

Über die Patentprüfung von Programmen für Datenverarbeitungsanlagen

Dr. Swen Kiesewetter-Köbinger*

2. November 2000

Zusammenfassung	3	Der Technikbegriff	8
Dieser Artikel soll einige der Probleme und Ungereimtheiten verdeutlichen, mit denen sich der Autor als ehemaliger Software-Entwickler und jetziger Patentprüfer bei der Prüfung programmbezogener Lehren konfrontiert sieht.		3.1 Sprachanalyseeinrichtung und Logikverifikation	9
		3.2 Die Glocke	10
		3.3 Zusätzlicher technischer Effekt	10
		3.4 Programme zur Hardwarebeschreibung	11
		3.5 Grenzen des Technikbegriffs	11
Inhaltsverzeichnis	4	Der objektive Schutzgegenstand	12
1 Ein Programm als solches	2	4.1 Offenbarung des Programmcodes	13
1.1 Der objektive Erfindungsgegenstand	3	4.2 Patentierung von Lösungen	13
1.2 Erfindungen müssen Lösungen sein	3	4.3 Programme als Lösungen . .	14
		4.3.1 Idee und Lösung . .	15
2 Der technische Charakter	5	5 Der Ausschlußkatalog als gesetzlicher Disclaimer	16
2.1 Die Grundannahme	6	6 Quellcode im Patentanspruch	18
2.2 Der Umkehrschluss	6	6.1 Klarheit von Quellcode . . .	18
2.3 Ausschluß auch technischer Lehren	7	6.2 Die wesentlichen Merkmale .	19
2.4 Technizität von Computerprogrammen	7	6.3 Schutzbereich von Patentansprüchen mit Quellcode . . .	19
		7 Schlussbemerkung	20

*Patentprüfer des Deutschen Patent- und Markenamtes, Abteilung 1.53. Der Beitrag gibt die private Meinung des Verfassers wieder.

1 Ein Programm als solches

In einer seiner jüngsten Entscheidungen¹ hatte der 17. Senat des Bundespatentgerichts darüber zu entscheiden, ob folgende Gegenstände patentfähig sind:

„Digitales Speichermedium, insbesondere Diskette, mit elektronisch auslesbaren Steuersignalen, die so mit einem programmierbaren Computersystem zusammenwirken können, daß ein Verfahren nach einem der Ansprüche 1 bis 17 ausgeführt wird.“

„Computer-Programm-Produkt mit auf einem maschinenlesbaren Träger gespeichertem Programmcode zur Durchführung des Verfahrens nach einem der Ansprüche 1 bis 17 wenn das Programmprodukt auf einem Rechner abläuft.“

„Computer-Programm mit Programmcode zur Durchführung des Verfahrens nach einem der Ansprüche 1 bis 17 wenn das Programm auf einem Computer abläuft“

Im Patentanspruch 1, auf den sich die genannten Patentansprüche beziehen, wurde ein

„Verfahren zur computergestützten Suche und/oder Korrektur einer

fehlerhaften Zeichenkette F_i in einem digital gespeicherten Text, der die entsprechende fehlerfreie Zeichenkette S_i enthält,

dadurch gekennzeichnet, daß

- a) die Auftretenshäufigkeit $H(S_i)$ der fehlerfreien Zeichenkette S_i ermittelt wird,*
- b) die fehlerfreie Zeichenkette S_i nach einer Regel R_j verändert wird, so daß eine mögliche fehlerhafte Zeichenkette f_{ij} erzeugt wird,*
- c) die Auftretenshäufigkeit $H(i_j)$ der Zeichenkette f_{ij} in dem Text ermittelt wird,*
- d) die Auftretenshäufigkeiten $H(i_j)$ und $H(S_i)$ verglichen werden und*
- e) basierend auf dem Vergleich in Schritt d) entschieden wird, ob die mögliche fehlerhafte Zeichenkette f_{ij} die gesuchte fehlerhafte Zeichenkette F_i ist.“*

beansprucht, und vom Patentprüfer als patentfähig gemäß den §§ 1-5 PatG erachtet. Dies impliziert nach derzeitigem Verständnis zumindest stillschweigend, dass der Anmeldungsgegenstand technischen Charakter besitzt. Der 17. Senat hat hiergegen keine Bedenken.

Die beanspruchten Gegenstände sollen die Aufgabe lösen, *„ein verbessertes Verfahren und Computersystem zur Suche und/oder Korrektur einer fehlerhaften Zeichenkette in einem Text zu schaffen“*.²

¹Bundespatentgericht 17 W (pat) 69/98

²DE 43 23 241 A1, Seite 2, Zeilen 35-36

Die Frage, worin denn objektiv die Erfindung zu sehen ist, ob in dem Verfahren des Patentanspruchs 1, dem Programm des Patentanspruchs 24, oder aber einem dem Patentanspruch 1 zugrundeliegendem Algorithmus, wurde leider nicht gestellt. Da diese Frage gar nicht aufgeworfen wurde, wird der BGH hierzu vermutlich nicht explizit Stellung nehmen. Dies könnte jedoch ein entscheidender Punkt zum Verständnis der Problematik der Patentierung von Programmen für Datenverarbeitungsanlagen generell sein.

1.1 Der objektive Erfindungsgegenstand

Der 17. Senat hat in seiner Entscheidung unter 1.3.2 erklärt:

„Nach ständiger Rechtsprechung ist unter dem Begriff „Erfindung“ eine Lehre zum technischen Handeln zu verstehen, die in Aufgabe und Lösung differenziert ist; die Lehre zum technischen Handeln besteht aus der Lösung eines technischen Problems.“³

Gleich im nächstem Absatz heißt es:

„Eine Ausführung des Verfahrens gelingt nur mit einem Computersystem, das in der Lage ist, die einzelnen Teile der Aufzeichnung quasi vollständig zu interpretieren und dadurch eine Durchführung

³Er bezieht sich hierbei auf Busse, PatG, 5. Aufl. §1 Rdn 18ff, 72 mwN; Benkard, PatG, 9.Aufl, §1 Rdn 40, 60; sowie BGH BIPMZ 1985, 28 - Acrylfasern.

der gewünschten Verfahrensschritte zu bewirken.“

Die „gewünschten Verfahrensschritte“ werden hier ganz klar nicht selbst als Lösung der Aufgabe angesehen. Weiter unten unter 1.4.3 heißt es:

„Ein Potential zur Erzeugung eines technischen Effekts ... kommt ... erst dem Computersystem mit dem vom Speichermedium in den Arbeitsspeicher geladenen Programm, das tatsächlich in der Lage ist, ein ggf technisches Verfahren auszuführen“

zu.

Anhand dieser Ausführungen läßt sich aber nur ein Schluß ziehen: nicht das (Arbeits-)Verfahren stellt die Lösung der Aufgabe und damit den objektiven Anmeldegegenstand dar, sondern einzig das Computersystem mit dem vom Speichermedium in den Arbeitsspeicher geladenen Programm, dessen Aufzeichnung quasi vollständig interpretiert und ausgeführt werden muß. Das Arbeitsverfahren ist lediglich die Wirkung der Ausführung des Programmcodes durch das in den Patentansprüchen 1, 22, 23 und 24 nicht näher charakterisierte Computersystem. Angesichts der Tatsache, dass bei praktisch allen programmbezogenen Anmeldungen diese Verfahren als Wirkungen beansprucht und auch häufig patentiert werden, ist diese richterliche Erkenntnis von besonderer Bedeutung!

1.2 Erfindungen müssen Lösungen sein

Nun lautet der §1 PatG wörtlich: „Patente werden für Erfindungen erteilt, ...“.

Als Lösung des angemeldeten Problems sieht der Senat in seiner Urteilsbegründung ausdrücklich weder das (Arbeits-)Verfahren, das digitale Speichermedium, das Computer-Programm-Produkt oder das Computer-Programm, sondern erst das „Computersystem mit dem vom Speichermedium in den Arbeitsspeicher geladenen Programm, das tatsächlich in der Lage ist, ein ggf technisches Verfahren auszuführen“. Nimmt man §1 (1) in Verbindung mit der Definition „Erfindungen sind Lösungen“ wörtlich, so dürfte allenfalls auf diesen Gegenstand ein Patent erteilt werden, sofern er neu ist, auf einer erfinderischen Tätigkeit beruht und gewerblich anwendbar ist.

Mit welchem Wortlaut ein derartiges Computersystem derzeit beansprucht wird, ist leider noch nicht veröffentlicht. Aus der Offenlegungsschrift⁴ lässt sich ein derartiges Computersystem aber erkennen. Neben den Speichermitteln 1, 3, 4, 10, 12, 14 und 17 sind anscheinend die Mittel 5, 6, 7, 8, 9, 11 und 13 zur Ausführung des Verfahrens wesentlich. Hierzu wird beschrieben:

„Die in den Prozessormitteln 2 beinhalteten Mittel 3, 5, 6, 7, 8, 11 und 13 sowie der Bus 16 müssen nicht als diskrete elektronische Bauteile ausgeführt sein, sondern können durch eine entsprechende Programmierung des Prozessors 2 erzeugt werden. Ein entsprechendes zur Realisierung des erfindungsgemäßen Verfahrens geeignetes Programm wird dabei mit dem Betriebsprogramm des Computersystems in an sich bekannter

Weise so zusammenwirken, daß das Computersystem die in Fig. 4 gezeigte Konfiguration annimmt.“

Das beanspruchte Arbeitsverfahren soll demnach vorrangig durch ein geeignetes Programm realisiert werden, das von einem Prozessor 2 abgearbeitet wird und dabei über den Bus 16 auf dessen Speicher 1, 3, 4, 10, 12, 14 und 17 zugreift. Computersysteme mit einem Prozessor, welcher über einen Bus auf seine Speicher zugreift und ein Programm aus dem Speicher abarbeitet, sind spätestens seit von Neumann bekannt. Die übrigen Merkmale (Aufteilung des Speichers und *Mittel zum ...*) sind demnach – soweit nachvollziehbar offenbart – durch das Programm im Speicher 17 bewirkt. Ein Programm für eine Datenverarbeitungsanlage wird nach §1 (2) 3. PatG aber nicht als Erfindung angesehen. Wenn aber das Programm selbst auf keiner erfinderischen Tätigkeit beruht, wie kann dann seine Ausführung auf einem bekannten Computersystem zu einer erfinderischen Wirkung in Form eines erfinderischen Verfahrens führen?

Eine zweite Lösungsvariante ist in obigem Zitat aber noch mit aufgeführt, da die Mittel 3, 5, 6, 7, 8, 11 und 13 auch als diskrete elektronische Bauteile ausgeführt sein können. Über die Beschaffenheit diese Bauteile lässt sich kein Hinweis finden. Selbst ihre genaue Funktionalität erscheint dem Autor unklar. Über diese ließe sich erst nach Kenntnis des zu bearbeitenden Programmcodes etwas sagen. Zwar ist bekannt, wie man mit bekannten IP-Blöcken z.B. über Rapid Silicon Prototyping⁵ ASICs entwickelt, jedoch müsste man hierzu eine Zu-

⁴DE 43 23 241 A 1, Fig. 4 mit Beschreibung
Seiten 9-10

⁵Richard Forster „Wie man IP na HDLi anpaßt“
in elektronik industrie 3-1999, 52-55

ordnung zwischen der gewünschten Funktionalität der Mittel 3, 5, 6, 7, 8, 11 und 13 und bekannten Logikblöcken erkennen können. Dem Autor ist dies anhand der Offenlegungsschrift nicht möglich.

Als einigermaßen erkennbare Hardwarelösung bleiben folglich nur mehr die Merkmale eines von Neumann Computersystems mit nicht näher definiertem Prozessor, Speicher und Bus offenbart, bei dem die Funktionalität des Prozessors ausschließlich durch die Ausführung eines geeigneten Programms erzeugt wird. In der Offenlegungsschrift ist zwar kein Programmcode offenbart, jedoch erscheint das angegebene Ausführungsbeispiel der Tippfehlerkorrektur nach Ansicht des Autors in einer objektorientierten Programmiersprache wie C++ mit Template-Klassen der Standard Template Library (STL) für die Instanzen des „linked tree“ durchaus programmierbar. Dieses Programm als solches ist nach §1 (3) in Verbindung mit §1 (2) 3. nicht patentierbar. Das Programm alleine kann aber die Aufgabe auch nicht lösen. In Verbindung mit dem Computersystem, das es ausführen soll, wird zwar kein Programm als solches beansprucht, jedoch kann es nach §1 (2) 3. dem Computersystem keine erfinderischen Merkmale hinzufügen.

Man möchte daher annehmen, dass nach dem Wortlaut des Ausnahmekatalogs des §1 PatG ein derartiger Gegenstand mangels erfinderischer Tätigkeit nicht patentfähig wäre.

2 Der technische Charakter

Als Ausweg aus dem Ausnahmekatalog des §1 PatG hat die Rechtsprechung jedoch Wege gefunden, wie trotz der Eindeutigkeit des §1 PatG explizit ausgenommene Gegenstände dennoch einer Patentierung zugänglich gemacht werden können.

Zu §1 Abs 2 und 3 PatG begründet hier der 17. Senat⁶

„Dieser sog Ausschlußkatalog beruht nach allgemeiner Auffassung auf dem Gedanken, daß den dort genannten Gegenständen der erforderliche technische Charakter fehlt (vgl Busse, PatG, aaO, §1 Rdn 37; Schulte, PatG, 5. Aufl, §1 Rdn 60, jeweils mwN; auch die Gesetzesbegründung, BIPMZ 1976, 332 li Sp, weist in diese Richtung, wenn es dort ua heißt, daß der Negativkatalog lediglich Gegenstände ausschließe, die in der „höchstrichterlichen Rechtsprechung bisher nicht als Erfindungen anerkannt sind (Computer-Programme)“; vgl auch EPA, BK 3.5.1 „Computerprogrammprodukt/IBM“, GRUR Int 1999, 1053, 4.1)“.

Welche Interessensgruppe diese allgemeine Auffassung trägt, wird nicht thematisiert. Der betroffene Kreis der Softwareentwickler scheint jedoch nur zu einem geringen Prozentsatz diese Auffassung zu teilen. Bei den Computerprogrammen gibt es sogar bei

⁶17 W (pat) 69/98

bekannten Patentfachleuten völlig konträre Meinungen⁷.

2.1 Die Grundannahme

In der Grundannahme wird behauptet, daß „nach allgemeiner Auffassung“ allen in dem Ausschlusskatalog des §1 Abs 2 genannten Gegenständen der „erforderliche technische Charakter fehlt“.

Anstatt nun zu beurteilen, ob es sich bei den Anmeldungsgegenständen um einen der in dem Ausschlusskatalog genannten Gegenstände handelt, wird in der täglichen Praxis nur mehr vereinfachend untersucht, ob der Anmeldungsgegenstand „technischen Charakter“ aufweist. Im §1 PatG wird das Wort Technik vom Gesetzgeber überhaupt nicht erwähnt. Ein mittelbarer Bezug ergibt sich erst über den Stand der Technik der Folgeparagrafen 3 und 4 PatG, woraus zu schließen ist, dass damit auch eine Erfindung im Sinne des §1 PatG notwendigerweise technischen Charakter aufweisen muss.

Bei zahlreichen Anmeldungen programmbezogener Lehren ist nun die Justiz zu dem Schluß gekommen, daß diese Lehren doch einen technischen Charakter aufweisen⁸. Ganz offensichtlich gibt es folglich Gegenstände in dem Ausschlusskatalog des §1 Abs 2 welche einen technischen Charakter aufweisen.

Nach allen Regeln der Logik müßte man an dieser Stelle erkennen, daß damit die

⁷vgl. Melullis GRUR 1998 843, der den ausführbaren Programmen technischen Charakter zuschreibt und Tauchert GRUR 1999 829, der dies bei dem Quelltext der Programme und dem ausführbaren Binärcode anscheinend verneinen möchte

⁸vgl. z.B. Schulte 5. Auflage, §1 Rdn 77 mwN aber auch X ZB 11/98 - Logikverifikation

Grundannahme widerlegt ist, da sie in dieser allgemeinen Fassung Ausnahmen aufweist. Der technische Charakter des Anmeldegegenstands kann somit nur ein **notwendiges**, nicht ein **hinreichendes** Merkmal für die Patentfähigkeit darstellen.

2.2 Der Umkehrschluss

Statt dessen wird aber der Umkehrschluss gezogen, dass auch bei den in §1 Abs 2 genannten Gegenständen eine Erfindung und damit Patentfähigkeit vorliegen kann, wenn sie eine technische Lehre enthalten⁹. Die Grundannahme wurde durch zahlreiche Entscheidungen des BGH und des BPatG widerlegt, wie gerade zuvor aufgezeigt wurde. Jede Schlussfolgerung, die auf einer fehlerhaften Grundannahme beruht, kann aber nur selbst fehlerhaft sein. Zwischen **notwendig** und **hinreichend** wird plötzlich nicht mehr unterschieden; ganz im Sinne des Art 27 Abs 1 TRIPS, dessen Anwendbarkeit noch zu klären wäre¹⁰. Für einen Programmentwickler, der durch die tägliche Praxis gezwungen ist, streng logisch zu denken, ist die Argumentation zur hinreichenden Patentfähigkeit technischer Lehren damit nicht nachvollziehbar. Die alleinige Beschränkung auf technische Lehren steht so nicht im §1 PatG, und die Argumentation, die versucht, den Ausschlusskatalog des §1 PatG darauf zu beschränken, ist schlicht falsch.

⁹vgl. Melullis GRUR 1998, 843 844

¹⁰vgl 17 W (pat) 69/98 1.4.5

2.3 Ausschluß auch technischer Lehren

Die logische Folgerung hieraus wäre, daß auch technische Lehren vom Patentschutz ausgeschlossen sein können, wenn sie unter den Negativkatalog fallen. Melullis führt hierzu aus:

„Der gesetzliche Tatbestand ist so formuliert, daß er nach Art einer Fiktion sogar eine technische Lehre vom Patentschutz ausschließen kann, wenn sie unter den Negativkatalog des Art. 52 Abs. 2 EPÜ fällt. Geht man vom reinen Wortlaut des Gesetzes aus, liegt es näher, den Grund für den Ausschluß nicht darin zu sehen, daß den dort genannten Gegenständen ein technischer Inhalt fehlt, sondern darin, daß sie aus der Sicht des Gesetzes einen Schutz durch das Patentrecht nicht verdienen oder diesem aus übergeordneten Gründen nicht unterstellt werden sollen.“¹¹

Genau hierin liegt der Punkt! Der Gesetzgeber hat als Kriterium gegen die Patentfähigkeit nicht auf das Fehlen einer **technischen** Lehre abgezielt, denn dieses wäre sehr einfach zu formulieren gewesen. Vielmehr hat er diese Gegenstände gerade als solche vom Patentschutz ausnehmen wollen. Auch hat der Gesetzgeber mit keinem Wort technische **Lehren** dem Patentschutz zugänglich gemacht, denn eine Lehre ist eine Regel für eine gedankliche Tätigkeit und damit ebenso wie Programme für Datenverarbeitungsanlagen nicht als Erfindung angesehen¹².

¹¹GRUR 1998, 843 851

¹²§1 (2) 3. PatG

Melullis nennt dies allerdings eine Fiktion. Nach dem üblichen Verständnis von einem Rechtsstaat ist es Sache des Gesetzgebers, Gesetze allgemeingültig zu erlassen; die Verwaltung hat diese Gesetze im Einzelfall anzuwenden; die Justiz hat für jede einzelne Verwaltungsmaßnahme einzeln zu entscheiden, ob die jeweilige Ausführung gesetzeskonform war. Wie der Wortlaut des Gesetzes in einem Rechtsstaat als Fiktion angesehen werden kann, und Justizentscheidungen in der Folge den Wortsinn des Gesetzes ins Gegenteil verwandeln können, ist für den Autor völlig unverständlich.

Aber ohne diese ganzen Konstruktionen wären Computerprogramme als solche sowie neue Anwendungen bekannter Gegenstände mit neuen Programmen nicht patentierbar.

2.4 Technizität von Computerprogrammen

Bei seiner Betrachtung geht Melullis „grundsätzlich von der Technizität jedes Computerprogramms“ aus. Genauer gesagt differenziert er zwischen

- „dem gedanklichen logischen Konzept ..., das der durch die Software zu lösenden Aufgabe zugrunde liegt, d.h. dem eigentlichen Programminhalt einschließlich der ihm zugrundeliegenden Programmidee und
- deren Umsetzung in die zur Ausführung erforderlichen einzelnen Schritte.“

„Computerprogramm als solches ist bei diesem Verständnis mithin

das außertechnische Konzept; d.h. die der Umsetzung in eine Handlungsanweisung an den Rechner vorausgehende Konzeption.“

Dem Autor ist bislang jedoch noch keine einzige deutsche oder europäische Patentschrift über programmbezogene Lehren begegnet, in der nicht gerade diese **Konzeption** als **Wirkungsanspruch** in Form eines „Verfahren zum ...“ beansprucht worden wäre.

Für Melullis ist demgegenüber

„Grundsätzlich patentfähig – auch weil technisch – ... die Umsetzung dieser zunächst nur gedanklichen Lösung in eine Handlungsanweisung für den Computer und dessen Ansteuerung, das heißt die der Verwirklichung dieses Programms dienenden technischen Konzeption, wie sie ihren Niederschlag in dem fertigen Programm gefunden hat.“

Das fertige Programm ist für ihn somit kein Programm als solches mehr und damit dem Patentschutz grundsätzlich zugänglich. Die Konzeption scheint auch er als einen Plan für eine gedankliche Tätigkeit anzusehen. Andere Autoren¹³ bezeichnen aber gerade diese codierten Befehlsfolgen als Programm als solches; haben aber keinerlei Bedenken, die zugrundeliegende Programmidee oder Geschäftsidee trotz ihrer Eigenschaft als Plan für gedankliche Tätigkeit als technisch einzustufen und damit zu patentieren.

¹³Tauchert, Mitt. 1999, 248 oder van Raden, GRUR 1995, 451

Alle diese Schlußfolgerungen scheinen aber wieder auf der unrichtigen Grundannahme über den Ausschlußkatalog zu beruhen.

3 Der Technikbegriff

Melullis¹⁴ verwendet eine häufig anzutreffende Argumentationslinie, indem er dem Begriff „Computerprogramm als solches“ eine eigene Definition gibt als „*außertechnisches Konzept; d.h. die der Umsetzung in eine Handlungsanweisung an den Rechner vorausgehende Konzeption*“. Nun ist das Gebiet der Technik aber faktisch nicht eingrenzbar. Der jeweilige Fachmann spricht iA von Programmierertechnik, Maltechnik, Vortragstechnik, Fußballtechnik, Redetechnik, Kulturtechniken (Lesen, Schreiben, Rechnen), Lauftechnik, Grifftechnik zum Spielen eines Musikinstruments. Selbst beim Geschlechtsakt wird gelegentlich von besonderen Techniken gesprochen. Im Brockhaus¹⁵ beginnt die umfangreiche Definition von Technik mit

„[von technica im internat. Gelehrtenlatein, dies aus grch. technē >Kunst<, >Fertigkeit<] hatte im alten Griechenland die Bedeutung von Kunst, Gewerbe, Handwerk, auch Wissenschaft und Kunsthandwerk sowie Kunstfertigkeit, etwas bestimmtes zu erreichen (z.B. T. der Malerei). I. e. S. ist T. heute konstruktives Schaffen von Erzeugnissen, Vorrichtungen

¹⁴aaO

¹⁵„Der Grosse Brockhaus“, Kompaktausgabe, 18. Auflage 1983

gen und Verfahren unter Benutzung der Stoffe und Kräfte der Natur und unter Berücksichtigung der Naturgesetze. ...“

Der BGH definiert:

„Technisch ist eine Lehre zum planmäßigen Handeln unter Einsatz beherrschbarer Naturkräfte zur Erreichung eines kausal übersehbaren Erfolgs, der ohne Zwischenschaltung menschlicher Verstandestätigkeit die unmittelbare Folge des Einsatzes beherrschbarer Naturkräfte ist.“¹⁶

3.1 Sprachanalyseeinrichtung und Logikverifikation

Dass sich auch der Technikbegriff des BGH ändert, verdeutlicht folgendes: Anweisungen an den menschlichen Geist wurden bislang vom BGH immer als nicht technisch zurückgewiesen. Seit der Entscheidung „Sprachanalyseeinrichtung“¹⁷, in welcher der Senat feststellt,

„daß vom menschlichen Verstand Gebrauch gemacht werden kann, ohne daß allein dadurch der Bereich des Technischen bereits verlassen wird, ergibt sich schon daraus, daß dem Patentschutz Lehren zum planmäßigen Handeln unter Einsatz beherrschbarer Naturkräfte zur Erreichung eines kausal übersehbaren Erfolgs zugänglich sind¹⁸.“

¹⁶Schulte 5. Aufl §1 Rdn 24 mwN

¹⁷X ZB 15/98

¹⁸unter Verweis auf BGHZ 53, 74, 79 - rote Taube

hat der BGH Anweisungen an den menschlichen Geist als nicht unbedingt patenthindernnd bezeichnet und für den Erfolg auf die **Unmittelbarkeit** des Einsatzes beherrschbarer Naturkräfte verzichtet. Auch in der Entscheidung „Logikverifikation“¹⁹ – bei der nurmehr gefordert wird, dass

„Wenn eine Lehre für ein Programm für Datenverarbeitungsanlagen durch eine Erkenntnis geprägt ist, die auf technischen Überlegungen beruht, ist mithin ein auch anderweit akzeptiertes und eine einheitliche Patentrechtspraxis für Europa förderndes Abgrenzungskriterium gegeben, das die Feststellung des erforderlichen technischen Charakters einer Lehre für ein Programm für Datenverarbeitungsanlagen erlaubt.“ –

hat der BGH den Einsatz beherrschbarer Naturkräfte als **unmittelbare Voraussetzung** der Patentfähigkeit verworfen und dies damit gerechtfertigt, dass

„der Technikbegriff des Patentrechts nicht statisch, das heißt ein für allemal feststehend verstanden werden kann. Er ist vielmehr Modifikationen zugänglich, sofern die technologische Entwicklung und ein daran angepaßter effektiver Patentschutz dies erfordern²⁰.“

Mit einiger Böswilligkeit lassen sich diese beiden Entscheidungen auch so verstehen,

¹⁹X ZB 11/98

²⁰unter Berufung auf BGHZ 52, 74, 76 - Rote Taube.

dass selbst die Anweisung an den Programmierer, ein Programm zu schreiben, das auf technischen Überlegungen beruht, vom Senat als technisch angesehen wird. Eine sehr anmelderfreundliche Interpretation zur Zusammenschau beider BGH-Entscheidungen findet sich gleichfalls bei Hössle²¹, der in „dem „nicht-statischen Technikbegriff“ eine würdige und griffige Verallgemeinerung (sieht), die in der Praxis gute Dienste leisten wird.“

3.2 Die Glocke

Ein anderes böswilliges Beispiel: Für Melullis ist grundsätzlich von der Technizität jedes Computerprogramms auszugehen, da das Programm den Computer steuert. Nun werden mit Hilfe von Programmen auch technische Informationen wiedergegeben²², die auf technischen Überlegungen beruhen²³. Der Unterschied, ob es sich um eine technische CAD-Zeichnung im Quellcode mit Steueranweisungen für ein CAD-Programm oder Druckprogramm handelt, oder aber um die in HTML geschriebene Darstellung von Schillers „Die Glocke“²⁴, mit Steueranweisungen für ein Browserprogramm, ist dabei eher akademischer Natur, denn beide enthalten Steueranweisungen, die von einem Rechner mit dem dafür vorgesehenen Programm zur Interpretation der Steueranweisungen ausgeführt werden können; selbst die wiedergegebene Information²⁵ ist jeweils technisch, auch wenn der metallurgische Aspekt bei Schiller in keiner

für den Glockengießer üblichen Sprache offenbart wurde.

3.3 Zusätzlicher technischer Effekt

Mit Hilfe von Programmen steuert jeder Rechner seine Peripherie. Es ist dabei vom Gesichtspunkt des Programmierers nebensächlich, ob diese Peripherie ein Bildschirm, ein Drucker, eine I/O-Karte, ein oder mehrere A/D- D/A-Wandler oder ein sonstiger für die Ansteuerung durch einen Prozessor gedachter Gegenstand ist. Mit einem Programm in Assemblersprache beschreibt man die Ansteuerung der zugrundeliegenden Hardware am unmittelbarsten; jeder einzelne Assemblerbefehl beschreibt und erzeugt bei seiner Ausführung einen unmittelbar technischen Effekt²⁶. Unter einem zusätzlichen technischen Effekt, der über das übliche Zusammenwirken zwischen Programm und Computer hinausgeht – wie er neuerdings von den Technischen Beschwerdekammern des EPA gefordert wird²⁷ – läßt sich dem gegenüber genau genommen nur mehr ein Effekt verstehen, der durch keinen der ausgeführten Programmschritte hervorgerufen wird, von der programmbezogenen Lehre folglich unabhängig ist und mit ihr nicht im Zusammenhang steht. Wieso ein solcher zusätzlicher technischer Effekt dann überhaupt mit einem Programm in Zusammenhang zu bringen ist, bleibt unklar. Wird der besagte technische Effekt jedoch durch die Ausführung eines oder mehrerer Programmschritte hervorgerufen, so läßt sich

²¹Mitt. 2000 343-346

²²§1 (2) 4. PatG

²³X ZB 11/98 - Logikverifikation

²⁴<http://www.fulgura.de/etc/glocke.htm>

²⁵§ 1 (2) 4. PatG

²⁶vgl. hierzu Melullis GRUR 1998 843 850 3a)

²⁷T 1173/97 - Computerprogrammprodukt und T 935/97 - Computer program product II

mit keiner Berechtigung mehr von einem *zusätzlichen* Effekt sprechen, da dies ja der beabsichtigte Effekt ist.

3.4 Programme zur Hardwarebeschreibung

VHDL- oder VERILOG-Programme beschreiben den Aufbau von hochintegrierten Schaltkreisen und steuern die Konfiguration von FPGAs (Field Programmable Gate Arrays)²⁸. Diese Schaltkreise werden über diese Programme entworfen, mit Hilfe weiterer Programme verifiziert und auf entsprechenden Logiksimulatoren getestet. Die Masken für die Waferproduktion werden auch von Programmen erzeugt, welche die VHDL- bzw. VERILOG-Programme auswerten²⁹. Der fertige IC, Prozessor, ASIC ist letztlich nichts anderes mehr als ein Produkt mehrerer Computerprogramme, bei dem die Lösung, wie die gewünschte Funktionalität herzustellen ist, letztlich einzig auf den in der gewählten Hardwarebeschreibungssprache geschriebenen Programmen beruht³⁰. Derartige Programme beinhalten sicherlich technische Überlegungen, da diese Programme ja konkrete Hardware beschrei-

²⁸vgl. z.B. die unzähligen IP-Core-Angebote der FPGA-Hersteller XILINX, ALTERA und Lattice/Vantice oder auch Richard Yuan, Neue Lösungen für CompactPCI-Anwendungen, Elektronik Informationen, 2-2000, 110-112; Ulf Pansa, Requirement-Engineering im Bereich Automotive, Elektronik Informationen 8-2000, 54-56; Rainer Kress et al., Java-basiertes Codesign, Elektronik Praxis, 8. Februar 2000, 110-114

²⁹vgl. X ZB 11/98 „Logikverifikation“ 4h) mit Verweis auf Schmidtchen, Mitt. 1999, 282,291 f.

³⁰Dave Bursky, Leveraging Intellectual Property Gets Easier As Standards Make Headway, Electronic Design, 26. Juni 2000, 91-102

ben und mit ihnen konkrete Halbleitertopographien erzeugt werden können. Aber fast niemand meldet den Code dieser Programme zum Patent an – obwohl deren eindeutige Befehlsabfolge als VHDL- bzw. VERILOG-Programm zum Verkauf angeboten wird und damit vollständig offengelegt ist – sondern fast ausschließlich deren gewünschte Funktionalität (also eine irgendwie zu lösende Aufgabe, siehe weiter unten).

3.5 Grenzen des Technikbegriffs

Letztendlich läßt sich demnach je nach Zeitgeist alles von Menschen gemachte als „*technisch*“ verstehen³¹. Abzugrenzen hiervon wäre allenfalls die göttliche Schöpfung. Über jegliche andere Abgrenzung des Technikbegriffs läßt sich nach Belieben streiten, da hierunter ein Maler etwas anderes verstehen wird, wie ein Physiker, ein Musiker etwas anderes wie ein Handwerker, ein Schriftsteller etwas anderes wie ein Ingenieur und ein Informatiker letztlich etwas anderes wie ein Jurist. Eine rein juristisch geprägte Technik-Definition zur Abgrenzung von Patentfähigem gegen Nichtpatentfähiges, welche den Technikbegriff der jeweiligen Zielgruppe unberücksichtigt läßt, erscheint vor diesem Hintergrund willkürlich. Als notwendiges und zugleich hinreichendes Kriterium für die Entscheidung ob ein Gegenstand patentfähig sein soll oder nicht, erscheint der Technikbegriff daher ungeeignet, da dem Gebot der Rechtssicherheit ab-

³¹Diamond v. Chakrabarty, 447 U.S. 303, 309 (1980); vgl. auch Esslinger, Hössle in Mitt 1999 327-329

träglich. Als ein notwendiges Merkmal neben anderen erscheint es durchaus sinnvoll und mit der langjährigen Rechtstradition des deutschen Patentrechts konform.

4 Der objektive Schutzgegenstand

Einen weiteren Weg zur Patentierung von Software zeigt schon der §1 Abs 3 selbst auf, indem er besagt

„Absatz 2 steht der Patentfähigkeit nur insoweit entgegen, als für die genannten Gegenstände oder Tätigkeiten als solche Schutz begehrt wird.“

Da das Schutzbegehren durch die Patentansprüche (§34 Abs 1 (2) PatG) festgelegt wird, ist die folgerichtige Reaktion, anstelle dieser konkreten Gegenstände davon abgeleitete zu beanspruchen. Anstelle des konkreten Programmes wird nur mehr das von dem Programm in einer Datenverarbeitungsanlage hervorzurufende Arbeitsverfahren als Wirkungsanspruch angemeldet.

Auch der Gegenstand, welcher der Entscheidung 17 W (pat) 69/98 des Bundespatentgerichts im Patentanspruch 1 zugrunde lag, wurde nach diesem Strickmuster generiert. Die Beschreibung des gewünschten Arbeitsverfahrens geht in der Programmentwicklung aber immer der eigentlichen Codierung voran und steht zumeist am Beginn der Designphase. Die Beschreibung des Arbeitsverfahrens entspricht demnach dem von Melullis so genannten „*außertech-*

nischen Konzept“³², dem er keinen Patentschutz zubilligen möchte. Das Konzept für die Erstellung eines Programms ist nichts anderes wie ein Plan für eine gedankliche Tätigkeit³³. Ein Patent hierauf würde es anderen verbieten, in gleicher Art zu denken.

Das beanspruchte Arbeitsverfahren kann auch als Wirkungsanspruch mit ausschließlich funktionalen Merkmalen gewertet werden. Hierzu sagt Schulte³⁴:

„Funktionelle Merkmale (functional features) definieren Teile des Gegenstandes des Schutzbegehrens nicht mit körperlichen Merkmalen, sondern durch die Angabe der Wirkung oder von Eigenschaften. Sie sind als technische Merkmale zu werten, die die Neuheit begründen können. Solche Wirkungsangaben oder Gestaltungsprinzipien sind eine Verallgemeinerung der Lehre, weil alle Mittel gleicher Wirkung unter Schutz gestellt werden, während die konkret offenbarten Mittel nur Beispiele sind. Sie sind zulässig, wenn durch die Angabe der Wirkung, der Eigenschaft oder des Prinzips: i) eine technische Lehre ausreichend offenbart ist; ii) der Stand der Technik die Verallgemeinerung der Lehre zuläßt; iii) Klarheit des Anspruchs darunter nicht leidet; iv) das Merkmal objektiv präziser nicht umschrieben werden kann.“

³²GRUR 1998 843 852

³³dieser Plan wird nach §1 (2) 3. PatG nicht als Erfindung angesehen

³⁴Schulte, 5. Auflage §35 Rdn 54f

Für ein Computerprogramm kann aber die von ihm hervorgerufene Wirkung normalerweise nicht als ausreichende Offenbarung angesehen werden, da die Lösung der Aufgabe „*Wie bringe ich das meinem Computer bei?*“ nicht offenbart wird. Objektiv präziser beschreibt auf jeden Fall der Programmcode die Lösung der Aufgabe.

4.1 Offenbarung des Programmcodes

Möglicherweise geht die Praxis der Anmelder, keinen Quellcode für programmbezogene Lehren zu offenbaren, selbst wieder auf den Ausschlußkatalog des §1 (2) PatG zurück, da ein Merkmal, das als nicht erfinderisch angesehen wird, auch nicht offenbart werden muss. Wenn aber die Programmierung der Lösung von programmbezogenen Lehren nur so weit offenbart werden braucht, dass sie „*jeden Fachmann in die Lage versetzen (muß), ohne eigene erfinderische Tätigkeit die gegebene Lehre zu vollziehen*“³⁵, so kommt man zu dem paradoxen Zustand, dass für programmbezogene Lehren so gut wie nichts mehr offenbart werden muss, da ja auch die Lösung beliebig schwieriger Programmieraufgaben keine erfinderische Tätigkeit begründen kann.

Wendet man folglich bei Computerprogrammen deren Ausschluss nur mehr auf die Offenbarungspflicht des Anmelders an, aber nicht mehr generell auf die Patentierbarkeit, so gelangt man zu einem Zustand, welcher der allgemeinen Anmeldung- und Patentierungspraxis nicht unähnlich erscheinen mag.

³⁵vgl. Schulte 5. Auflage §1 Rdn 39 mwN

4.2 Patentierung von Lösungen

Diese weltweit praktizierte Vorgehensweise, Wirkungen zu patentieren, hat jedoch verhängnisvolle Folgen. §1 Abs 1 PatG besagt, dass jeder Erfindung, so sie neu ist und auf einer erfinderischen Tätigkeit beruht, Patentschutz zusteht. Daraus folgt jedoch, dass jeder neuen, nicht naheliegenden Lösung derselben Aufgabe Patentschutz zusteht³⁶. Gewährt man jedoch Patentschutz auf die zu lösende Aufgabe, so erteilt man damit ein Monopol für alle möglichen Lösungen der zu lösenden Aufgabe. Man verletzt damit das Recht am geistigen Eigentum derer, die eine andere, neue, nicht naheliegende Lösung der selben Aufgabe entwickeln. Auch wenn es nicht explizit Sache des Prüfers ist, sich über mögliche zukünftige Verletzungen eines Patents Gedanken zu machen³⁷, so ist er doch schon aufgrund des §52 Abs 1 Bundesbeamtengesetz dazu verpflichtet, bei seiner Amtsführung auf das Wohl der Allgemeinheit Bedacht zu nehmen.

Sinnvollerweise kann über den Schutzbereich eines Patentbesitzes – d.h. der Bereich aller naheliegenden, nicht erfinderischen Gegenstände – nicht der Patentprüfer entscheiden, da er mögliche spätere Verletzungsfälle noch nicht kennen kann. Dies ist einzig die Aufgabe des Verletzungsrichters.

Melullis will anscheinend aus einem vergleichbaren Grunde lediglich „*die Umsetzung in die zur Ausführung erforderlichen*

³⁶mündlicher Vortrag LRD Flad, Patentrechtsskurs 1 für Patentprüfer im DPMA

³⁷vgl. „Lindenmaiersche Dreiteilungslehre“ in Kurt von Falck, GRUR 1984 392-397

Schritte³⁸ patentieren. Er begründet hierzu:

„Dieser gedankliche Ansatz vermeidet die Gefahr, daß die Patentierung von Software zu einer Monopolisierung des Denkens führen könnte, die alle anderen Menschen von bestimmten Formen der Benutzung ihres Verstandes ausschließen würde, und trägt damit einem Anliegen Rechnung, das hinter dem Verbot in Art. 52 Abs 2 steht.“

Dem kann der Autor nur zustimmen.

4.3 Programme als Lösungen

Dass eine Aufgabe keine Erfindung ist, sondern eine Erfindung nur in der Lösung der Aufgabe liegen kann, hat der BGH bereits festgestellt³⁹. Auch das Bundespatentgericht erkennt, dass die Tätigkeit des einschlägigen Fachmanns erst bei der Realisierung einsetzt⁴⁰. Um festzulegen, was bei programmbezogenen Erfindungen nun Aufgabe und Lösung darstellt, lohnt es sich einen Blick in ein Lexikon bzgl. Programme zu werfen. Der Autor zitiert hier nur aus den 4 gerade greifbaren Quellen. Die Auswahl ist dadurch rein willkürlich.

- Lexikon der Datenverarbeitung⁴¹: *„In der Datentechnik bezeichnet man eine Anweisung oder eine Folge von An-*

weisungen zur Lösung einer bestimmten Aufgabe als Programm. (Der Begriff „Anweisung“ ist hier im allgemeinen Sinn zu verstehen, nicht in dem besonderen, der ihm als Anweisung einer Programmiersprache zukommt.)“ und später: *„Das Ausarbeiten von Programmen wird als programmieren bezeichnet.“*

- Der Grosse Brockhaus⁴²: *„1) Datentechnik: vollständige Anweisung für automatisch gesteuerte Maschinen wie Werkzeug- oder Textilmaschinen, Rechenanlagen u. a., nach der die zur Bearbeitung einer Aufgabe erforderl. Arbeitsgänge nacheinander, z.T. auch gleichzeitig, ablaufen sollen.“* und im nächsten Absatz: *„Bei der Datenverarbeitung mit digitalen Rechenanlagen ist der Lösungsgang einer auszuführenden Aufgabe derart in allen Einzelheiten darzustellen (zu programmieren), daß die Rechenanlage eindeutige Anweisungen für den Arbeitsablauf erhält.“*
- Lexikon der Informatik und Datenverarbeitung⁴³: *„Eine zur Lösung einer Aufgabe vollständige Anweisung zusammen mit allen erforderlichen Vereinbarungen. Ergänzung: (1) Der Begriff „Aufgabe“ ist in der vorliegenden Definition nicht spezifiziert. In der Praxis wird hierunter meist eine Datenverarbeitungsaufgabe oder - von einem anderen Blickwinkel*

³⁸GRUR 1998 843 851f

³⁹„Kreiselegge“ GRUR 1984, 194-196

⁴⁰20 W (pat) 50/96 „Radio-Daten-System“ in BIPMZ 1998, 87-88

⁴¹2. Auflage, 1969, Verlag Moderne Industrie: unter „Programm“

⁴²Kompaktausgabe, aktualisierte 18. Aufl. 1983. unter „Programm“

⁴³2. Aufl. 1986, Oldenbourg Verlag. unter „Programm“

betrachtet - ein Teilstück aus einem menschlichen Informationsverarbeitungsprozeß (Informationsverarbeitung) verstanden, der vollständig formalisiert, in Algorithmen überführt schließlich von einem Rechner ausgeführt werden kann.“

- Encyclopædia Britannica⁴⁴ „*computer program, detailed plan or procedure for solving a problem with a computer; more specifically, an unambiguous, ordered sequence of computational instructions, necessary to achieve such a solution.*“

Konsens unter den Fachleuten scheint hier also zu sein, dass immer das Programm die Lösung der Aufgabe darstellt⁴⁵. Hier beißt sich die Katze wieder in den eigenen Schwanz: bei der Lösung einer Aufgabe durch einen Computer ist immer das ausführbare Programm selbst (als solches) mit seinen eindeutigen Anweisungen die Lösung. Da nur in der Lösung einer Aufgabe eine Erfindung gesehen werden kann, für Programme für Datenverarbeitungsanlagen als solche aber kein Schutz begehrt werden darf, dürften Programme für Datenverarbeitungsanlagen folglich nicht patentiert werden.

Ignoriert man jedoch, dass nur für die Lösungen von Problemen Patente erteilt werden, und beansprucht statt dessen das Arbeitsverfahren (die Idee, Aufgabe oder Wirkung) der Datenverarbeitungsanlagen, so

⁴⁴„<http://www.britannica.com/bcom/eb/-article/printable/9/0,5722,25459,00.html>“ [recherchiert am 11.09.2000]

⁴⁵wie dies auch der 17. Senat in 17 W (pat) 69/98 feststellt

steht vordergründig der Patentierung nichts mehr entgegen. Aber ein Patent auf das Arbeitsverfahren einer DVA beansprucht gerade wieder Schutz gegenüber allen anderen Programmen mit der selben Wirkung, unabhängig davon, ob diese neu und nicht naheliegend sind oder nicht. Das Recht am geistigen Eigentum anderer wird verletzt und der technische Fortschritt wird behindert.

4.3.1 Idee und Lösung

Zur Verdeutlichung ist hier vielleicht ein Beispiel angebracht. Der Autor dieses Artikels hatte vor seiner Zeit als Patentprüfer unter anderem die Instrumentensteuerungssoftware für die beiden FoRS-Fokalinstrumente des Very Large Telescope Projekts der Europäischen Südsternwarte (ESO) zu entwickeln. Jedes dieser Instrumente weist über 50 motorische Antriebe und eine Vielzahl weiterer sensorischer und aktuatorischer Elemente auf. Die Idee (das Lösungsprinzip) wie die Ansteuerung, Koordination und Überwachung dieser Vielzahl von Komponenten zu bewerkstelligen sein könnte, wurde während der Rückreise von einer Arbeitsbesprechung im Zug von Göttingen nach München geboren. Der zeitliche Aufwand hierfür betrug ca. eine Stunde. (Nach den Bestimmungen des Reisekostengesetzes wurde diese Zeit nicht einmal als Arbeitszeit anerkannt.) Die Suche nach geeigneten Bibliotheksfunktionen, Objektklassen und Datenstrukturen, mit denen sich diese Idee programmieren lassen könnte, dauerte ca. 2 Arbeitsmonate. Für die erste lauffähige Programmversion, welche die zugrunde liegende Idee konkret verwirklichte, mussten weitere 4 Monate Programmierarbeit ver-

gehen. Nach einem halben Mannjahr konnte die Lösung folglich erstmals präsentiert werden. Dies geschah durch einen Vortrag beim Auftraggeber ESO, dessen Vortragsfolien am gleichen Tage im Internet veröffentlicht wurden⁴⁶. Für die Fertigstellung des Programms incl. der üblichen Behebung von Fehlern müssen ca. 2 Mannjahre veranschlagt werden.

Das Arbeitsverfahren ließ sich nach ca. einer Stunde Nachdenkens so konkret formulieren wie dies bei den heutzutage üblicherweise beanspruchten Verfahren zum Betreiben eines Computers geschieht. Die Ausführung des Programms (der Lösung der Aufgabe „wie bringe ich es meinem Computer bei“) gelang jedoch erst mit der geschilderten Programmierarbeit. Wenn durch das Patentrecht nunmehr „*die Ideen und Grundsätze*“⁴⁷ geschützt werden sollen, ist dies blanker Hohn gegenüber der Leistung dessen, der diese Ideen und Grundsätze in die Tat umzusetzen vermag. Auf welcher gesetzlichen Grundlage dieser Schutz von Ideen und Grundsätzen basieren soll, ist nicht erkenntlich. Ideen und Grundsätze sind nichts anderes als Pläne für gedankliche Tätigkeiten und damit explizit durch §1 (2) 3. nicht erfinderisch. Ein Schutz dieser Ideen und Grundsätze als solche⁴⁸ würde anderen verbieten so zu denken.

⁴⁶<http://bigbang.usm.uni-muenchen.de:8002/-DOKU/icsTalk>

⁴⁷Statusbericht über die Rechtsprechung und Erteilungspraxis in Bezug auf softwarebezogene Erfindungen, Konferenz über wirtschaftspolitische Aspekte der Patentierung von Software, Bundesministerium für Wirtschaft und Technologie, 18. Mai 2000

⁴⁸§1 (3) in Verbindung mit §1 (2) 3.

5 Der Ausschlußkatalog als gesetzlicher Disclaimer

Es wird auch häufig versucht, den Begriff „Programm als solches“ unklar erscheinen zu lassen^{49,50} oder dieses neu zu definieren⁵¹, um damit die Zahl der vom Patentschutz ausgeschlossenen Gegenstände zu minimieren. Aber folgende Argumentation erscheint dem Autor überzeugender: Wenn für die genannten Gegenstände oder Tätigkeiten als solche kein Patentschutz begehrt werden kann, so entspricht dies einem gesetzlich aufgezwungenem Disclaimer. Bei programmbezogenen Erfindungen hätte dies zur Folge, dass ein Programm als solches niemals ein Patent verletzen könnte, da einem Programm als solchem kein Schutz gebührt. Auch ein aus dem Stand der Technik bekannter Gegenstand, der lediglich ein neues (nach §1 (2) 3. nicht erfinderisches) Programm ausführt, kann kein Patent verletzen, da „*die Verletzungsform gegenüber dem StdT keine patentfähige Erfindung*“ darstellt⁵². Eine Patentverletzung scheidet damit aus. Die Möglichkeit der freien Programmgestaltung der Programmierer wäre dadurch gewährleistet. Vieles spricht dafür, dass es dem Gesetzgeber ehemals darum ging, eben diese freie Programmgestaltungsmöglichkeit zu wahren; ebenso wie er

⁴⁹Schmidtchen aaO 1.1. a) - d)

⁵⁰Betten: „Clarification of the Exclusion of „Computer Programs“ in Art. 52(2) EPC“ In: UNION ROUND TABLE „Patenting of Computer Software“ 9./10. Dezember 1997 EPA München

⁵¹Melullis GRUR 1998 843 852

⁵²Schulte 5. Auflage §14 Rdn 30 7.22 unter Berufung auf BGH GRUR 86, 803 (6) Formstein mwN

dies für Pläne, Regeln und Verfahren für gedankliche Tätigkeiten vorgesehen hatte.

Eine vergleichbare Argumentation führt auch Melullis für das hinter den Computerprogrammen stehende gedanklich logische Konzept an und führt hierzu aus:

„Bei dieser Betrachtung stellen die Programme für Datenverarbeitungsanlagen nur eine weitere Erscheinungsform der Sachverhalte dar, in denen auch denkbare Regelungen und Vorschläge für das menschliche Verhalten erfaßt werden können, bei denen die Begründung eines Ausschließlichkeitsrechts eine Gefahr für die Freiheit des Denkens bedeuten könnte.“

Der Sicherung dieser Freiheit gewährt er jedoch nur eine untergeordnete Rolle, damit er den Patentschutz der ausführbaren Programme begründen kann.

Spricht man dem Schutz dieser Freiheit des Denkens wie des Programmierens jedoch eine zumindest gleichwertige Rolle zu – wie dies der Autor tut – so wäre die Ausschlussliste des §1 PatG dahingehend zu verstehen, dass genannte Gegenstände oder Tätigkeiten vom Patentrecht nicht gehindert werden dürfen; unabhängig davon, ob sie eine Lehre zum technischen Handeln aufweisen oder nicht. Diese Sichtweise scheinen auch die Mehrzahl der Gegner der Softwarepatentierung zu teilen, die durch die Patentierung von Software – egal ob in Form von Ideen, Konzepten, Algorithmen (mathematischen Methoden) oder aber in Form des konkreten Quellcodes – den boomenden Fortschritt ihres Wirtschaftssektors gefährdet sehen. Die hohe Zahl an Unterzeich-

nern der EuroLinux-Petition bestätigt dies. In den Computerwissenschaften wie auch in der Computerindustrie ist der treibende Faktor der Entwicklung die Offenlegung der eigenen Entwicklung und die Verbesserung bestehender Programme. Ohne die schnelle und ungehinderte Verbreitung und Diskussion der „Request for Comments“, welche die Basis des Internets bilden, ist der heutige Boom der Softwareindustrie undenkbar. Den wesentlichen Charakter der wissenschaftlichen Entwicklung und Meinungsbildung hat Wilhelm Busch einmal sehr anschaulich dargelegt:

*Die Wissenschaft sie ist und bleibt
was einer ab vom andern schreibt.*

Programme in Form des Quellcodes als Sprachwerke leben davon, daß sie den jeweiligen Erfordernissen schnell angepasst werden können. Nur mit dieser Freiheit kann es freischaffende Programmentwickler geben – so wie freischaffende Ärzte, Künstler oder Schriftsteller.

Aber zurück zur Betrachtung des Ausschlusskatalogs als gesetzlicher Disclaimer. Es wäre sicherlich interessant, wie ein Verletzungsrichter darauf reagieren würde, wenn der Beklagte zu seiner Verteidigung angäbe, er hätte nur ein „Programm als solches“ programmiert, aus der durch den gesetzlichen Wortlaut begründeten Überzeugung heraus, dass „Programme als solche“ keine Patente verletzen können, da auf „Programme als solche“ ja auch kein Patentschutz gewährt werden könne.

Dass bei jeder Programmierung das Programm gespeichert, getestet und damit ein Arbeitsverfahren ausgeführt werden muss, war und ist selbstverständlich. Ein auf diese

Teilaspekte der Programmentwicklung gerichteter Patentschutz kann folglich ebensowenig vom Gesetzgeber gewollt gewesen sein. Wenn die vorgenannte Argumentation vielleicht nicht jeden zu überzeugen vermag, so muss sie jedoch zumindest berechnete Zweifel aufwerfen. Wenn manchmal in dubio pro inventore gefordert wird⁵³, so kann man in diesem Falle mit zumindest gleicher Berechtigung ein in dubio pro reo fordern, auch wenn Zweifel im Patentrecht nicht angebracht sind. Ob die genannte Argumentation logisch einwandfrei ist, mag die öffentliche Diskussion oder aber ein konkreter Einzelfall ergeben.

6 Quellcode im Patentanspruch

Will man Programme für Datenverarbeitungsanlagen selbst als Erfindungen ansehen⁵⁴ und diese *als solche*⁵⁵ dem Patentschutz zugänglich machen – wie dies bei der geplanten Novelle des EPÜ vorgesehen ist – so sollte man sich genauere Gedanken darüber machen, was dies für Folgen haben könnte.

Betrachtet man nochmals die oben zitierten Definitionen eines Programms, so erkennt man, dass alle für einen Computer verständlichen Anweisungen, die eine Aufgabe lösen, ein Programm darstellen. Der Fachmann kennt nun Programme in binärer Form bzw. deren Darstellung in Hexcode, sowie Programme in einer der vielen Programmiersprachen (Ja-

va, C++, Tcl/Tk, Perl, Lisp, Python, Postscript, HTML, T_EX, Ada, Fortran, Pascal, Basic, Cobol, ...). Auch Mischformen graphischer Programmierung mit textuellen Ergänzungen sind bekannt (LabView und andere). Ist das Programm auf einem Rechner ausführbar und bewirkt es das zu implementierende Verfahren, so ist es als Lösung der Aufgabe zu betrachten, egal ob es direkt durch den Prozessor ausführbar ist – oder erst mit Hilfe weiterer bekannter Programme (Betriebssystem, Compiler, Interpreter, Emulator, Virtual Machine, ...). Zusammen mit der Angabe, unter welchen Bedingungen dieses Programm ausführbar ist (Prozessortyp, Betriebssystem, Programmbibliotheken oder was auch immer die Ausführbarkeit beeinflusst,) ist der Programmcode nach Ansicht des Autors sogar die einzig unzweideutige Offenbarungsform der Lösung der Aufgabe (bei Brockhaus heißt es „eindeutig“, bei britannica.com „unambiguous“).

6.1 Klarheit von Quellcode

Da jeder Programmcode einer mathematischen Formel, einem chemischen Strukturdiagramm oder einer mikrobiologischen DNA-Sequenz an Eindeutigkeit, Kürze und Prägnanz in der Beschreibung der Lösung in nichts nachsteht, sollte er als Merkmal eines Patentanspruchs nicht zu beanstanden sein, da alle Umschreibungen geeignet sind, „*die einen technischen Tatbestand wiedergeben, zB auch Formeln, die mathematische Beziehungen zwischen Physikalischen Größen enthalten. Nichttechnische Merkmale sind dagegen zu streichen, es sei denn, sie tragen zum besseren Verständnis der technischen Lehre bei, wie uU*

⁵³vgl. Beier GRUR Int. 1989 1 14

⁵⁴§1 (2) 3. PatG

⁵⁵§1 (3) PatG

ein Algorithmus“⁵⁶. Das Pochen auf die Erfordernis der deutschen Sprache macht hier keinen Sinn, da man zwar jedes Programm auch in deutscher Sprache abfassen kann, dies aber der Erfordernis an Klarheit und Knappheit, die sich nach dem Verständnis des Fachmanns richtet⁵⁷, zuwiderläuft. Schickedanz⁵⁸ listet hier zahlreiche deutsche, US-amerikanische und europäische Patentanmeldungen auf, bei denen „Formel-Ansprüche“ enthalten sind. Er bringt auch einen wunderschönen Vergleich von Quellcode in verschiedenen Programmiersprachen für den Euklidischen Algorithmus.

6.2 Die wesentlichen Merkmale

Streicht man Programme für Datenverarbeitungsanlagen aus dem Ausschlußkatalog des §1 PatG bzw. des Art. 52 EPÜ, so werden folglich auch Programme als Erfindungen angesehen. Für jedes Programm sind dessen einzelne Anweisungen wesentlich. Wird ein Programm selbst, oder aber ein Gegenstand der ein Programm umfasst beansprucht, so müssen folglich auch die wesentlichen Anweisungen des Programms als notwendige Merkmale zur Lösung der Aufgabe im Patentanspruch angegeben werden⁵⁹. Jede Anweisung eines Programms kann dann einen erfinderischen Schritt begründen. Gleiches müsste dann auch für neue und „erfinderische“ Datendeklarationen gelten. Diese Merkmale (Quelltext)

müßten mit anderen Quelltexten aus dem Stand der Technik auf Neuheit und erfinderische Tätigkeit verglichen werden. Der Autor kann sich dies zwar vorstellen; dass dies aber von den Befürwortern der Patentierbarkeit von Software so gewollt wird, kann er sich beim besten Willen nicht vorstellen.

6.3 Schutzbereich von Patentansprüchen mit Quellcode

Der Schutzbereich eines Patentanspruchs mit Quellcode sollte sich danach richten, in welcher Form der Anmelder seine Lösung offenbaren will.

Würde jemand beispielsweise ein Programm in binärer Form beanspruchen, so beschränkt sich sein Schutzbereich ausschließlich auf diese eine Ausführungsform auf den hierfür angegebenen Prozessorplattformen. Ist auch nur ein Bit anders, so kann man nicht einmal von einer naheliegenden Änderung sprechen, denn woher soll der Fachmann wissen, welches Bit relevant ist und welches nicht. Der Anmelder wäre in diesem Falle mit seinem Urheberrecht (Copyright) sicherlich besser bedient.

Würde er ein Programm z.B. in Java Sourcecode beanspruchen, so sind sicherlich einige Abwandlungen dieses Programms als naheliegend anzusehen. So darf die Namensgebung für Variablen, Klassen und Instanzen keine Rolle spielen. Auch so mancher Programmteil zur Fehlerbehandlung dürfte nur geringe Bedeutung haben, außer diese Fehlerbehandlung wäre für das Lösungsprinzip wichtig. Bei einer Umsetzung des Java-Codes in C++-code hätte der Autor jedoch bereits starke Bedenken, ob dies immer mit ausschließlich nahelie-

⁵⁶vgl. Schulte 5. Aufl. §35 Rnr 54a mwN

⁵⁷vgl. Schulte 5. Aufl. §35 Rdn 54

⁵⁸Mitt 2000 173 - 180

⁵⁹vgl. Schulte §35 Rdn 54b mwN

genden Schritten möglich ist (Trivialprogramme ausgenommen). Eine Übersetzung in Lisp würde sicherlich weit mehr als nur naheliegende Schritte benötigen. Selbst eine Portierung von einem Betriebssystem zu einem anderen oder der Wechsel von einer Programmbibliothek zu einer anderen kann haarsträubende Schwierigkeiten mit sich bringen. Welche Änderungen hier noch naheliegend sind und welche nicht, lässt sich von außen praktisch nicht mehr erkennen, außer man portiert den Code selbst. Der tatsächliche Schutzbereich eines Anspruchs mit Quellcode in einer hardwarefernen Programmiersprache wäre aber sicherlich auch sehr nahe am Programmtext. Es scheint daher auch in diesem Falle für den Anmelder besser zu sein, sich auf sein Urheberrecht zu berufen, anstatt kostspielige Patente zu beantragen.

Im praktischen Beispiel kann bereits die Festlegung der richtigen Endbedingung einer einzelnen Iterationsschleife Ideen benötigen, die nicht naheliegend sind, so dass durch die Festlegung der Endbedingung der Iterationsschleife bereits ein erfinderischer Schritt getan wurde. Der Leser mag daraus einen kleinen Eindruck gewinnen, welche unübersehbare Vielzahl an Lösungsmöglichkeiten bereits ein kleiner Ausschnitt eines Programmtextes liefern kann. Da aber dann praktisch jede mögliche Neuschöpfung eines Programms selbst patentierbar wäre und als nicht naheliegende Lösungen der selben Aufgabe auch die anderen Patente nicht verletzen könnte, wären diese Patente praktisch wertlos und würden lediglich vor dem Kopieren der eigenen Werke schützen. Man besäße das Urheberrecht nochmals in Patentform, garniert mit zusätzlichen Gebühreneinnahmen für die Ämter.

Es besteht jedoch dann auch die Gefahr, dass versucht wird, jeden dieser einzelnen Schritte separat zu beanspruchen. Dies hätte jedoch katastrophale Folgen für jeden Programmierer. Er müsste faktisch bei jeder Zeile Programmtext überprüfen, ob nicht darauf gerade ein Patent erteilt wurde. Freies Programmieren wäre praktisch nicht mehr möglich.

Beansprucht man jedoch nur seine prozessor-, sprach- und betriebssystem-unabhängige Programmidee als Wirkungsanspruch, und bekommt dafür auch noch ein Patent auf ein „*Verfahren zum ...*“, so ist man fein raus. Jede Lösung, die diese Programmidee – als Plan für eine gedankliche Tätigkeit – realisiert, verletzt das Patent. Hierzu muss man seine Programmidee noch nicht mal selbst realisiert, geschweige denn diese Realisierung offenbart haben. Nur leider wurde dann gar keine Lösung als patentfähige Erfindung beansprucht, sondern nur eine Plan für eine gedankliche Tätigkeit.

7 Schlussbemerkung

Wie man an dieser langen Aufzählung von Problemen bei der derzeitigen Patentierung von Programmen für Datenverarbeitungsanlagen sehen kann, gäbe es mehr als genügend Streitgründe für Einspruchs- und Nichtigkeitsverfahren. Die Tatsache, dass gegen die derzeitige Praxis der Patenterteilungen für Programmbezogene Lehren bei gleichzeitig erbittert ausgetragenen Diskussionen zwischen Befürwortern und Gegnern, keine wahre Einspruchs- und Prozessflut hereinbricht, kann einen nur verwundern. Möglich, dass die Schmerzschwelle der

Software-Entwickler noch nicht überschritten ist. Spätestens wenn die Patentierbarkeit von Software vom Gesetzgeber abgesegnet ist, wird aber diese Schmerzschwelle erreicht sein.

Gleichfalls möglich ist aber auch, dass keine der Seiten die andere mehr verstehen kann, da beide von unvereinbaren Grundsätzen ausgehen. Ein Indiz für letzteres mag folgende Episode sein: Bei dem hilflosen Versuch des Autors einem befreundeten Informatiker zu erklären, wieso Programme als solche nicht patentierbar wären, programmbezogene Lehren mit einem zusätzlichen technischen Effekt jedoch schon, erklärte dieser unverblümt: „Ihr spinnt’s ja komplett!“. Diese deutliche Aussprache hat dem Autor doch einiges zu Denken gegeben⁶⁰.

⁶⁰Der geneigte Leser mag den teilweise emotionalen Stil dieses Artikels entschuldigen, aber wer keine Emotionen äußert, ist nicht mit Herz und Seele bei der Sache.