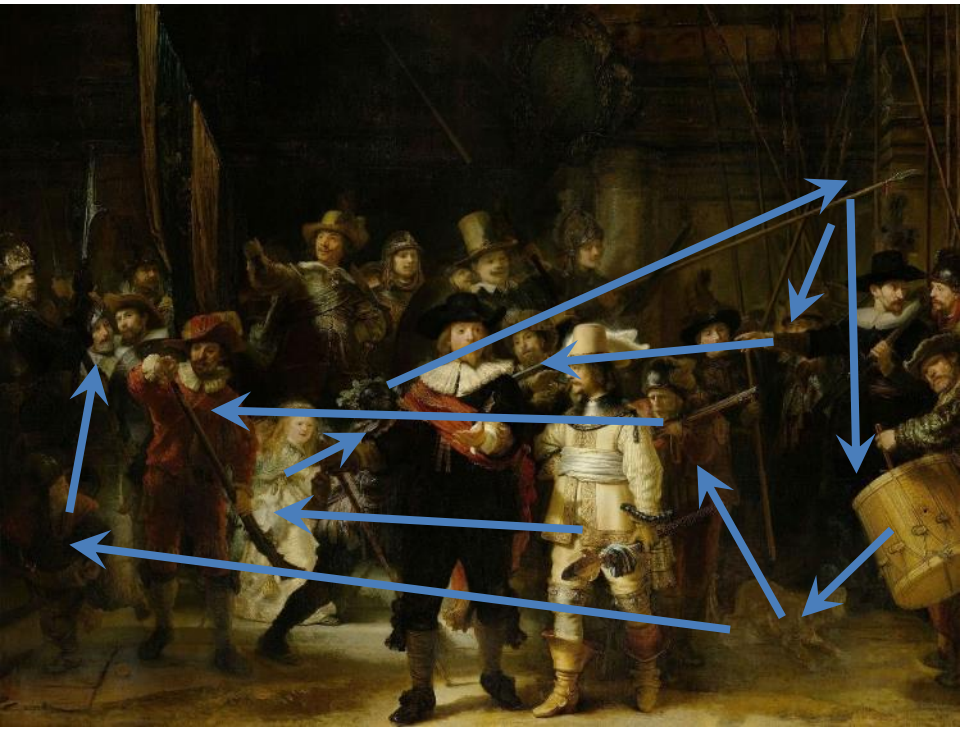
Software Metaphors: Examples

- Program objects
  - Scope
  - Assignment
  - Lifetime
- Control structures
  - Decision
  - Loop
- Modular structures
  - Library
  - Package
- Program workflow
  - Thread
  - Lazy computation
- Interface design
  - Menu
  - Palette
  - Files and folders
- Design patterns
  - Factory method
  - Delegation
  - Future
- Software analysis
  - Bad smell
  - Refactoring
  - Code mutant

Application of Metaphors from Fine Arts to Programming Teaching

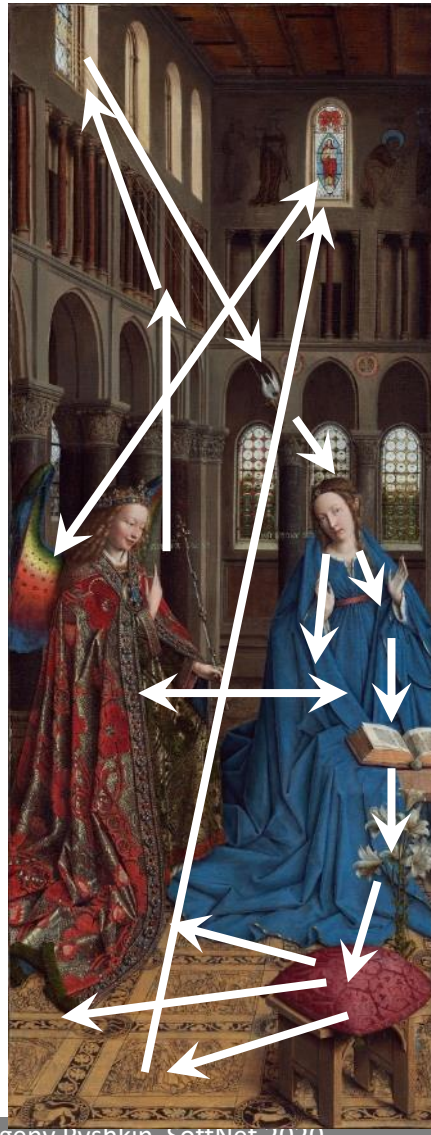
L'oeil suit les chemins qui lui ont été ménagés dans l'oeuvre

Paul Klee, quoted in George Perec's *La Vie, mode d'emploi* (1978)



Unless you write a Chinese character in the right order of brush strokes, it will never look beautiful!

道



Acknowledgements to Collaborators

Natalia Bogach

Peter the Great St. Petersburg Polytechnic University

Vlatko Davidovski

Cognizant Business Consulting Strategy & Transformations

Vitaly Klyuev

University of Aizu

John Blake

University of Aizu

Maxim Mozgovoy

University of Aizu

Takako Yasuta

Fukushima Medical University

Andrei Kuznetsov

JetBrains

IARIA
International Academy, Research, and Industry Association
SoftNet 2020
Porto
October 18 –22, 2020

CHALLENGES OF DISTANT LEARNING IMPLEMENTATION OF A PROGRAMMING CLASS

Evgeny Pyshkin

University of Aizu