Laudatio del Prof. Dr. D. Ricardo Peña Marí

con motivo de la investidura como doctor "honoris causa" del

Excmo. Sir. Charles Antony Richard Hoare

10 de mayo de 2013

Excelentísimo Sr. Rector Magnífico
Excelentísimas e Ilustrísimas Autoridades Académicas
Señores Profesores, Personal de Administración y Servicios, Estudiantes
Señoras y Señores

Professor Hoare, we welcome you to our University. It is a high honour for the Universidad Complutense to confer this Honorary Doctorate to a scientist like you, with such a long and fruitful career. We are very proud to receive you in our community as a colleague.

I must proceed in Spanish...

Sir Charles Antony Richard Hoare, conocido comúnmente entre los informáticos como Tony Hoare, es uno de los científicos que más huella han dejado en nuestra disciplina. En su larga vida científica, ha sentado los fundamentos de muchas de las teorías y técnicas que hoy enseñamos en las facultades de Informática de todo el mundo. Sus aportaciones han contribuido decisivamente a hacer de la programación de computadores una disciplina científicamente fundada, partiendo de una situación inicial en la que se parecía más a un arte, a un oficio artesanal, o en sus propias palabras, a una brujería, que a una ciencia.

Participó en los años sesenta en el diseño de los primeros lenguajes de programación estructurados, sentó las bases de la verificación formal de programas, estableció mecanismos de programación para la construcción fiable de sistemas operativos, creó modelos matemáticos para el razonamiento sobre programas concurrentes, y en los últimos años está dedicado a establecer relaciones entre distintos formalismos, tratando de crear una teoría unificadora de la programación.

Para los estudiantes de informática, Tony Hoare es el creador del famoso Quicksort, un algoritmo de ordenación cuya eficiencia promedio supera a la de todos los otros algoritmos de ordenación previos y posteriores.

Este Doctorado Honoris Causa pretende humildemente reconocer esta dilatada y fructífera trayectoria.

Tony Hoare nació en Colombo, capital de la antigua Ceilán, y se graduó en Lenguas Clásicas y Filosofía por la Universidad de Oxford. Comenzó trabajando en la Universidad Estatal de Moscú, donde ideó el algoritmo Quicksort. Entre 1960 y 1968 trabajó para la empresa británica Elliot Computers, una de las primeras compañías europeas que fabricaban computadores científicos. Allí dirigió el desarrollo de un compilador para el lenguaje Algol-60, y se involucró en el diseño de sus lenguajes sucesores: Algol-W, y Simula-67. En 1968 pasó a la universidad y fue profesor de la Queen University of Belfast. A partir de 1977, pasó a liderar el Programming Research Group de la Universidad de Oxford, donde transcurrió la mayor parte de su carrera profesional. Actualmente es Investigador Principal en el Microsoft Research Center de Cambridge.

Su artículo de 1969 "An axiomatic basis for computer programming", visto en perspectiva, puede considerarse como el equivalente a las Leyes de Newton de la programación. Por primera vez se enunciaban unas leyes lógicas que permitían comprender el significado de los programas independientemente de las máquinas que los ejecutan. Su lógica, llamada en su honor Lógica de Hoare, sentó las bases para el razonamiento matemático sobre los programas y continúa siendo usada en la actualidad, tanto manualmente como mediante el uso de herramientas. Con ella se ha descrito la semántica de numerosos lenguajes de programación.

Estos años, los finales de los 60, son cruciales para la historia de la programación. Son años de fracasos estrepitosos en el desarrollo de grandes sistemas software, como el del paradigmático sistema operativo OS/360 de IBM.

Se habla abiertamente de una "crisis del software", y el nombre "Ingeniería del Software" dado a una famosa conferencia patrocinada por la OTAN en 1968, se usa más con la intención de expresar un deseo que de describir la realidad predominante, en la que el software se construía con muy poca técnica y con unos lenguajes completamente inseguros.

Tony Hoare contribuyó en gran medida a cambiar esta situación, actuando en varias direcciones: primeramente, criticó duramente lenguajes como PL/I y Algol-68 que él consideraba formaban parte del problema. En ese sentido, abandonó junto con otros colegas el grupo IFIP 2.1 ocupado en la definición de este lenguaje Algol-68, y se unió al grupo 2.3 que había de definir buenas metodologías de programación. En su opinión, los lenguajes debían venir después y no antes de las metodologías.

En segundo lugar, fue cofundador del movimiento llamado de la "programación estructurada", editor del libro original con dicho nombre y autor de uno de sus tres capítulos. Este libro contenía las semillas cuyos frutos disfrutamos hoy y que todavía son parte esencial de las metodologías actuales: la necesidad de razonar formalmente sobre los programas, la necesidad de disminuir el número de detalles a tener en cuenta simultáneamente, la técnica llamada de diseño descendente, la necesidad de retrasar las decisiones de representación de las estructuras de datos, la peligrosidad de usar la instrucción go-to, y otras ideas que chocaban frontalmente con las prácticas del momento.

Finalmente, colaboró con su colega Nicolás Wirth en el desarrollo del lenguaje Pascal, el lenguaje de referencia de la programación estructurada, y proporcionaron una semántica formal para el mismo. Era la primera vez que a un lenguaje de programación se le daba un significado matemático.

Su trabajo de 1972 "Proof of correctness of data representations" extiende el razonamiento a las estructuras de datos que implementan un tipo abstracto, concepto este último en el origen de lo que hoy conocemos como programación orientada a objetos. Sus aportaciones de "invariante de la representación" y "función de abstracción" se enseñan actualmente en los cursos de estructuras de datos.

A partir de 1974, sus trabajos sobre monitores proporcionan un mecanismo elegante y fiable para gobernar la concurrencia en un solo computador, y para razonar sobre la corrección de este tipo de programas. Los programas concurrentes proporcionan la ilusión de que el computador realiza

varias tareas simultáneamente y se trata de un tipo de programación mucho más difícil y propensa a errores que la secuencial. Gracias a los monitores de Hoare no serán ya posibles desastres como los del sistema OS/360 previamente citado.

Muy poco después, aparecen en el mercado los primeros microprocesadores y se hace evidente que la programación concurrente va a convertirse enseguida en programación distribuida en la que intervendrán muchas máquinas simultáneamente. Con ello, los mecanismos preexistentes, incluidos sus monitores, que suponen la existencia de una memoria común, van a quedar obsoletos. El es el primero en proponer un mecanismo, sus Procesos Secuenciales Comunicantes (CSP) de 1978, para domesticar el nuevo monstruo. La compañía INMOS, que fabrica redes de microprocesadores, se inspira en ellos y le pide su colaboración para desarrollar el lenguaje Occam. Se trata de un bello ejemplo de transferencia entre la universidad y la industria y de cómo las buenas ideas se pueden transformar en muy poco tiempo en productos industriales útiles para la sociedad.

Convierte a continuación su lenguaje CSP en un modelo algebraico de procesos y se preocupa por encontrar para este modelo operacional una semántica denotacional al estilo de las de su predecesor en Oxford, Christopher Strachey. Con ello extiende la capacidad de razonamiento formal a los programas concurrentes distribuidos. Su libro de 1985 "Procesos Secuenciales Comunicantes" puso todo este material en forma didáctica y ha sido libro de referencia en numerosos estudios de Posgrado.

Por todos estos méritos, en 1980 recibió el Premio Turing de la ACM (Association for Computer Machinery), el máximo galardón que un informático puede obtener, considerado en nuestro ámbito como el equivalente al Premio Nobel. En la mención del premio se indica que "su trabajo se caracteriza por una combinación inusual de profundidad, originalidad, elegancia e impacto". Se da un Premio Turing por año a la persona o personas que más han contribuido con su investigación al avance de nuestra ciencia. Precisamente en 2012 se celebró en todo el mundo el Año Turing, centenario del nacimiento de Alan Turing, considerado como el fundador de los computadores y de la ciencia informática.

Para la Universidad Complutense es un gran honor conceder la mención de Dr. Honoris Causa a un científico como Vd. que ha sido distinguido con el premio Turing. Estamos muy orgullosos de recibirle en nuestra comunidad como colega.

El Premio Turing no supuso para Vd. un freno en realizar buena investigación. A partir de 1994 Vd. se interesa por elaborar teorías unificadoras de la programación que engloben como casos particulares muchas de las teorías especializadas que utilizamos en la actualidad. Se trata de un objetivo de largo alcance al que todavía se sigue dedicando en el presente. A este respecto dedica numerosos trabajos y un libro publicado conjuntamente con su colega He-Jifeng en 1998. En su visión, una ciencia madura debe disponer de esas teorías unificadoras. No hay nada equivalente en el campo de la programación al papel que juega la Mecánica Cuántica en la Física o el Álgebra en las Matemáticas.

En el primer caso, la teoría explica como casos particulares las teorías preexistentes sobre el calor, el electromagnetismo y la óptica. En el segundo, la teoría subsume las propiedades de muchas clases de números y de estructuras matemáticas previas a su aparición. Más importante aún es que dichas teorías unificadoras son capaces de predecir la aparición de nuevos fenómenos. Es el caso del recientemente descubierto bosón de Higgs, que fue predicho por una teoría unificadora de la Física elaborada cincuenta años atrás.

En su visión, una teoría unificadora de la programación ha de basarse en predicados y relaciones y tener un "sabor" algebraico. De hecho, Vd. ya nos ha proporcionado algunas de sus leyes. El objetivo final de esta teoría es alcanzar la corrección total de los programas que construimos los humanos, y que actualmente están tan plagados de errores.

En 2003 Vd. lanzó un Gran Desafío a la comunidad científica: profundizar en las teorías ya existentes, y crear las que fueran necesarias, para llegar a una meta difícil, pero en su opinión alcanzable, el Compilador-Verificador. Si se alcanzara dicha meta, la creación de programas sería una tarea mucho más profesional y científicamente fundada de lo que es actualmente. La interacción de los ingenieros con dicha herramienta crearía programas correctos por construcción y las pruebas de ejecución tan solo confirmarían dicha corrección, un ideal que hoy nos parece todavía lejano.

Cada artículo suyo se convierte inmediatamente en un clásico y es ampliamente citado por trabajos posteriores de otros investigadores. En una entrevista que concedió recientemente a un periódico español, Vd. nos alertó del peligro de medir la capacidad científica de los investigadores por el número de artículos que publican cada año. Esta práctica, extendida por las agencias de evaluación y universidades del todo el mundo incluida España, es muy perniciosa para la creación de buena ciencia. Estimula la fragmentación innecesaria de los

trabajos, la repetición parcial de los mismos, y el no realizar suficiente experimentación antes de enviar un trabajo a publicar. Algunos de sus artículos más citados le han supuesto a Vd. más de dos años de investigación. Seguramente una agencia de evaluación de nuestro país le reconvendría por su baja productividad.

Otro dato que revela su exquisito cuidado por el trabajo bien hecho, es que Vd. retiró un artículo de la revista Comunicaciones de la ACM, una vez que este había sido aceptado por el Comité Editorial, porque Vd. no estaba del todo satisfecho y había descubierto una forma de mejorarlo. Para que la audiencia pueda valorar la importancia de este gesto, diré que la revista CACM es una de las más prestigiosas de nuestro campo. Muy pocos investigadores, españoles o extranjeros, consiguen publicar en ella, y se considera un gran mérito científico tener un artículo publicado allí.

Vd. fue nombrado miembro de la Royal Society en 1982, siguiendo los pasos de Alan Turing, el padre de la Computación. Ha recibido numerosas distinciones, como la AFIP Harry Goode Medal, el Premio Kioto de Ciencias de la Información, la IEEE John von Neumann Medal, el ACM SIGPLAN Distinguished Achievement Award, así como diez Doctorados Honoris Causa en diferentes países. En 2000 fue nombrado Sir por la Reina Británica por sus servicios a la educación y a las ciencias de la computación. Este título lo ostentaban muy pocos científicos antes que Vd. en el Reino Unido. Entre ellos cabe destacar a Sir Isaac Newton, padre de la Mecánica. Vd. podría con toda justicia ser calificado como el padre de la Programación científicamente fundada.

La relación de Vd. con nuestra universidad se establece sobre todo a partir de su obra científica. Es difícil encontrar un investigador de los departamentos de la Facultad de Informática de la UCM que no conozca sus trabajos. En particular, el contenido de sus publicaciones sobre verificación de programas y sobre concurrencia forma parte de las enseñanzas que se imparten en los grados y másteres de la Facultad de Informática de la UCM. A nivel de investigación, se han defendido en los últimos años varias tesis doctorales que utilizan o extienden sus teorías, tanto en el ámbito de los modelos concurrentes como en el de la verificación y certificación de programas.

A nivel más personal, no ha sido infrecuente coincidir con Vd. en algunos congresos y escuelas de verano. En particular, hemos enviado a la Escuela de Verano de Marktdoberdorf, que Vd. ha dirigido durante más de veinte años, a docenas de nuestros graduados. Vd. fue conferenciante invitado en el CEDI

(Conferencia Española de Informática) en Granada en 2005, al cual asistieron cerca de 2.000 investigadores. El contacto más reciente se ha producido el mes de Junio pasado, debido a su participación como conferenciante invitado en el congreso Mathematics of Program Construction, celebrado precisamente en esta Facultad.

Decía Séneca que "conceder un beneficio a un hombre de honor es, en parte, recibirlo". Para mi universidad, para mi facultad, y para mi personalmente, es un gran honor y un valioso regalo otorgarle a Vd. este Doctorado Honoris Causa por la Universidad Complutense de Madrid, y lo es más aún el hecho de que Vd. haya aceptado recibirlo. Sir Tony Hoare, le damos las gracias por sus magníficas e inspiradoras contribuciones a la ciencia de la programación.

Sir Tony Hoare, we thank you for your magnificent and inspiring contributions to the science of programming.