

Canonical Group Ltd
27 Floor
Millbank Tower
21-24 Millbank
London SW1P 4QP

30 April 2009

**Submission to The Registry of the Enlarged Board of Appeal
European Patent office
Case number G 3/08**

BY Fax and Email

Fax: +49 89 2399 301

Submission by Canonical Group Ltd ("Canonical"), in response to the referral to the Enlarged Board of Appeal concerning the patentability of computer programs ("software")

Canonical is the commercial sponsor of the open source operating system, "Ubuntu", distributing the Ubuntu open source operating system on a free and libre basis, to over 20,000,000 users across the globe and a rapidly growing user base. Ubuntu is available in dozens of languages and runs on almost all laptop, desktop and server computers.

Recently Ubuntu has found great success on small form factor laptops known as 'netbooks' and is the Windows alternative on the most popular models from Dell, Toshiba and HP. Canonical, with a recently opened office in Taiwan, increasingly makes the product available across a range of netbook, notebook and desktop models as a pre-installed operating system through direct contract with the PC manufacturers.

The lack of a licence fee for Ubuntu makes it an attractive choice against Microsoft's or Apple's alternatives but more important to its growing success is the investment which the Ubuntu project has made in making it an attractive, innovative desktop experience that allows users to make the most of web-delivered applications that are the other great source of innovation, inspiration and growth in the software ecosystem.

Canonical commends the decision of Alison Brimelow, President of the European Patent Office in referring the question of patentability in software to the European Patent Office Board of Appeal at a time when inconsistent judicial decisions in particular between the UK Patent Office and the European Patent Office have created a state of confusion and uncertainty. Canonical would encourage a balanced consideration of the responses to the referral and a consideration by the Extended Board, not only the voice of the large corporation(s) but individual developers, represented by organisations such as FFII.

It is Canonical's clear and firm belief that software does not require patents as an additional right in Europe and that use of software patents in practice has been seen to be anti-competitive and stifling to innovation.

== Background ==

Canonical is a young company having been founded in 2004 and has its headquarters in the UK. Canonical currently employs 250 people, around 180 of these being software developers. Ubuntu also benefits from individuals and organisations, not employed or contracted by Canonical, contributing software from the Open Source community. In its support of such open source software projects, Canonical and the developers it works with are at the forefront of software innovation worldwide.

== Open Source==

Open Source projects work in different ways but from a collaborative basis, which encourages constant innovation in a collaborative environment. The collaborative nature of Open Source has been seen to lead to the development of software of an extremely high quality. Software bugs and security problems are fixed quickly in response to community reaction and open source, rapidly creating secure software of the highest calibre.

In the Open Source ecosystem software is produced through community projects and released on very generous, broad licences granting royalty-free copyright usage. The community has self-regulated and originated systems of standardisation including for approval of these licences, creating a simple and understandable ecosystem of approved licence formats and understandable licence grants.

In common with closed source software, sometimes referred to as proprietary software, Open Source software provides binary executables. However, unlike closed source software, Open Source developers provide the human readable source code with their software. For closed source software the "secret sauce" in the intellectual property is hidden in the "closed" source code and the value to the owner of this software is in the hidden portion.

Open Source licences governing the further distribution and modification of open source software often encourage or require that modifications be provided with the source code on the same licence terms. This "openness" creates a cascade effect and access to human readable code which removes the need to re-invent the wheel and gives developers the opportunity to start their work a step ahead.

Open source, by sharing its "secret sauce" offers a head start to innovators and allows them to build on existing layers of code or innovation.

The growth of open source and its uptake in Europe can be seen in a 2008 Gartner report which stated that "85 percent of companies surveyed currently using OSS in their enterprises and the remaining 15 percent expecting to in the next 12 months".

1 Gartner report "User Survey Analysis: Open-Source Software, Worldwide, 2008

This may well be in recognition of the fact that the ensuing cost savings of²FLOSS potentially saves industry over 36% in software R&D investment that can result in increased profits or be more usefully spent in further innovation.

== Intellectual Property in Software ==

In Europe, computer programs, i.e. software, are protected by copyright as a creative work. It is Canonical's clear and firm belief that software does not require further regulation from any other form of intellectual property and that attempts to provide further protection via patents are and have been counter-productive.

Copyright arises at law and does not require registration in Europe. It is a free protection offered to the creative work of software developers and their employers. It is an appropriate way to protect both closed and Open Source software. Money is not a barrier to entry to the reward of intellectual property protection of copyright in software. It is therefore a genuine reward and protection for small companies, community projects and individual developers as well as large corporations, offering everyone automatic protection on a level footing. This is very relevant in Europe as ³"more than three fifths of the worldwide FLOSS developer community live in the EU".

Patents are intended to protect inventions and by doing so, to encourage innovation. Patentable inventions are described in Article 52 of the European Patent Convention which then goes on to make clear that certain things are specifically excluded and cannot attract patent protection:

(1) (2) **The following in particular shall not be regarded as inventions** within the

meaning of paragraph 1:

(2) (a) discoveries, scientific theories and mathematical methods;

(3) (b) aesthetic creations;

(4) (c) schemes, rules and methods of performing mental acts, playing games or doing
business, and **programs for computers**;

2 Study on the:

Economic impact of open source software

on innovation and the competitiveness of the Information and Communication Technologies (ICT) sector in the EU

Final report

Prepared on November 20, 2006

3 Study on the:

Economic impact of open source software

on innovation and the competitiveness of the Information and Communication Technologies (ICT) sector in the EU

Final report

Prepared on November 20, 2006

(5) (c) presentations of information.

(3) **Paragraph 2 shall exclude the patentability of the subject-matter or activities referred to therein only to the extent to which a European patent relates to such subject-matter or activities as such.**

Article 52(2) clearly states that software methods are not inventions and do not attract the coverage of patent protection, however the EPO has become increasingly focused on the analysis of the use of the words "as such" and whether this in fact should imply that there is some inclusion of computer programs in the patent regime. The phrase "as such" does not have any special meaning in law. It is our view that this is an attempt to analyse wording with a view to achieving software patentability despite the clear intention of the legislation. Article 52 clearly states that computer programs "shall not be regarded as inventions".

In attempting to follow through the logic of the "as such" debate the Board of the EPO has further engaged in a second somewhat artificial analysis, and created a ground for the distinction (as to whether patentability exists) to be made, depending on whether or not the program has a "technical character". It is hard to imagine how software can not have a technical character?

Canonical would respectfully suggest that the wording of Article 52 should be interpreted in a simple fashion as it was probably intended and that this artificial distinction between "software" and "software as such" be removed. In turn this would remove the need for the tenuous further consideration of "technical effect"

There is good reason for this exclusion of computer programs as patents exclude anyone other than their inventor from distributing the patented invention unless they hold a licence. This restrictive protection allows the patent holder to create a monopoly position for the patent owner and this can be seen to be something which is not and would not be beneficial to software innovation.

The debate is well known to the Board and a full analysis of the case law surrounding this unnecessary at this juncture.

== Abuse of Patents for software ==

Sadly, since the introduction of software patents in the US in the 1980s and the creep effect of this into European patent protection of software, the use of patents to "protect" software has developed into an abusive system. The rise of non-practising entities otherwise known as the "patent trolls" and litigation with the clear and sole intention of abusing expensive software patents to enforce market dominance and stop innovation, have become synonymous with software patents.

A number of large and wealthy organisations have used their significant financial means to build up very extensive patent portfolios, in some cases as little more than a deterrent to those who might otherwise seek to enforce patents against them. Although the desire to protect themselves against breach of a software patent is understandable, the problem with

this is its creation of a self perpetuating cycle of patent acquisition and enforcement. This ability to buy a share of the market of course restricts new entrants to the market who may not have the financial acumen to buy their way in. In this way market players in a dominant position are able to abuse the patent system to sustain their dominance rather than compete on a level playing field.

Patent searches are expensive and all but impossible. Patent litigation is notoriously expensive. A proliferation of patent grants in the US in software has made it almost impossible for a developer to carry out appropriate searches for pre-existing patents without a prohibitive expense. Not carrying out the searches leaves a developer in fear of litigation.

A market place in patent enforcement has grown up over a number of years. Initially this was evidenced only through companies' own enforcement teams approaching third parties and requiring them to cease use of software allegedly infringing a patent or to buy a licence. More recently through the patent troll (an agent acting for patent holders using their patents to catch out the unwitting software developer with the threat of litigation for breach of patent) a further cause of fear and prohibition to innovation has developed on the back of software patents in what must be seen as little more than a protectionist racket.

Spoiling tactics by companies in the US, such as instructing almost every patent agent in the US to ensure that they are conflicted, due to complex conflict of interest rules applicable to patent attorneys in the US, have also been used by some owners of software patents. An abusive tactic restricting other's ability to obtain appropriate representation and evidence, perhaps, of the flood of abuse opened by software patents.

Such abuse of the patent system - a system set up to reward and protect innovation - is creating a culture of fear amongst both software developers and users. This fear cannot promote technology or encourage innovation.

== Legislation ==

The proposed Software Patent Directive 2002/0047/COD did not succeed in the legislative process. Whilst the output from this referral should seek to achieve clarity it should be noted that it would not be appropriate for this to usurp the legislative process.

Canonical requests that the Extended Board to consider its answers to the Referral questions in the context of the above.

= Answers to Questions =

1. Can a computer program only be excluded as a computer program as such if it is explicitly claimed as a computer program?

Answer: No.

The exclusion of software from patentable subject matter is complete and artificial analysis such as that of "as such" and "technical effect" should not determine whether a computer

program is patentable.

2.(a) Can a claim in the area of computer programs avoid exclusion under Art. 52(2)(c) and (3) merely by explicitly mentioning the use of a computer or a computer-readable data storage medium?

Answer: No.

Whether a patent may be granted is based on whether the innovative and non-obvious aspects of the application lie within patentable subject matter, i.e. the software or computer program. Use of a computer or other hardware or accessory does not make innovations technical, so whether use of this is mentioned or not is not relevant to the decision.

2(b) If question 2(a) is answered in the negative, is a further technical effect necessary to avoid exclusion, said effect going beyond those effects inherent in the use of a computer or data storage medium to respectively execute or store a computer program?

Answer: No.

This analysis is artificial and inappropriate.

If the answer to this question was to be Yes, then the criteria mentioned in the second half is too narrow and for this to amount to an innovation there would be a need for a further requirement ensuring that known uses of existing hardware, or uses of known things for their intended purpose. i.e. using speakers to produce sound or a screen to display things should not allow a software innovation to become "technical".

3. (a) Must a claimed feature cause a technical effect on a physical entity in the real world in order to contribute to the technical character of the claim?

Answer: Yes on condition that there are further requirements, see 3(b)

A technical effect is not an appropriate basis for this, see 2(b), above.

The requirement of a technical effect is, in any event, an insufficient basis. The question which should be answered is rather whether there is a computer program and if this is what the claim relates to?

3.(b) If question 3(a) is answered in the positive, is it sufficient that the physical entity be an unspecified computer?

Answer: No.

3.(c) If question 3(a) is answered in the negative, can features contribute to the technical character of the claim if the only effects to which they contribute are independent of any particular hardware that may be used?

Answer: No

As set out above the basis for acceptance of a patent claim ought not to be whether there is a technical character.

4. (a) Does the activity of programming a computer necessarily involve technical considerations?

Answer: No.

Writing software can be complex, just as writing a novel or composing music, but none of these acts are "technical" in the sense of patent law hence the protection of software under copyright as an artistic work. We assume that "programming" means writing software in the context of this question.

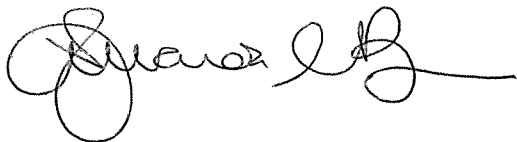
4. (b) If question 4(a) is answered in the positive, do all features resulting from programming thus contribute to the technical character of a claim?

Answer: The answer to 4(a) is No, however if it was Yes then this would be irrelevant to the technical character of a claim.

4. (c) If question 4(a) is answered in the negative, can features resulting from programming contribute to the technical character of a claim only when they contribute to a further technical effect when the program is executed?

Answer: No

However, this question relies on inappropriate analysis of "technical effect". We would commend a simple approach using the clear language in Article 52 and avoiding any artificial analysis.

A handwritten signature in black ink, appearing to read 'Amanda Brock', with a long horizontal flourish extending to the right.

Amanda Brock
General Counsel
Canonical