Michel RAYNAL

Member of Academia Europaea
Senior Member Institut Universitaire de France

Emeritus Professor, IRISA-ISTIC, Université de Rennes 1
Campus de Beaulieu, 35042 Rennes, France

Distinguished Chair Professor, Hong Kong Polytechnic University

https://team.inria.fr/wide/team/michel-raynal/
raynal@irisa.fr

February 2021

Part 1/3:

CV, Biography

and a View of my Job in my Scientific Domain

Next parts:

- Part 2: Recognition, management of scientific activities and research grants in the recent past.
- Part 3: Past and present research activities, and list of publications.

To make the reading of each part as independent as possible from the other parts, a few short paragraphs appear in several parts.

Contents

Michel RAYNAL
Senior Member Institut Universitaire de France
Member of Academia Europaea
Full Professor, Université de Rennes 1
IRISA, Campus de Beaulieu, 35042 Rennes, France
Scientific impact (as measured by Google Scholar):
h-index: 61 | i10-index: 304
Citations: 14295
Number of co-authors: 183
Guide2Research Int'l: 1627 Nat'l: 26

1 Curriculum vitæ

▷ Male, French, Born January 16, 1949 (Douelle, Lot, France). Married, one son.
▷ IRISA, Université de Rennes 1, Campus de Beaulieu, 35042 Rennes-cedex, France.
▷ raynal@irisa.fr. Tel (Office): 33 2 99 84 71 88, https://team.inria.fr/wide/team/michel-raynal/

Education

▷ "Doctorat d'Etat" in *Informatics*[1], University of Rennes, 1981.
▷ PhD in *Informatics*, University of Rennes, 1975 (PhD grant from French CNRS).
▷ Engineer diploma (INSA, Rennes), 1973.
▷ French Baccalauréat in Sciences (1968) and Literature (1969).

Professional experience and international recognition

▷ 2019: ACM Sigops France "*Outstanding Career*" Award.
▷ 2018: IEEE "*Outstanding Technical Achievement in Distributed Computing*" Award.
▷ 2017: *Distinguished Chair Professor*, Polytechnic University (PolyU), Hong Kong.
▷ 2015: Elected member of *Academia Europaea*.
▷ 2015: SIROCCO Prize "*Innovation in Distributed Computing*".
▷ 2013: *Adjunct Professor*, Polytechnic University (PolyU), Hong Kong.
▷ 2010: Senior Member, *Institut Universitaire de France* (IUF).
▷ 1989: Full professor, IRISA, University of Rennes, France (Prof. first class 1989, except. class 1998).
▷ 1984-1989: Associate professor, IRISA, University of Rennes, France (Prof. 2d class).
▷ 1984-2002: Founder and head of ADP (Algorithmes distribués et protocoles) research group, IRISA.
▷ 1981-1983: Professor, founder and head of the CS dpt, ENST (Telecom engineer school), Brest.
▷ 1976-1981: full-time researcher, INRIA, Rennes.

Research interests

▷ Distributed algorithms, distributed systems, distributed computability, synchronization, dependability.
▷ Fundamental principles that underlie the design and construction of distributed computing systems.

Outside informatics

▷ I enjoy literature, rugby, hiking, and cats. I am a wine amateur (Vigneron d'honneur de la confrérie de Saint-Emilion) and enjoys Cahors's wine (A *Malbec* that is the darkest wine in the world!).
▷ My Erdös number is 2 (Erdös → Zaks → Raynal).
▷ Cited in the French Whoswho since 2009.

[1] As nicely stated by E.W. Dijkstra (1920-2002): "*Computer science is no more about computers than astronomy is about telescopes*". Hence, to prevent ambiguities, I use the "European" word *Informatics* in place of *Computer science*. On a pleasant side, there is no more "computer science" than "washing machine science".

2 Biography

After obtaining an engineer diploma from INSA (Institut National des Sciences Appliquées de Rennes), I obtained a PhD grant from the CNRS, and defended in 1975 a PhD (supervised by J.-P. Verjus) the topic of which was related to synchronization. I was then hired as a full-time researcher by IRIA (now INRIA) from 1976 until 1981, where I worked on abstract data types, protection, synchronization, and programming languages. In 1981 I obtained the "Doctorat d'Etat" degree in informatics, the title of which was "Contribution à l'étude de la coopération dans les langages et les systèmes informatiques". Then, I moved to Brest (France) in an engineer school (namely ENST de Bretagne, a French engineer school on telecommunications, sister-school of ENST ParisTech), where, as a professor, I created and managed the "computer science and engineering" department.

In 1984, I moved back to the university of Rennes where, since then, I have been a professor in informatics. At IRISA (CNRS-INRIA-University joint computing research laboratory located in Rennes), I founded a research group on Distributed Algorithms in 1984 (one of the very first groups on this topic in Europe at that time). Moreover, I had a consulting position at CNET (national research center of Trance-Telecom) during the period 1983-1986.

My research interests includes distributed algorithms, distributed computing systems, distributed computability and dependability. My main interest lies in the fundamental principles that underlie the design and the construction of distributed computing systems. I have been Principal Investigator of many research grants in these areas (founded by the European community, private companies -such as Alcatel, GEC-Alsthom and France-Telecom-, or the French government). I have also obtained grants from bi-national research programmes between France and other countries such as Brazil, Hong-Kong, Israel, Italy, Japan, Mexico, Portugal, and USA (Santa Barbara, Georgia Tech, Kansas State U.). I have been invited by more than 55 universities all over the world (Europe, North and South America, Africa, and Asia) to give lectures on distributed algorithms and distributed computing.

Up to now, I published 177 papers in journals. These journals cover both theory and practice. Among them, there are the following prestigious journals: the Journal of the ACM, Algorithmica, SIAM Journal of Computing, Acta Informatica, Distributed Computing, The Communications of the ACM, Information and Computation, Journal of Computer and System Sciences, JPDC, IEEE Transactions on Computers, IEEE Transactions on Software Engineering, IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on TPDS, IEEE Computer, IEEE Software, Journal of Supercomputing, IPL, PPL, Theoretical Computer Science, Theory of Computing Systems, Real-Time Systems Journal, The Computer Journal, etc. I have also published 360 papers in conferences (ACM STOC, ACM PODC, ACM SPAA, IEEE ICDCS, IEEE DSN, DISC, COCOON, IEEE IPDPS, ICDCN, OPODIS, Europar, FST&TCS, IEEE SRDS, etc.), and written twelve books devoted to parallelism, distributed algorithms and systems (published by the MIT Press, Wiley & Sons, Morgan & Claypool, and Springer). I have been an invited speaker in more than 34 international conferences (including the prestigious DISC, IEEE ICDCS, SIROCCO, EUROPAR, ICDCN, IEEE NCA, and OPODIS conferences). Among my publications, 527 are listed when querying DBLP. My current *h-index* is 61 (as computed by Google Scholar).

I currently serve in the editorial board of three international journals. I served in program committees for more than 160 international conferences (including ACM PODC, DISC, ICDCS, IPDPS, DSN, LADC, SRDS, SIROCCO, ICDCN, SSS, NETYS, OPODIS, etc.) and chaired the program committee of more than 20 international top conferences (including DISC -twice-, ICDCS, SIROCCO, ICDCN, OPODIS, and NETYS). I received 8 best paper Awards in international top conferences (IEEE ICDCS three times in a row 1999, 2000, and 2001, SSS in 2009 and 2011, Europar in 2010, DISC in 2010, PODC in 2014). In the past five years, I chaired ICDCN 2013 (devoted to distributed computing and networked systems, LNCS 7730) and NETYS 2014 (devoted to networked systems, LNCS 8593). I also wrote three books (500 pages each) one on fault-tolerant synchronization (Springer 2013), one on

failure-free distributed computing (Springer 2013), and one on fault-tolerant message-passing systems (Springer 2018).

I served as the chair of the steering committee leading the DISC symposium series in 2002-2004, I have been a member of the steering committees of ACM PODC (ACM Symposium on the Principles of Distributed Computing) during the period 2005-2009, and I am a member of the steering committees of IEEE ICDCS (Int'l Conference on Distributed Computing and Systems), and ICDCN (Int'l Conference on Distributed Computing and Networks). Since 2006, I am the European representative in the IEEE technical committee on Distributed Computing. Since 2010, I am a senior member of the *Institut Universitaire de France*. In 2015, I was awarded the Prize "*Innovation in Distributed Computing*" (award ceremony during the SIROCCO 2015 conference), and was elected member of *Academia Europaea*. Since 2017 I have a Distinguished Chair Professor position at the department of computing at Hong Kong Polytechnic University. In 2018, I received the *Outstanding Technical Achievement Award* from the IEEE Technical Committee on Distributed Processing. In 2019, I received the *Outstanding Career Award* from ACM Sigops France. I am currently the editor of the *Synthesis Lectures on Distributed Computing Theory* published by Morgan & Claypool.

On visiting (research/teaching) positions
Lots of stay (two weeks to several months) in a lot of labs and universities all over the world.

▷ IBM Almaden (California, USA),
▷ University of Santa Barbara (USA),
▷ Austin University (USA),
▷ Georgia Tech, Atlanta (USA),
▷ EPFL, Lausanne (Switzerland),
▷ Université de Montréal (Canada),
▷ Hong-Kong Polytechnic University,
▷ University of Nanjing (China),
▷ University of Guangzhou –Canton– (China),
▷ Federal University of Salvador de Bahia (Brazil),
▷ UNAM (Mexico),
▷ Universidade Rey Juan Carlos (Madrid),
▷ University of the Basque Country (San Sebastian, Spain),
▷ Konkuk university (Seoul, South Korea),
▷ Tokyo Denki University (Japan),
▷ Seikei University (Japan),
▷ JAIST (Japan),
▷ University of Lisboa (Portugal),
▷ Università di Roma La Sapienza (Italy),
▷ Università di Napoli (Italy),
▷ Ecole nationale d'ingénieurs de Tunis (Tunisia),
▷ Institut national d'informatique d'Alger (Algeria),
▷ Université de Yaoundé (Cameroun),
▷ ENSIAS, Rabat (Morocco),
▷ Ben Gourion University (Israël),
▷ Sun Yat-Sen University (Guangzhou, China),
▷ Hosei University, Tokyo (Japan),
▷ East China Normal University (Shanghai, China),
▷ Mohammed VI Polytechnic University (Morocco).

3 A glimpse at a few figures

Miscellaneous	01/2021
Books	12
Chapters in books/encyclopedia	10
Papers in peer-reviewed Journals	177
Papers in peer-reviewed Conf. proceedings	360
Best paper awards (in top conferences)	8
Invited Papers (Int'l Conf + workshops)	34
Supervised PhD	38
Co-supervised PhD	10
Invited Courses/Lectures	$>$ 55
PC Member	$>$ 180
PC Chair and Proceedings Editor	$>$ 25
PhD Committees	$>$ 190
Nb of co-authors (as cited by DBLP)	183

Numerical impact factors	01/2021
h-index Google scholar	61
i-10 index Google scholar	304
Nb citations Google scholar	14295
Articles cited by DBLP	527

Among my 360 articles in conferences	01/2021
Published by ACM (not counting short)	39
Published by IEEE	132
Published by Springer LNCS or LIPICS	135
Published by North Holland	11
PODC papers regular, short	18, 20
DISC papers regular, short	20, 5
ICDCS papers	24
SPAA papers	6
SIROCCO papers	15
OPODIS papers	16
SRDS and DSN papers	20

Among my 177 articles in journals	01/2021
IEEE Transactions in Parallel and Dist. Systems	16
IEEE Other Trans. (TC, TSE, TKDE, TDCS, TMC)	14
Distributed Computing	9
Journal of Parallel and Distributed Computing	11
Theoretical Computer Science (TCS)	13
Information Processing Letters	18
Parallel Processing Letters	10
JACM, SIAM JC, Algorithmica, JCSS, Springer ToCS Acta Informatica, Information & Computation	18
The Computer Journal	4
Magazines: IEEE Computer, IEEE Software, CACM	6
Articles in French Journals	18
Articles in Asian Journals	3

Downloaded chapters of my Springer books up to	06/2020
Concurrent programming ... foundations	46870
Distributed algorithms in message-passing systems	38221
Fault-tolerant ... an algorithmic approach	

4 Academic genealogy

Sources: https://academictree.org/physics/tree.php?pid=25655&fontsize=1&pnodecount=4&cnodecount=2 and https://genealogy.math.ndsu.nodak.edu

Johan Bernoulli (1667-1748)
Leonhard Euler (1707-1783)
Joseph Louis Lagrange (Giuseppe Luigi Lagrangia, 1736-1813)
Siméon Denis Poisson (1781-1840), Co-advised by: Pierre-Simon de Laplace (1749-1827)
Michel Chasles (1793-1880)
Gaston Darboux (1842-1917)
Émile Borel (1871-1956)
Georges Valiron (1884-1955)
Jean Kuntzmann (1912-1992)
Louis Bolliet (1928)
Jean-Pierre Verjus (1943)
Michel Raynal.

5 My view

5.1 What is research? What is teaching? A personal view

Our job (as a University Professor) combines research and teaching activities, which are the two faces of a same coin. This section presents personal views (that consequently are both partial and questionable). The style is voluntarily informal and I am conscious that the way this view is exposed is a little bit schematic or even provocative.

Research To me "research" is an *adventure, both personal and collective, of intellectual nature* (it is not a joint venture ...). We are working in a scientific domain (basically, "informatics" can be abstracted as the science of operations), and our job is to set and answer questions (specifically, given a set of operations, what can be computed, and -if any- which is the best solution). More generally, our job is to think -within our scientific domain- for the long run[2].

(I basically share the point of view expressed by Odeh Goldreich in his essai "On our duties as scientists"[3].) For us, application domains are more important because they ask for new solutions, than for their today economical value. We never have to forget that "it is not by improving the candle technology that electrical lamps have been discovered, understood and mastered".

Teaching For Henri Lebesgue (1875-1941) "teaching" was the activity of "penser à haute voix devant les étudiants" (to teach is to think in a loud voice in front of students"). It is not a quiz exercise. I share this view. Of course, new technologies (e.g. embedded in a new programming language) have to be taught, but ultimately, a pupil does not learn how to write in reading explanatory leaflets of washing machines, but in reading great authors from the literature. This means that our lectures have to provide the students with (1) enough food for their brain in order they be able to address and correctly solve the problems they will encounter, and (2) enough background knowledge and insight in order they still have a job in twenty years! To quote Lamport: "Teaching is not an accumulation of facts"[4].

Students are our "products", and, due to what they learned, each generation of students gives rise to a new way of thinking when they are going to work in a company, each generation entailing its "small revolution" in the software industry. That is our main impact on the society. Each new student generation makes the industry moves to better knowledge and better practices. That is why teaching is fundamental. And the teaching activity is difficult because we have to teach in a way as simple as possible, and simplicity is very difficult to reach (as Blaise Pascal wrote "I am sorry for having written such a long letter, I had not enough time to write a shorter one", and Albert Einstein wrote "Make it as simple as possible, but not simpler"). This motivated me to write books, and a great lot of survey papers on emerging topics.

[2]To better understand a part of the the duality researcher/engineer, I sometimes parallel it with the duality historian/journalist. The job of a historian is to analyze and relate events in time, in order to provide us with a continuous, consistent and global view of things that have happened. Differently, the job of a journalist is to relate (and sometimes analyze) the last events that occurred. Similarly to a historian, the job of a professor is to work and investigate scientific domains in order to provide the students with a deep and global view of these domains. Obtaining a view of a domain that allows to provide students with a deep scientific background requires a strong involvement in research. As for a journalist who, differently from a historian, works with "today" facts/inputs, the job of an engineer (differently from a theory researcher) depends highly on the current (perishable) technology. Of course, this view is a little bit schematic, but nevertheless captures a difference in two extreme behaviors encountered in the scientific and engineering communities working in informatics.

[3]http://www.wisdom.weizmann.ac.il/ oded/on-duties.html.

[4]In "Teaching concurrency", *ACM Sigact NEWS*, 40(1):58-62, 2009.

5.2 My view of informatics and distributed computing

Informatics can be defined as the meeting point between mathematics and technology[5]. Roughly speaking, its two components, *computing science* and *computing engineering*, can be seen as complementary facets: computing science is to understand, computer engineering is to build. Said in other words, we are concerned with a science of *abstraction*, namely, creating the right model for a problem and devising the appropriate mechanizable techniques to solve it. This is specifically true in (fault-tolerant/dynamic/large-scale/etc.) distributed computing where finding models that are realistic while remaining abstract enough to be tractable, was, is, and still remains a real challenge.

Distributed computing was born in the late seventies when people started taking into account the intrinsic characteristics of physically distributed systems. The field then emerged as a specialized research area distinct from networks, operating systems and parallelism. Its birth certificate is usually considered as the publication in 1978 of Lamport's most celebrated paper "*Time, clocks and the ordering of events in a distributed system*". Since then, several high level journals and (mainly ACM and IEEE) conferences are devoted to distributed computing.

Distributed computing arises when one has to solve a problem in terms of entities (usually called processes, agents, sensors, peers, actors, processors, nodes, etc.) such that each entity has only a partial knowledge of the many parameters involved in the problem that has to be solved. While parallelism and real-time can be characterized by the words *efficiency* and *on time computing*, respectively, distributed computing can be characterized by the word *uncertainty*. This uncertainty is created by asynchrony, failures, unstable behaviors, non-monotonicity, system dynamism, mobility, low computing capability, scalability requirements, etc. Mastering one form or another of uncertainty is pervasive in all distributed computing problems[6]. Finally, as the aim of a theory is to codify knowledge in order it can be transmitted (to students, engineers, practitioners, etc), research in distributed computing theory is fundamental. When something works we must know why it works, and when something does not work ... we must know why it does not work.

5.3 Why research in distributed computing is fundamental

Looking to the past to appreciate the future One of the main problems in the fifties, sixties and even seventies, was to produce efficient (sequential and parallel) programs. It appears that to attain this goal, researchers have spent lots of efforts in establishing strong results in algorithms and formal languages. The benefit is obvious. The results in algorithms and formal languages allowed us to replace tricks by scientific solutions based on systematic approaches. Now, thanks to their lectures on formal languages and algorithms students know what can be done, what cannot be done, and what can be done efficiently.

The same analysis holds for lock-based concurrency. The problem of mastering multiprogramming was addressed in the late sixties and early seventies. Thanks to the work of Brinch Hansen, Dijkstra and Hoare (among others), basic concepts to master lock-based synchronization were developed (e.g., semaphores and monitors) and an associated methodology based on invariants was developed[7]. Thanks to these results, students know how to manage and cope with multithreaded computing, and how to analyze multiprocess programs in failure-free environments.

Today, it is an obvious fact that languages and synchronization are useful, and, due to lots of associated results (e.g., the fact that the classes of deterministic FSA and the class of non-deterministic FSA are equivalent), that they are among the elements that set up informatics as a science[8]. We cannot

[5] In some sense, we could say that "Informatics is the language of technology".

[6] A foundational paper of distributed computing is the celebrated paper "*Impossibility of distributed consensus with one faulty process*" by Fischer M.J., Lynch N.A., and Paterson M.S., published in the Journal of the ACM, 32(2):374-382, 1985. This paper established the domain on sane foundations.

[7] "The origin of concurrent programming", Springer-Verlag, 534 pages (2002), Edited by P. Brinch Hansen.

[8] I have considered here only two domains (languages and synchronization). Of course, this list is not exhaustive. I could

imagine mastering object-oriented programming or software engineering without relying on the scientific background accumulated in language theory, synchronization and other basic domains. The actual advances in software engineering is (partially) an implicit output of these early results.

Why DC is fundamental The computational universe surrounding us today is clearly very different from that envisioned by designers forty years ago. Even the most futuristic visions of that time of supercomputing and parallel machines (which have guided the research and absorbed a consequent part of the research funding) are far from today's computational realities. More specifically, computing devices are conquering the world. They are spreading out everywhere (and we could now nearly say that a high speed train or a plane is a sophisticated local area network with "additional devices").

The today computing applications are characterized by networked entities communicating with each other, cooperating towards common tasks or the solution to a shared problem, and acting partially in an autonomous way. Said, differently, the computational world is inherently *distributed*. So, mastering information science and information technology in the future goes through mastering distributed computing.

The moral of the story is that we have to do *today* research in the basics of distributed computing if we want to be able to master *future* applications, to known what can be done, what cannot be done, what can be done efficiently, etc., despite physical program distribution, asynchrony, failures, mobility, dynamism, unstable behavior, scalability requirements, etc. We have to go from tricks to a scientific knowledge that can be transmitted to, and exploited by, engineers.

5.4 My view in French

Sous le titre "L'informatique, science et technique" le texte qui suit est paru –sous licence Creative Commons– dans le numéro 171 (2015) de la revue EpiNet (Revue lectronique de l'EPI – (association "Enseignement Public et Informatique"–)

Entre Bill Gates, Mark Zuckerberg, le Web Netscape, Facebook, etc. d'un coté, Alan Turing, les limites du calcul et la thèse de Church-Turing de l'autre, qu'est et où se situe l'informatique ? Un monde technique où priment la vitesse et la quasi-instantanéité ou bien un monde de la pérennité ? Tout enseignant-chercheur est interpellé par cette question tant dans sa façon de définir son enseignement et de le faire passer auprès des étudiants, que dans la thématique et la problématique de son activité de recherche.

Née de préoccupations concrètes (que l'on en voit l'origine dans les procédés de calcul inscrits sur les tablettes babyloniennes, la machine de Pascal ou la deuxième guerre mondiale, l'informatique est pour moi, enseignant-chercheur du supérieur, avant tout la science des algorithmes et des machines capables de les interpréter. Elle se caractérise par une démarche essentiellement constructive. Décidabilité, recherche de solutions optimales (algorithmes), définition et construction de machines (au sens large, c'est-à -dire comprenant systèmes et langages) en constituent ainsi le cœur. C'est en ce sens que l'informatique ne peut être ni réduite, ni confondue avec les avancées technologiques (par ailleurs remarquables) qui l'alimentent et qu'elle alimente. Toutefois cette perception de notre discipline (que d'aucuns pourraient qualifier de trop "académique", voire de "passéiste" !), ne doit pas être dissociée du monde des applications dont elle est issue : ce monde lui confère une dimension technique qu'il serait à son tour réducteur de ramener à un recueil de recettes. Il est par ailleurs important de noter que cette dualité science-technique de notre discipline façonne la perception que nous avons des algorithmes et des machines qui s'avèrent être importants à un instant donnné[9].

have taken similar examples in other domains such as databases, computability, algorithms, etc.

[9] Aujourd'hui, le monde des applications est fortement caractérisé par la dimension "monde de la vitesse et de

Ainsi ma philosophie d'enseignant a toujours consisté à donner aux étudiants cette double perception de l'informatique en faisant en sorte qu'ils perçoivent le monde de la pérennité comme un des prérequis indispensables pour mener à bien leurs futures réalisations. J'utilise souvent avec eux l'image suivante : "Vous avez une culture scientifique (en l'occurrence physique) qui vous permet de ne pas être mystifié lorsque quelqu'un prétend avoir inventé un moteur avec un rendement de 100%: votre connaissance des principes de la thermodynamique vous autorise à ne pas le croire. Il en va de même en informatique : la différence entre un hacker et vous passe par la connaissance d'un certain nombre de résultats qui vous permettront de distinguer ce qui est dans le domaine du réalisable de ce qui ne l'est pas, et, pour ce qui est faisable, par l'apprentissage des résultats (concepts, techniques et méthodes) qui vous permettront de résoudre vos problèmes. Le but de l'enseignement réside précisément là. Dans un contexte d'études supérieures, le savoir-faire est important, mais le savoir ne se réduit pas au savoir-faire (ni le savoir-faire au savoir). Le but de l'enseignement est de faire la part de chacun et de les enseigner tous deux en conséquence". C'est cet état d'esprit qui préside à mes cours et à présidé à l'écriture de mes ouvrages : assimiler les principaux résultats de recherche dans un domaine (à savoir les algorithmes pour les systèmes répartis) afin d'en faire passer l'essentiel auprès des étudiants. Une autre chose qu'il m'arrive de dire aux étudiants est la suivante "Votre connaissance des concepts "premiers" de la discipline et de la technologie matérielle ou logicielle (par exemple, tel matériel, tel langage ou tel système spécifique) est importante car c'est souvent elle qui vous permettra d'avoir un travail après avoir obtenu votre diplôme. La connaissance des concepts fondamentaux et le recul par rapport à ceux-ci sont encore plus importants car ce sont eux qui vous permettront d'avoir encore du travail dans vingt ans".

l'instantanéité" de l'informatique. Il suffit pour s'en convaincre de regarder les appels d'offres financés par la communauté européenne et les agences de recherche nationales.