

高级软件工程

第二周 (Sep. 15)

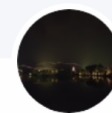
主讲：罗铁坚

助教：周文璋、俞永生、姚敏

上课：周一、三 上午 10:30—12:10

答疑：周一、三 下午 2:30—3:30 (学园2-485)

联系：tjluo@ucas.ac.cn 69671829



2021高级软件工程

群号：544158863



扫一扫二维码，加入群聊。



问题

- 1、 人类发明软件的动机是什么？
- 2、 软件主要解决什么类型问题？ 它的优点和局限性是什么？
- 2、 用软件解决业务问题的基本条件和途径是什么？
- 3、 为什么说人工智能或机器人是软件的一种能力。
- 4、 你认为设计和实现软件必须解决什么关键问题？

讨论主题

- 计算机和软件从无到有，经历了80年的跌宕起伏，发展成一个让人激情澎湃的重要行业。它影响和重塑了几乎人类的所有事业。(参见书：“The Technical and Social History of Software Engineering—1930-2019”)。
- 这段历史给我们什么启示？
- 为什么软件工程项目的失败率高于其他行业？
- 为什么生命周期达到20年的软件产品（或系统）不超过50%。80%软件的生命周期不超过5年。
- 介绍可计算理论做出贡献的几个关键人物，然后提出软件工程面临3个基本问题，设计和实现软件的9类工作任务和2种实现途径。

回顾教学大纲

大纲章次	章名称	章学时	授课教师	大纲小节次	小节名称	节学时
第1章	引言(第1周)	4		1.1	学科发展动机	1.0
				1.2	预期学习效果	1.0
				1.3	知识能力及学习策略	1.0
				1.4	课程目的和考核要求	1.0
第2章	研究主题（第2周）	4		2.1	理论奠基者	0.8
				2.2	三类问题	0.8
				2.3	二种途径	0.8
				2.4	九项任务	0.8
				2.5	练习与问题	0.8
第3章	领域问题和应用场景（第3、4周）	8		3.1	讨论重点	1.1
				3.2	案例1 操作系统和应用框架（Linux and Ruby on Rail）	1.1
				3.3	案例2 交互程序设计语言环境（Jupyter notebook）	1.1
				3.4	案例3 玩魔方软件（Web 版与实体版）	1.1
				3.5	案例4 池塘大战软件（智能体博弈）	1.1
				3.6	案例5 软件版本管理与持续集成平台（GitHub等）	1.1
				3.7	练习与问题	1.1
第4章	业务建模和数据模型（第5、6周）	8		4.1	讨论重点	1.6
				4.2	领域知识与建模语言	1.6
				4.3	业务数据的外在形式与计算机的内部表示	1.6
				4.4	数据表示与应用功能	1.6
				4.5	练习与问题	1.6
第5章	软件架构和应用框架（第7、8周）	8		5.1	讨论重点	1.6
				5.2	架构与功能	1.6
				5.3	架构与数据	1.6
				5.4	抽象共性与知识重用	1.6
				5.5	练习与问题	1.6

回顾教学大纲

第6章	设计模式和代码重构（第9周）	4
第7章	用户体验和接口设计（第10周）	4
第8章	软件验证与自动测试（第11周）	4
第9章	系统安全和防御设计（第12周）	4
第10章	规模应用与性能评价（第13周）	4
第11章	集成部署和运营服务（第14周）	4

罗铁坚	6.1	讨论重点	0.8
	6.2	共性与特殊功能的分离	0.8
	6.3	应对功能变化的策略	0.8
	6.4	案例分析	0.8
	6.5	练习与问题	0.8
	7.1	讨论重点	0.8
	7.2	用户体验设计原则	0.8
	7.3	信息表示的目的与方法	0.8
	7.4	选择合适的可视化形式	0.8
	7.5	练习与问题	0.8
	8.1	讨论重点	0.8
	8.2	程序正确性证明	0.8
	8.3	软件验证方法	0.8
	8.4	构造测试用例	0.8
	8.5	练习与问题	0.8
	9.1	讨论重点	1.0
	9.2	软件安全的原因	1.0
	9.3	防御措施的建构	1.0
	9.4	练习与问题	1.0
	10.1	讨论重点	1.0
	10.2	观察性能指标	1.0
	10.3	提升应用性能	1.0
	10.4	练习与问题	1.0
	11.1	讨论重点	1.0
	11.2	持续集成与部署	1.0
	11.3	软件上线与服务模式	1.0
	11.4	练习与问题	1.0

研究主题

- 计算机和软件从无到有，经历了80年的跌宕起伏，发展成一个让人激情澎湃的重要行业。它影响和重塑了几乎人类的所有事业。(参见书：“The Technical and Social History of Software Engineering—1930-2019”)。
- 这段历史给我们什么启示？为什么软件工程项目的失败率高于其他行业？为什么生命周期达到20年的软件产品（或系统）不超过50%。80%软件的生命周期不超过5年。
- 本章首先了解一下对可计算理论做出贡献的几个关键人物，然后提出软件工程面临3个基本问题，设计和实现软件的9类工作任务和2种实现途径。

图灵的思想

It is customary, in a talk or article on this subject, to offer a grain of comfort, in the form of a statement that some particularly human characteristic could never be imitated by a machine. It might for instance be said that no machine could write good English, or that it could not be influenced by sex-appeal or smoke a pipe. I cannot offer any such comfort, for I believe that no such bounds can be set. But I certainly hope and believe that no great efforts will be put into making machines with the most distinctively human, but non-intellectual characteristics such as the shape of the human body; it appears to me to be quite futile to make such attempts and their results would have something like the unpleasant quality of artificial flowers. Attempts to produce a thinking machine seem to me to be in a different category. The whole thinking process is still rather mysterious to us, but I believe that the attempt to make a thinking machine will help us greatly in finding out how we think ourselves.

ALAN TURING

用计算机解决问题方法的奠基者



David Hilbert

Challenge

“Entscheidungsproblem” (Decision Problem). Mathematics must be decidable. “We must know, we will know!”



Kurt Gödel (1931):
Incompleteness Theorems



Alonzo Church (1936):
Lambda Calculus



Alan Turing (1936):
Turing Machine

函数编程模式与Lambda演算的影响

希尔伯特、哥德尔、邱琦、图灵的等人奠定了计算机技术的理论基础。

函数编程模式与Lambda演算、MapReduce 是否有关系？下面的函数式程序的是继承了Lambda演算的语法结构的表示。

下面分别用5种程序设计语言（面向函数式编程模式）来表达Lambda演算，观察到在各种实现方式中，我们可以发现他们与面向过程编程模式不同，函数式模式可以用声明方式和不需要循环变量和中间变量来表达处理集合类的对象。表达更简洁和高效，现代的程序设计语言几乎都拓展了这种函数式编程的能力（如JavaScript, Java, Python, Ruby, Clojure等）。

函数编程模式与Lambda演算的影响

Functions - passed to variable definitions

Javascript

```
function foo() { return 42}  
var bar = function() { return 43}  
x = [foo(), bar()]
```

```
Array [  
  42,  
  43,  
]
```

EcmaScript6

```
baz = () => 42  
baz()
```

```
42
```

Clojure

```
(defn foox [] 42)  
(def bax (fn [] 43))  
[(foox) (bax)]
```

```
[42 43]
```

Ruby

```
def foow  
  42  
end  
baw = →() { 43 }  
[foow(), baw[]]
```

```
[42, 43]
```

函数编程模式与Lambda演算的影响

Functions - passed as arguments to functions

In function languages, you can create anonymous functions i.e. functions that don't have a name.

Javascript

```
[1,2,3].map(function(x) { return x + 1})  
Array [  
  2,  
  3,  
  4,  
]
```

EcmaScript6

```
[1,2,3].map((x) => x+1)  
Array [  
  2,  
  3,  
  4,  
]
```

Clojure

```
(map inc [1 2 3])  
(2 3 4)
```

Ruby

```
[1,2,3].map {|x|  
  x + 1  
}  
[2, 3, 4]
```

函数编程模式与Lambda演算的影响

Functions that return functions

An important function in FP is `partial` (or `curry`).

`partial` takes a function and a list of arguments and returns a function where some arguments are fixed.

Let' s see it in action!

Javascript

We are going to use [Ramda](#).

```
function add(x, y) { return x + y }  
var add10 = R.partial(add, [10])  
add10(19)  
29
```

EcmaScript6

```
const add = (x, y) => x + y  
const add10 = R.partial(add, [10])  
add10(19)  
29
```

Clojure

```
(defn add [x y] (+ x y))  
(let [add10 (partial add 10)]  
  (add10 32))  
42
```

Ruby

In ruby, it' s called `curry`.

```
add = ->(x,y) do x + y end  
add10 = add.curry.(10)  
add10.(9)  
19
```

函数编程模式与Lambda演算的影响

Declarative style

Let's compare `map` and `for` in javascript.

With `map`, you don't deal with low-level details

You only express what to do with each element of the array.

```
[1,2,3].map(x => x + 1)
```

```
Array [  
  2,  
  3,  
  4,  
]
```

With `for`, you have deal with low-level details.

You have to express:

- how the array is iterated
- what is the name of the index
- how the elements are combined in the returned array

```
const arr = [1,2,3]  
const f = x => x + 1  
let retArr = []  
for(let i = 0; i < arr.length; i++) {  
  retArr.push(f(arr[i]))  
}  
retArr
```


```
Array [  
  2,  
  3,  
  4,  
]
```

获得图灵奖学者的研究主题和学科贡献的关键词

Analysis of Algorithms **Artificial Intelligence**
Combinatorial Algorithms Compilers **Computational Complexity**
Computer Architecture Computer Hardware **Cryptography**
Data Structures Databases Education Error Correcting Codes Finite Automata Graphics
Interactive Computing Internet Communications List Processing Numerical Analysis
Numerical Methods Object Oriented Programming Operating Systems Personal Computing
Program Verification Programming
Programming Languages Proof Construction Software
Theory Software Engineering
Verification of Hardware and Software Models Computer Systems Machine Learning
Parallel Computation



八位对软件工工程学科贡献的图灵奖人物




PHOTOGRAPHS

ROBERT (BOB) W FLOYD

United States – 1978

Research Subjects

Software

 ACM DL
AUTHOR PROFILE

Bibliometrics: publication history

Publication years	1960-2007
Publication count	32
Citation Count	853
Available for download	22
Downloads (6 Weeks)	181
Downloads (12 Months)	1,635



MICHAEL STONEBRAKER

United States – 2014

Research Subjects

Databases
Software



Bibliometrics: publication history

Publication years	1971-2012
Publication count	287
Citation Count	5,473
Available for download	100
Downloads (6 Weeks)	2,721
Downloads (12 Months)	25,171



PHOTOGRAPHS

FREDERICK ("FRED") BROOKS

United States – 1999

Research Subjects

Computer Architecture
Operating Systems
Software Engineering



Bibliometrics: publication history

Publication years	1957-2012
Publication count	84
Citation Count	2,713
Available for download	42
Downloads (6 Weeks)	283
Downloads (12 Months)	2,053



PHOTOGRAPHS

EDMUND MELSON CLARKE

United States – 2007

Research Subjects

Verification of Hardware and Software Models



Bibliometrics: publication history

Publication years	1976-2012
Publication count	220
Citation Count	8,222
Available for download	49
Downloads (6 Weeks)	479
Downloads (12 Months)	4,233



八位对软件工科学科贡献的图灵奖人物



PHOTOGRAPHS

E. ALLEN EMERSON

United States – 2007

Research Subjects

Verification of Hardware and Software Models



Bibliometrics: publication history

Publication years	1980-2011
Publication count	108
Citation Count	4,187
Available for download	25
Downloads (6 Weeks)	328
Downloads (12 Months)	2,830



JOSEPH SIFAKIS

France – 2007

Research Subjects

Verification of Hardware and Software Models



Bibliometrics: publication history

Publication years	1976-2012
Publication count	145
Citation Count	1,992
Available for download	16
Downloads (6 Weeks)	263
Downloads (12 Months)	2,310



PHOTOGRAPHS

KRISTEN NYGAARD

Norway – 2001

Research Subjects

Object Oriented Programming



Bibliometrics: publication history

Publication years	1966-2002
Publication count	19
Citation Count	607
Available for download	10
Downloads (6 Weeks)	37
Downloads (12 Months)	353



PHOTOGRAPHS

OLE-JOHAN DAHL

Norway – 2001

Research Subjects

Object Oriented Programming



Bibliometrics: publication history

Publication years	1966-2002
Publication count	21
Citation Count	567
Available for download	8
Downloads (6 Weeks)	40
Downloads (12 Months)	313

上面介绍的希尔布特、邱琦、图灵及其几十位图灵奖获得者对计算机科学和软件工程有突出的贡献。但并不意味着，只有这些人推动了学科进步。↵

2.2 三类问题←

什么是软件？软件工程面临什么基本问题？←

软件是利用计算机拓展人类理解事物、执行任务和解决问题的智力系统（虽然它会嵌入和融入物理系统或产品，但它与物理系统或产品是有明显区别）。人类的智力(Human Cognition) 可分为5个层级(Bloom理论)：记忆、理解、应用、分析、评价和创造，计算机软件 (Machine Intelligent) 的能力是不断逼近或超越人类智力。计算机软件经过80年的研究与发展，已经在某些专门领域（计算、记忆、棋类、电竞、海量信息检索、自动生产线等领域）超过了人类。而软件的研发过程则是理解业务问题，通过设计程序代码或训练模型等方式把解决问题的“智能”封装到计算机系统的一项活动。我们认为，**任何人工智能的载体是计算机软件，它们解决问题的核心能力体现为软件的功能（更准确的预测和更优化的行动决策）。**←

对软件的创造构思、设计代码、实现系统和组织运营（CDIO，Conceive – Design – Implementation – Operation ）的知识进行系统化归纳提炼而逐步形成了软件工程这个学科领域。软件工程的使命和任务是探索和实践高效研发面向解决实际问题的优质智力系统或产品的理论方法和实现技术。←

三类根本问题之一

软件工程是要产生有价值并且可运行的系统或产品, 这些系统或产品一定是解决某一个(类) 业务问题。而软件工程的产出物的**有型物是程序代码（或机器学习训练出的模型）和数据信息**。为了得到**有价值**和**可运行的系统**, 软件工程的理论和实践必须解决如下三类根本问题。↵

1. 构思创造发明**有价值的软件系统或产品**和找到**检验其质量**的理论和方法。↵

软件发展历史表明, 80%的软件失败(延期交付或使用年限短等) 的原因是软件系统或产品并没有应用价值(显性功能)。20%的软件失败是原因是没有达到性能和安全等非功能质量要求。理论上, 是否存在或找到更好的方法来**推断或预测**拟开发的软件系统有价值并满足质量要求。实践中, 是否可根据拟解决的领域问题**归纳出有价值的使用场景或找到优化目标函数和有效的机器学习训练数据集?** 需求是否合理、准确或有智力成份? 是否存在或找到最小的测试用例能覆盖和捕捉到软件运行时所有可能条件分支或异常情况? ↵

三类根本问题之二

2. 探寻应对领域问题和软件需求变化的**可动态升级软件模块和架构**的理论和方法。←

软件架构是指为了满足软件质量要求（包括需求的实现），采用什么样的软件组件及其组合（调用）关系，以及对需要开发的软件组件，采用什么样的代码组织方式，采用什么样的组件或子系统的集成方式来实现目标软件系统。好的软件架构还需要满足软件需求变化后仍能保证后续的迭代演化后，不导致软件系统退化（指软件质量下降）。面对仍在使用的遗留（旧）系统，为了升级维护（排错和新增功能）软件系统，如何对程序代码进行新增或重构，仍能保证系统正常运行。是否能“找出”原有软件模型的业务逻辑或测试验证的数据或代码。←

三类根本问题之三

3. 探讨测算软件设计实现和部署运营的**效益模型**的理论和方法。↵

创新软件的构思、软件系统的设计与实现、软件的部署和运维是消耗资源的，如何合理分配软件开发项目中的技术投入、人力安排、时间计划、质量标准等方面的资源，并进行合理优化。通常来说，开发出高质量的软件所需的成本是比较高的。如何寻找软件开发费用与投资人和用户满意度平衡点？能否设计出一种可持续发展的软件演化途径和模式？↵

总结

推动计算机软件学科发展的动力来自于提出好的基础研究科学问题。解决历史上（1900年）的23个数学问题之一，即数学上的任何函数都可以用Lambda演算表示和完成计算，而Lambda演算表示和计算又可以用图灵机（机械计算模型）来完成计算，从而给计算机发展打下了坚实的理论基础。

软件工程失败率高的原因是没有意识到或忽略这三个根本问题，或者说，没有解决这三个根本问题的理论和方法。

计算机软件本质是解决领域问题的能力系统，人工智能只是软件的一种能力，它的模型训练和落地应用都需要现有的软件理论和技术的支持。

我们观察和归纳的两种设计和实现软件基本途径，回答了他们各自的特定要求、关注点及检验标准。

在软件产品或服务的**创造构思、设计代码、实现系统和组织运营过程中**，我们提出了实现具有软件产品或服务必须完成的九项任务。

思考题

- 1、人类发明软件的动机是什么？软件主要解决什么类型问题？它的优点和局限性是什么？
- 2、用软件解决业务问题的基本条件和途径是什么？
- 3、为什么说人工智能或机器人是软件的一种能力。
- 4、你认为设计和实现软件必须解决什么关键问题？