A Tool for Quantum Software Evolution

Luis Jiménez-Navajas[a,b], Ricardo Pérez-del Castillo[b,c] and Mario Piattinia[a,d]

[a] *University of Castilla-La Macha, Paseo de la Universidad 4, Ciudad Real, 13071, Spain*
[b] *aQuantum by Alarcos Research Group, Paseo de la Universidad 4, Ciudad Real, 13071, Spain*
[c] *Social Sciences & IT Faculty, University of Castilla-La Mancha, Av. Real Fábrica de Seda s/n, Talavera de la Reina, 45600, Spain*

Abstract

Quantum computing has been growing drastically for the last year due to all the possible applications that this new paradigm brings as well as its incomparable computational power. Therefore, the new information systems that will be developed in a future might be influenced by this paradigm. However, discarding the legacy information systems is not an option if those systems embed mission-critical knowledge over time. Furthermore, quantumfy every business process does not make sense because the high cost that it requires. This is why, in a future, organizations will adapt their classical information systems with new quantum applications, evolving their legacy information systems into hybrid information system. To accomplish this evolution, this paper proposes a technique of software modernization using model-driven engineering based on the Knowledge Discovery Metamodel (KDM) standard.

Keywords 1

Reengineering, Reverse Engineering, Quantum Computing, KDM, Q#

1. Introduction

During the history of technology and science, different theories, paradigms and methodologies have emerged, changing the way of working. Computer science, and so software engineering, is not an exception with the evolution of programming, i.e., develop software nowadays in assembly language is a madness. But, the background of computing is still the same, which is based in the Boolean algebra as one abstraction of the classical binary computers [1].

Quantum computing is a new paradigm which takes advantage of the characteristics of quantum mechanics. The history of quantum mechanics started in 1892 when Gustav Kirchhoff introduced the concept of black body [2], which is a cavity (opaque and non-reflective) that permits the entrance of light but not its exit. He observed that, thought the light could not go out, the black body emitted thermal radiation (heat). This is because the energy of the light is re-emitted all over the cavity in different wavelengths. In 1900, Max Planck Max Planck explained that the energy emitted on a black body can be defined if the light can be emitted and absorbed into a finite number of "packages" of energy localized in a space [3]. Those packages are called "quantums of energy" or photons. His researches started the race of the first quantum revolution, as well to win the Nobel Prize of Physics in 1918.

After several theories and researches, Paul Dirac developed a model which joined the most important equations of the quantum revolution, as well as the theory of the special relativity, describing elemental specific particles [4]. Finally, with John von Neumann, Dirac stablished the Postulates of Quantum Mechanics.

As said before, quantum computing applies the phenomena of the Postulates of Quantum Mechanics to computing. Some of these phenomena are superposition and entanglement. Furthermore, the "quantum of energy" described by Max Planck appears in quantum computing replacing the classical

Q-SET'20: 1st Quantum Software Engineering and Technology Workshop, October 13, 2020, Denver — Broomfield, Colorado, USA

EMAIL: luis.jimeneznavajas@uclm.es (L. Jiménez-Navajas); ricardo.pdelcastillo@uclm.es (R. Pérez-Castillo); mario.piattini (M. Piattini)
ORCID: 0000-0001-6257-7153 (L. Jiménez-Navajas); 0000-0002-9271-3184 (R. Pérez-Castillo); 0000-0002-7212-8279 (M. Piattini)

bit (the smaller unit of information) with qubits, acquiring an exponential growth in computational power.

Although this new technology is not mature enough, the organizations must be prepared to face a modernization towards it. Having in the business processes a computational power for simulation and calculation faster than the actual supercomputer will gave to the organizations a high value and will bring new market opportunities. This modernization does not mean to fully discard the classical information systems, because some of the supported business processes will not be probably supported through quantum computing, and even if those business processes are quantumfied, the gain will be low against the high cost. The same happens with the information systems that embedded a vast amount of mission-critical knowledge during their evolution

The solution of the evolution of the actual systems towards quantum computing could be in the develop of classical-quantum systems, where in this paper have been denominated hybrid information systems. These systems consist of a master classical system that make requests to quantum computers (typically in the cloud) to compute specific quantum algorithms. However, the evolution from classical systems into hybrid has not been treated before and there are no methods for dealing with this problem.

Our proposed solution is based on reengineering, specifically on Architecture-Driven Modernization [5]. This modernization is an evolution of traditional reengineering which follows Model-Driven Engineering (MDE) approach advocating the usage of KDM [6]. KDM ensures the representation, in technology-agnostic way, the architecture of the system including its interrelationships. The abstraction achieved through KDM ensures the interoperability between reverse engineering tools and other software modernization tools.

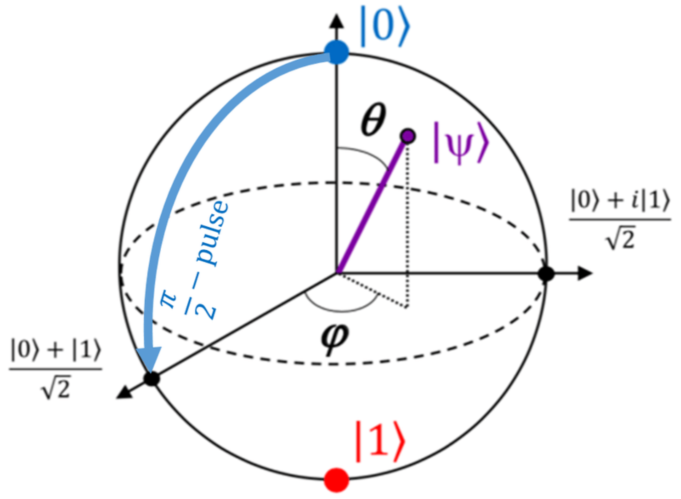
The remaining of the paper is structured as follows: Section 2 explains the state of art of quantum computing and reengineering. Then, Section 3 introduces Quantum Software Reengineering where the proposed technique is framed.

2. State of the Art

2.1. Quantum Computing

Quantum computing started in the eighties when Paul Benioff proposed a quantum mechanical model of the Turing machine [7]. Some years later, the physicists Richard Feynman [8] and Yuri Manin [9] discovered the potential of how a quantum computer can simulate and process than a classical computer cannot. After a few years, the mathematician Peter Shor, developed an algorithm for quantum computers which factors integers, this means that by means of quantum computing, RSA-encrypted communications could be decrypted [10]. It should be noticed that RSA-encryption is used for digital signatures and not forgetting that HTTPS protocol uses this kind of encryption between clients and servers.

Quantum mechanics changes the traditional bit, the smaller unit of information, with qubit (quantum bit). A qubit is usually represented with the electron spin or photons among other subatomic particles. A qubit is a multiple status quantum system (not only defined by zero and one as a classical bit) where there exist infinite possible values, as a sphere (see **Figure 1**). This means that a qubit state might be zero and one at the same time. This phenomenon is known as superposition [11]. Superposition is the secrete of the exponential computational power of quantum computers since n qubits are represented by a superposition state vector in 2^n.


|0⟩
θ
|ψ⟩
$\frac{|0\rangle + i|1\rangle}{\sqrt{2}}$
$\frac{\pi}{2} - pulse$
φ
$\frac{|0\rangle + |1\rangle}{\sqrt{2}}$
|1⟩

$$|\psi\rangle = cos\frac{\theta}{2}|0\rangle + e^{i\varphi}sin\frac{\theta}{2}|1\rangle$$

Figure 1. Representation of a qubit based on the model of Felix Bloch [12]

The qubit's mathematical definition of its state is represented by a vector following a special notation named ket-bra notation. This quantum state is represented with the letter ψ, where $|0\rangle$ and $|1\rangle$ act for depiction of the two different energy levels (which are 0 and 1). The letters α and β indicate the probabilities or proportion of the qubit to be $|0\rangle$ and $|1\rangle$, i.e., $|\psi\rangle = (\alpha\,|0\rangle + \beta|1\rangle)$. Thus, the probabilities are $\|\alpha\|^2$ for $|0\rangle$ and $\|\beta\|^2$ for $|1\rangle$.

Another important quantum phenomenon is the quantum entanglement. This property of quantum mechanics was predicted, not discovered, by Albert Einstein, Boris Podolsky and Nathan Rosen, and first named as EPR Paradox. The paradox suggested that the Laws of Quantum Mechanics were incomplete because until that moment no one could explain how, if two particles that interacted in the past (so, are interlaced), why if you manipulated the state one of those particles, the other's one state also changes instantly, with no apparent connection between them [13]. This new property violated the theory of relativity. After a few years, John S. Bell demonstrated mathematically the EPR Paradox concluding that the quantum mechanic's phenomena cannot be explained in terms of classical physics [14].

Likewise happened in sixties with the space race, nowadays there is a similar race to get new kind of supremacy, called quantum supremacy. This race's goal is to demonstrate that quantum computing can solve a problem that a classical computer cannot[15]. Among many other, the head-to-head competitors of this race are IBM and Google. In fact, Google recently announced that has already achieved the quantum supremacy with a 54-qubit computer [16]. However, participate in this race is difficult due to the actual price of quantum computers, which can ascend up to dozens of millions of euros besides all the maintenance that requires and only a few companies can afford it.

Together with the quantum computers, various quantum programming languages have been developed such as Q# or QASM among others. These programming languages include abstractions for building quantum gates and other quantum operations [13]. Most of these programming languages are open-source and can be used by anyone interested. On the one hand, having these open-source languages encourages the people to contribute on the global knowledge, developing algorithms and new theories. On the other hand, there are not guides of good practices to develop quantum code. There are not guidelines of how quality quantum software must be developed. To alleviate this problem the Talavera Manifesto for Quantum Software Engineering and Programming [14] proposes good practices for the correct development of quantum software.

2.2. Traditional Reengineering

Software technology evolves over time, so information systems can (or should) evolve consequently. This evolution can have negative effects on those systems that were developed in the past, like degradation or aging, making those information systems *legacy*, which means that the source code that was developed could be technologically obsolete [17]. Reengineering allows the preservation of the business knowledge, making possible to carry out evolutionary maintenance of the legacy information systems assuming low risks and low costs [18]. The overall reengineering process is typically presented as a "horseshoe" model [19], see Figure 2, where reengineering consists of three main stages:

1. Reverse engineering: the system is analysed to identify its components and interrelationships and create abstract representations of the system in another form or at a higher level of abstraction.
2. Restructuring: the transformation from one representation form to another at the same relative abstraction level. This stage can consist of refactoring, i.e., the internal structure is improved while preserving the subject system's external behaviour (functionality and semantics). Or additionally, it can add new functionality at this abstraction level.
3. Forward engineering: the final stage consists of the renovation by generating the new source code and other software artefacts at lower abstraction level.

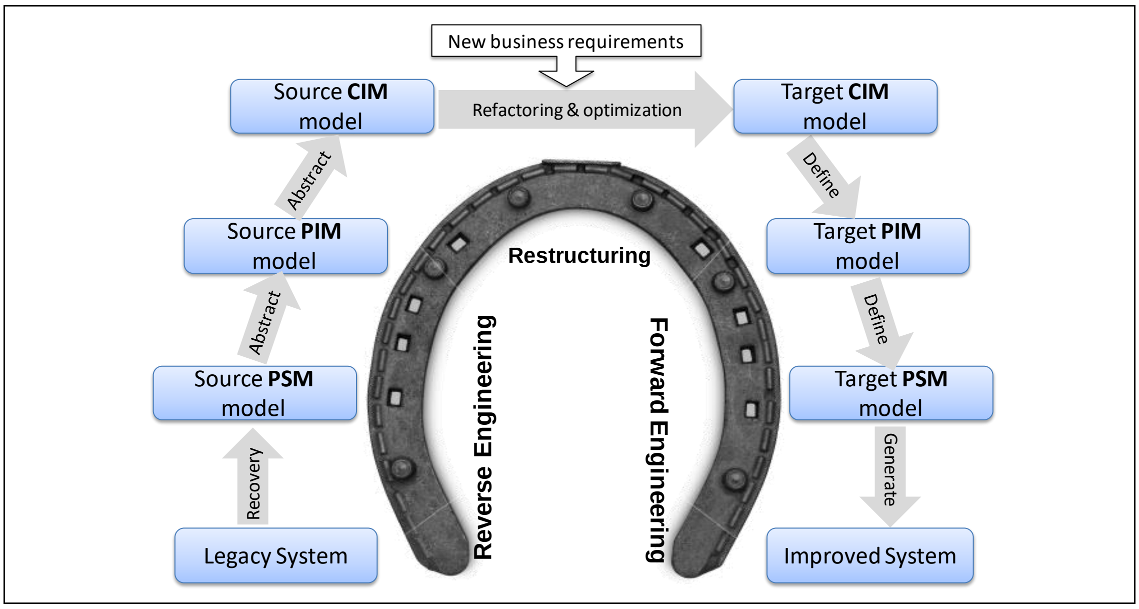


Figure 2. Horseshoe Modernization Model (taken from [20]).

The traditional reengineering projects fail when dealing with specific challenges like the standardization and automation of the reengineering process [21]. First, standardization constitutes a problem since the reengineering process is typically carried out in an *ad hoc* manner. Thus, reengineering projects must focus their efforts on a better definition of the process. Furthermore, the code cannot be the only software asset that the standardization covers, since *"the code does not contain all the information that is needed"* [22]. The reengineering process must be formalized to ensure an integrated management of all of the knowledge involved in the process such as source code, data, business rules, and so on. Second, automation is also a very important problem. In order to prevent failure in large complex legacy systems, the reengineering process must be more mature and repeatable [23]. In addition, the reengineering process needs to be aided by automated tools so that companies can handle the maintenance cost [21].

In order to address the mentioned problems, traditional reengineering evolved toward Architecture-driven Modernization (ADM) [24]. ADM consists of the use of tools that facilitate the analysis, refactoring and transformation of existing system towards a modernization for supporting new requirements, migration of systems or even their interoperability. To accomplish this, ADM makes use of reengineering and Model-Driven Engineering (MDE) [25], where software development's approach is done through defined abstract models and automatic transformation between them.

As a part of ADM initiative, the OMG released the Knowledge Discovery Metamodel (KDM) within a broad set of proposed standards [26]. KDM addresses the main challenges that appear in the modernization of legacy information systems and it is the cornerstone of the set of proposed standards, since the other standards are defined around KDM [6]. KDM uses the OMG's standards for representing the models through XMI.

KDM defines a metamodel which represents the software artifacts which are involved in the legacy information system, providing an accurate view of the functions and structures of it. Reverse engineering techniques use KDM to build high-abstraction level models in a bottom-up manner starting from software legacy artifacts.

The KDM specifies a metamodel to represent legacy knowledge metamodels because KDM is defined as an Entity-Relationship model according to the MOF (Meta Object Facility) meta-model [27] like UML, CWM and SPEM. KDM can be seen as a metamodel to represent platform-independent models. KDM can also work as common interchange format shared for reverse engineering tools, software analysis tools, and any other modernization tool.

3. Quantum Software Reengineering

As said in the introduction, quantum software reengineering might be the solution for dealing with the challenges that the evolution toward hybrid systems bring. Based on software modernization (its evolution adding an MDE approach), quantum software engineering could be used in three complementary scenarios:

1. Migrate existing, isolated quantum algorithms and integrate them into the hybrid information systems.
2. Migrate classical legacy information systems toward hybrid architectures that support the integration of classical-quantum information systems.
3. Transform or add new business operations supported by quantum software that will be integrated into the target hybrid systems.

In Figure 3 is shown the overall quantum reengineering process, where it can be seen that the solution proposed to the evolution towards hybrid systems uses standards as KDM and UML as main cores. The first stage of quantum reengineering is reverse engineering which analyses existing information systems artefacts such as the source code, database schemas, etc. Not only classical information systems (scenario 1) could be inspected but quantum programs (scenario 2) too. The output of this phase is a set of KDM files which represents all the different perspectives and concerns of the legacy information systems, preserving all the business knowledge acquired along the time and reducing the impact on the implantation of quantum programs.

The second stage is restructuring (see Figure 3), where the KDM models are transformed into high-abstraction level models which represents both the analysis and design aspects of the target hybrid system with UML. Once the representations are done, the software engineers could model the target hybrid system.

The last phase consists of forward engineering (see Figure 3), which is composed with a set of techniques that takes the representation done in the previous phase and generate the source code for target hybrid system. Nowadays, exists a vast number of source code generators from UML models. However, there are not generators of quantum source from high level abstraction models.

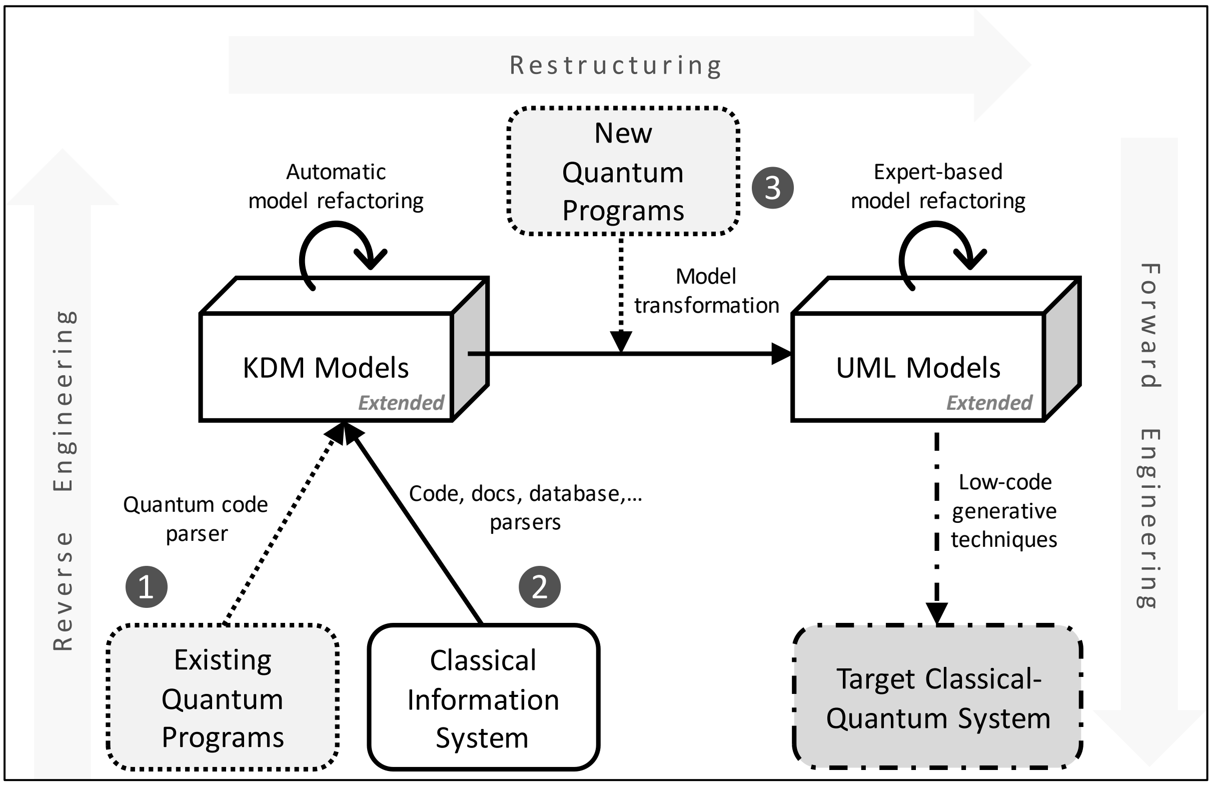

Restructuring
New Quantum Programs
3
Automatic model refactoring
Expert-based model refactoring
Model transformation
KDM Models
Extended
UML Models
Extended
Reverse Engineering
Forward Engineering
Quantum code parser
Code, docs, database,... parsers
Low-code generative techniques
1
2
Existing Quantum Programs
Classical Information System
Target Classical-Quantum System

Figure 3. Quantum reengineering process.

4. Acknowledgements

This research has been partially funded by the G3SOFT (SBPLY/17/ 180501/ 000150), and GEMA (SBPLY/17/180501/000293) projects funded by the 'Direc-ción General de Universidades, Investigación e Innovación – Consejería de Edu-cación, Cultura y Deportes; Gobierno de Castilla-La Mancha'. This work is also part of the projects BIZDEVOPS-Global (RTI2018-098309-B-C31) and ECLIPSE (RTI2018-094283-B-C31) funded by Ministerio de Economía, Industria y Com-petitividad (MINECO) & Fondo Europeo de Desarrollo Regional (FEDER); and SMOQUIN (PID2019-104791RB-I00) funded by Spanish Ministry of Science and Innovation (MICINN).

5. References

[1] Washburn, S.J.E.E., *Boolean algebra in electronic circuit design.* 1954. **73**(2): p. 164-164.
[2] Kirchhoff, G., *Licht; ff.* Ann. Phys. Chemie, 1860. **109**: p. 275.
[3] Planck, M., *Ueber das Gesetz der Energieverteilung im Normalspectrum.* Annalen der Physik, 1901. **309**: p. 553-563.
[4] Dirac, P.A.M., *The Quantum Theory of the Electron.* Proceedings of the Royal Society of London Series A, 1928. **117**: p. 610-624.
[5] Pérez-Castillo, R., I.G.R. de Guzmán, and M. Piattini, *Architecture-driven modernization*, in *Modern Software Engineering Concepts and Practices: Advanced Approaches*. 2011, IGI Global. p. 75-103.
[6] Pérez-Castillo, R., I.G.R. De Guzmán, and M. Piattini, *Knowledge Discovery Metamodel-ISO/IEC 19506: A standard to modernize legacy systems.* Computer Standards and Interfaces, 2011. **33**: p. 519-532.
[7] Benioff, P., *The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines.* Journal of Statistical Physics, 1980. **22**: p. 563-591.

[8] Feynman, R.P., *Simulating physics with computers.* International Journal of Theoretical Physics, 1982. **21**: p. 467-488.
[9] Manin, I.I., *Vychislimoe i nevychislimoe*. 1980.
[10] Shor, P.W., *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer.* SIAM Journal on Computing, 1997. **26**: p. 1484-1509.
[11] Schrödinger, E., *Die gegenwärtige Situation in der Quantenmechanik.* Die Naturwissenschaften, 1935. **23**: p. 807-812.
[12] Bloch, F., *Nuclear induction.* Physical Review, 1946. **70**: p. 460-474.
[13] Einstein, A., B. Podolsky, and N. Rosen, *Can Quantum-Mechanical Description of Physical Reality Be Considered Complete?* 1935. **47**: p. 777.
[14] Bell, J.S., *On the einstein podolsky rosen paradox.* Physics Physique Fizika, 1964. **1**(3): p. 195.
[15] Preskill, J., *Quantum computing and the entanglement frontier.* 2012: p. 1-18.
[16] *Google's Quantum Blog.* 2019.
[17] Ulrich, W.M., *Legacy Systems: Transformation Strategies.* 2002.
[18] De Lucia, A., F. Ferrucci, G. Tortora, and M. Tucci, *Emerging Methods, Technologies, and Process Management in Software Engineering.* Emerging Methods, Technologies, and Process Management in Software Engineering, 2007: p. 1-276.
[19] Kazman, R., S.G. Woods, and S.J. Carriere, *Requirements for integrating software architecture and reengineering models: CORUM II.* Reverse Engineering - Working Conference Proceedings, 1998: p. 154-163.
[20] OMG, *Architecture-Driven Modernization (ADM): Knowledge Discovery Meta-Model (KDM), v1.4. https://www.omg.org/spec/KDM/1.4/PDF*. 2016, OMG. p. 372.
[21] Sneed, H.M., *Estimating the Costs of a Reengineering Project*. Proceedings of the 12th Working Conference on Reverse Engineering. 2005: IEEE Computer Society. 111 - 119.
[22] Müller, H.A., J.H. Jahnke, D.B. Smith, M.-A. Storey, S.R. Tilley, and K. Wong, *Reverse engineering: a roadmap*, in *Proceedings of the Conference on The Future of Software Engineering*. 2000, ACM: Limerick, Ireland.
[23] Canfora, G. and M.D. Penta, *New Frontiers of Reverse Engineering*, in *2007 Future of Software Engineering*. 2007, IEEE Computer Society.
[24] Ulrich, W.M. and P.H. Newcomb, *Information Systems Transformation.* Information Systems Transformation, 2010.
[25] Schmidt, D.C., *Developing Applications Using Model-Driven Design Environments.* IEEE Computer Society, 2006. **39**(2): p. 25-32.
[26] OMG. *Architecture-Driven Modernization Standards Roadmap*. 2009; Available from: https://www.omg.org/adm/ADMTF%20Roadmap.pdf.
[27] OMG, *Meta Object Facility (MOF™) Version 2.0.* 2006.