Kommunikation & Recht

Betriebs-Berater für
● Medien ● Telekommunikation ● Multimedia

7/8
K&R

22. Jahrgang Juli/August 2019 Seiten 433 – 532

dfv Mediengruppe · Frankfurt am Main

RA Dr. Florian Deusch, Ravensburg und Prof. Dr. Tobias Eggendorfer, Weingarten*

Zur Kompatibilität beim Updating verbundener Systeme

Wenn ein Update den Arbeitsplatzrechner lahmlegt, ist das lästig – erwischt es ein System in der Produktionssteuerung, entsteht ein hoher Schaden. In modernen Unternehmen sind die verschiedenen Systeme vielfältig voneinander abhängig, wer haftet unter welchen Bedingungen für ihre Funktion bei Updates? Nach Auffassung der Autoren gibt es hier einen „spezifizierten Kompatibilitätsanspruch" der IT-Anwender, der diese Risiken angemessen zwischen Softwareprovider und Nutzer verteilt.

I. Problematiken beim Updating verbundener IT-Systeme

Alles ist mit allem verbunden. Industrie 4.0, Smart Home, kommunizierende Maschinen. Für alle Systeme gibt es in mehr oder weniger regelmäßigen Abständen Updates. Was gilt, wenn die Updates zu Kompatibilitätsproblemen zwischen den unterschiedlichen, aber miteinander verbundenen Systemen führen? Gibt es hier einen „Kompatibilitätsanspruch" des Nutzers?

Upgrades der Textverarbeitung machten früher Dateien älterer Versionen unlesbar;[1] was geschieht in Industrie 4.0, im Smart Home oder der vernetzten Klinik? Lässt sich die Notwendigkeit von Updates (Fehlerkorrekturen) möglicher Weise durch verbesserte Qualitätssicherheit reduzieren und so Kompatibilitätsrisiken minimieren?

Die vorliegenden Fragen sollen anhand einer konkreten Fallkonstellation (Abschnitt II.) aus technischer und rechtlicher Sicht beantwortet werden (Abschnitte III. und IV.).

II. Ausgangsfall

Update-Problematiken verdeutlichen sich z. B. an folgendem Fall:

Das Unternehmen U. hat seine SPS-gesteuerte Produktionsmaschine mit seinem ERP-System verbunden, das auf dem Cloud-Server mit dem Betriebssystem B. läuft. U. hat von B. deshalb eine Serverlizenz erworben und eine passende Abonnement-Lizenz über die Nutzung der Cloud- bzw. Online-Services des B.

Der Hersteller des Betriebssystems B. installiert ein Sicherheitsupdate auf dem Cloud-Server. Dies führt dazu, dass die Kommunikation zwischen der SPS-gesteuerten Produktionsmaschine mit dem ERP-System nicht mehr funktioniert. Wer haftet für den Ausfall? Wer veranlasst, verantwortet und zahlt die notwendigen Kompatibilitätstests?

III. Kompatibilitätsprobleme bei Updates in der vernetzten 4.0 Welt

Das folgende Kapitel definiert zunächst einige Begriffe aus Sicht der Informatik, um anschließend mögliche Ursachen von Softwarefehlern sowie Wege zu deren Behebung zu diskutieren.

1. Begriffe

Software kann aktualisiert werden durch Updates und Upgrades.

a) Updates

Ein Update korrigiert Nutzungsprobleme einer vorhandenen Software, ohne dabei planmäßig an der Benutzerschnittstelle oder an der Funktionalität Änderungen vorzusehen.[2]

Ursachen für solche Updates können Sicherheitslücken im System sein, die das Update behebt oder fehlerhafte Funktionen, wie z. B. eine nicht funktionierende Druckfunktion oder Schwierigkeiten beim Importieren von Daten. Ein Update sollte daher – qua definitionem – stets ohne Probleme installiert werden können.

In der Realität ist das leider nicht immer so, da nicht alle Hersteller von Software die Definition verinnerlicht haben.

Auch mangelhafte Qualitätssicherung in der Software-Entwicklung macht gelegentlich Updates erforderlich. Umso problematischer ist es, wenn derartige Updates entweder frühere Fehler wieder „aktivieren"[3] oder neue Probleme verursachen.

b) Upgrades

Die Informatik unterscheidet Upgrades von Updates durch einen Versionssprung in der Software mit neuen Funktionen, neuen Benutzerschnittstellen und sonstigen Anpassungen. Im Entwicklungsprozess wird die Software auf einer anderen Entwicklungsstufe weiterentwickelt, losgelöst von der vorangegangenen Programmversion.[4] Microsoft hat z. B. den Schritt von Windows 8 auf Windows 10 als Upgrade dargestellt.

Durch die neuen Funktionen eines Upgrades können Inkompatibilitäten mit alter Software auftreten. Software, die unter Mac OS X Sierra lief, ist unter der Nachfolgeversion Mojave nicht mehr zwingend lauffähig. Hersteller

* Der Autor *Deusch* ist als Rechtsanwalt, Fachanwalt für Informationstechnologierecht in der Anwaltskanzlei Dr. Gretter tätig und Lehrbeauftragter an der Hochschule Ravensburg-Weingarten; der Autor *Eggendorfer* ist Professor für IT-Sicherheit an der Hochschule Ravensburg-Weingarten und freiberuflicher IT-Berater. Beide Autoren sind zudem als Datenschutzbeauftragte tätig. Der Beitrag beruht auf einem Vortrag der Autoren beim 15. Forum für Verbraucherrechtswissenschaft an der Universität Bayreuth. Mehr über die Autoren erfahren Sie auf S. XII. Alle zitierten Internetquellen wurden zuletzt abgerufen am 26. 6. 2019.

1 Textdateien im Word-Format .doc (Word-Versionen vor 2007) sind nicht ohne weiteres im Format .docx lesbar, siehe https://support.office.com/de-de/article/ausw%C3%A4hlen-der-textcodierung-beim-%C3%96ffnen-und-speichern-von-dateien-60d59c21-88b5-4006-831c-d536d42fd861. Weitere Beispiele: https://www.welt.de/wirtschaft/webwelt/article147593917/Diese-Programme-laufen-nicht-mehr-unter-Windows-10.html; https://support.office.com/de-de/article/Bekannte-Probleme-mit-Office-und-Windows-10-28d56d84-c16c-4763-8eac-e51e05a37a16 (alte Office-Version läuft mit neuem Windows nicht richtig).

2 Brockhaus, Enzyklopädie, 21. Aufl. 2006, Band 28, Stichwort Update.

3 Beispielhaft: https://www.computerworld.com/article/3032206/office-2010-patch-kb-3114750-clobbers-outlook-calendar-again.html.

4 Siehe Fn. 2 sowie Shaw, Strategies for Managing Computer Software Upgrades, 2001, p. 4.

der Systeme informieren ihre Kunden in der Regel frühzeitig über Upgrades, so dass Bestandssoftware mit ausreichend zeitlichem Vorlauf umgestellt werden kann.

In vielen Systemen werden dazu bestimmte Funktionsaufrufe als „Deprecated" markiert.[5] Damit haben Entwickler über die Dauer von mehreren Upgrades Zeit, ihre Anwendungen anzupassen.[6]

c) Aktualisierungen als Rechtsbegriff

In rechtlicher Hinsicht hat sich bislang keine eindeutige Abgrenzung zwischen Upgrade und Update durchgesetzt. Teilweise wird das Upgrade als ein „nicht geschuldetes Addendum zur ursprünglichen Software" beschrieben.[7] Der Gesetzgeber hat jedenfalls keine Unterscheidung vorgenommen, sondern den Begriff „Aktualisierung" verwendet.[8]

d) API/Programmierschnittstelle

Soweit Programme mit anderen Programmen kommunizieren müssen, müssen die Entwickler zuvor den Kommunikationsweg definieren. Diese Funktion erfüllt die sogenannte API (Application Programming Interface = Programmierschnittstelle).

Wenn ein Entwickler eine größere Marktmacht als ein anderer Entwickler hat, definiert er diese Programmierschnittstelle einseitig, etwa wenn er eine Software herstellt, an die viele andere Programme andocken können sollen (z. B. PlugIns für Firefox oder Gimp). Diese Definition ist ein De-Facto-Standard, denn nur über diese API ist eine Interaktion mit der Software möglich.

Die API muss alle wesentlichen Funktionen vollständig und unmissverständlich dokumentieren, eine „heimliche" Änderung ohne Vorwarnung führt denklogisch zu Fehlfunktionen in Software, die auf die API der anderen Software aufsetzt.

Dass eine unzureichende Dokumentation oder zu häufige Änderung der API problematisch sind, hat die EU Kommission 2004 in einem kartellrechtlichen Verfahren der Free Software Foundation gegen Microsoft festgestellt.[9] Die API-Dokumentationen von Microsoft waren unvollständig und wurden zu oft geändert. Dadurch wurden Wettbewerber bewusst behindert.

e) Anwendungssoftware

Anwendungssoftware sind Programme, die Anwender nutzen, um konkrete Nutzungsfälle (Use-Cases) aus ihrem Tätigkeitsfeld zu erledigen. Beispiele dafür sind Buchungssoftware von Reisebüros, Textverarbeitungen, Tabellenkalkulationen, Grafikprogramme oder Kanzleisoftware. Im Ausgangsfall (oben Abschnitt II.) ist die ERP-Software Anwendungssoftware; auch die Steuerung der Produktionsmaschine sei für die vorliegende Diskussion als Anwendungssoftware betrachtet.

f) Betriebssysteme

Ein Betriebssystem stellt grundlegende Systemfunktionen zur Steuerung von Hardware zur Verfügung und verwaltet die Hardware-Ressourcen so, dass ein oder mehrere Anwendungsprogramme ungestört voneinander auf dem System lauffähig sind.[10] Damit Anwendungsprogramme die nötigen Systemfunktionen, wie z. B. das Schreiben von Dateien oder die Darstellung von Ausgaben in Fenstern, nutzen können, stellen Betriebssysteme eine API zur Verfügung.[11]

g) Kompatibilität

Unter Kompatibilität versteht die Informatik, dass zwei Systeme fehlerfrei zusammenarbeiten können. Dazu definieren die Systemanbieter jeweils gültige APIs und z. B. Netzwerkprotokolle.

Inkompatibilitäten können auftreten, wenn die Schnittstellendefinitionen nicht eingehalten oder geändert werden.

Die Richtlinie über digitale Inhalte[12] differenziert in Art. 2 Nr. 10 und 12 wie folgt:

- *Kompatibilität* als die Fähigkeit digitaler Inhalte oder digitaler Dienstleistungen, mit Hardware oder Software zu funktionieren, mit der digitale Inhalte oder digitale Dienstleistungen derselben Art in der Regel genutzt werden, ohne dass diese konvertiert werden müssen.
- *Interoperabilität* als die Fähigkeit digitaler Inhalte oder digitaler Dienstleistungen, mit anderer Hardware oder Software als derjenigen, mit der digitale Inhalte oder digitale Dienstleistungen derselben Art in der Regel genutzt werden, zu funktionieren.

h) Software-Fehler

Während sich der Irrglaube hält, dass Software aufgrund ihrer Komplexität nie fehlerfrei sein könne und Kunden trotz demonstrierter Mängel Softwareprodukte einkaufen[13], gibt es andere Industriezweige mit oft sogar höherer Produktkomplexität, in denen Mängel von Kunden nicht geduldet werden, wie zum Beispiel die legendären Mängel des „Elchtests" oder beim aktuellen Dieselskandal.[14]

Tatsächlich lässt sich Software mit geeigneten Qualitätssicherungsmaßnahmen weitestgehend fehlerfrei gestalten (siehe unten Ziffer 4). Die folgenden Abschnitte geben jedoch zunächst einen Überblick über Fehlerfolgen und deren Ursachen.

5 https://www.kernel.org/doc/html/v5.0/process/deprecated.html.

6 Die rechtliche Behandlung von Upgrades muss indes einer gesonderten Darstellung vorbehalten bleiben.

7 *Schmidt-Kessel* in seinem Vortrag „Funktionsverluste der Abnahme in Softwareverträgen" beim 15. Forum Verbraucherrechtswissenschaft an der Universität Bayreuth. Ein „nicht geschuldetes Addendum" dürfte am ehesten dem Upgrade entsprechen, wohingegen ein Update wohl eher im Rahmen der Mängelbeseitigung geschuldet sein dürfte. Dabei werden jedoch Upgrades ausgeblendet, die sich nicht als Addendum, sondern als Aliud im Vergleich zu bisherigen Versionen darstellen.

8 So jüngst Art. 8 Absatz (2) der Richtlinie des Europäischen Parlaments und des Rates über bestimmte vertragsrechtliche Aspekte der Bereitstellung digitaler Inhalte und digitaler Dienstleistungen, beschlossen durch den Rat der EU am 15. 4. 2019, siehe https://www.consilium.europa.eu/de/press/press-releases/2019/04/15/eu-adopts-new-rules-on-sales-contracts-for-goods-and-digital-content/(nachfolgend: Richtlinie über digitale Inhalte). Sie ist von den EU-Mitgliedstaaten binnen zwei Jahren umzusetzen.

9 https://fsfe.org/activities/ms-vs-eu/timeline.en.html; https://fsfe.org/activities/ms-vs-eu/CEC-C-2004-900-final.pdf.

10 *Tanenbaum/Herbert*, Modern Operating Systems, 4. Aufl. 2015, S. 3 - 5.

11 Z. B. die API-Informationen von Microsoft für das Betriebssystem Windows 10 unter https://docs.microsoft.com/de-de/uwp/; Updates dazu https://docs.microsoft.com/de-de/windows/uwp/whats-new/windows-10-build-18362-api-diff.

12 Siehe oben Fn. 8.

13 Bei einer Verkaufspräsentation von Windows 98 durch Bill Gates sind schwerwiegende Funktionsstörungen der Software aufgetreten, siehe https://www.youtube.com/watch?v=IW7Rqwwth84.

14 Zum Elchtest: https://www.spiegel.tv/videos/900790-die-a-klasse-und-der-elchtest; zum Dieselskandal beispielhaft https://www.ingenieur.de/tag/dieselskandal/. Zur Produktkomplexität: Software für den VW Golf mit ca. 12 000 000 Zeilen Programmcode (https://www.auto-motor-und-sport.de/neuheiten/vw-golf-8-problem-der-100-millionen-golf/); Windows 10 dagegen: 50 000 000 Zeilen Programmcode (https://code.org/loc), aber kein Fahrwerk, keine Bremsen, keine Aircondition, keine Sitzverstellung wie bei Autos.

2. Inkompatibilitäten

Funktioniert die Zusammenarbeit von zwei Systemen nicht wie vorgesehen, spricht man von einer Inkompatibilität. Die Ursachen dafür können vielfältig sein: Die API kann unvollständig, missverständlich oder fehlerhaft implementiert sein. Auch ein Programmierfehler beim Anbieter oder beim Nutzer der API kann zu unerwünschten Ergebnissen führen.

a) Fehlfunktion ohne Sicherheitsrisiko

Aus Sicherheitssicht als lästig, aus Sicht eines möglichen Sachmangels jedoch als durchaus relevant sind nichtsicherheitsrelevante Programmierfehler anzusehen. Dabei erfüllt die Software eine nach Spezifikation oder Nutzungszweck anzunehmende Funktion nicht korrekt.[15] Ein Beispiel dafür kann der (mittlerweile endlich) behobene Bug in MS Word dienen: Er führte dazu, dass Bildunterschriften von Bildern „versehentlich" falsch oder überhaupt nicht angezeigt wurden, oder die Bildunterschriften konnten nicht automatisch zu einem vollständigen Abbildungsverzeichnis zusammengeführt werden.

Die Hersteller reagieren auf solche Fälle sehr unterschiedlich: Einige erklären den Fehler zu einem erwünschten Verhalten („It's not a bug, it's a feature")[16] und kümmern sich frühestens bei einem kommenden Upgrade um Behebung. Andere Anbieter beheben derartige Fehler mit einem zeitnahen Update.

b) Fehlfunktion mit Sicherheitsrisiko

Kritischer sind Programmierfehler, über die Dritte ein System dazu veranlassen können, Funktionen auszuführen, die vom (rechtmäßigen) Nutzer nicht erwünscht sind, z. B. zur Ausführung von Programmcode eines Hackers wie beim Buffer Overflow oder zur Datenpreisgabe, wie bei der SQL-Injection. Solche Lücken bergen für den Betreiber der Software neben dem eingeschränkten Einsatz durch einen fehlerhaften Programmablauf erhebliche weitergehende Risiken, etwa Datenschutzverstöße oder Produkthaftungsrisiken. Problematisch ist z. B. fehlerhafte Software in Herzschrittmachern, über die unbefugte Dritte den Schrittmacher umkonfigurieren können.[17]

In der Praxis werden daher für solche Probleme zeitnah Updates bereitgestellt. Dabei ist das sogenannte „responsible disclosure" Verfahren gängig: Der Hersteller wird über das Problem vertraulich informiert, bereitet das Update vor (und sollte es auch testen), der Entdecker der Sicherheitslücke bekommt Gelegenheit, sich von ihrer Behebung zu überzeugen und veröffentlicht frühestens zeitgleich mit dem Update die Sicherheitslücke. Einige Hersteller sträuben sich zwar aus Marketinggründen, sicherheitsrelevante Programmierfehler einzuräumen. Dies kann dazu führen, dass die Sicherheitslücken vor ihrer Behebung publik werden, was oft „hektische" Zwischenupdates erfordert.[18] Dies ändert indes das grundlegende Vorgehen beim „responsible disclosure" nicht.

c) Logikfehler

Für Programmierfehler verantwortlich können Denk- oder Logikfehler des Entwicklers sein: Bestimmte Annahmen erweisen sich im Nachgang als falsch oder zu kurzsichtig, bestimmte Prozessschritte wurden bei der Programmierung übersehen oder falsch verstanden. Manchmal finden sich Implementierungen unnötiger Komplexität, weil zugrundeliegende IT-Prozesse oft nur unvollständig überblickt werden. Solche Fehler können für alle Arten von Fehlfunktionen verantwortlich sein. Ein Beispiel für solche Denkfehler ist die Heartbleed-Sicherheitslücke.[19]

d) Implementierungsfehler

Implementierungsfehler sind meist handwerkliche Versäumnisse. Programmierer verzählen sich (Off-by-One, Out-of-Bounds-Read) oder verwenden eine unsichere Funktion, für die es eine sichere Alternative gibt (strcpy statt strncopy).[20]

e) Compiler-Fehler

In seltenen Fällen ist die Software zwar logisch richtig und auch handwerklich ordentlich, aber Fehler entstehen durch die Übersetzung des Programm- bzw. Quellcodes in Maschinencode. Diese Übersetzung führt üblicherweise ein gesondertes Computerprogramm aus, der sogenannte „Compiler." Bei der Stagefright-Lücke in Android kam es zu einem Integer-Overflow, obwohl die Programmierer sorgfältig die Wertebereiche prüften. Wie sich später herausstellte, hatte der Compiler das Ergebnis einer Multiplikation zweier 32-Bit-Ganzzahlen, das in eine 64-Bit-Ganzzahl geschrieben werden sollte, als 32-Bit-Zahl belassen. Damit wurden die höchsten Stellen der Zahl abgeschnitten, was zu einem sicherheitsrelevanten Fehlverhalten führte.[21]

f) Aufwand bei Inkompatibilitäten

Problematisch ist es insbesondere, wenn wie im Ausgangsfall (Abschnitt II.) das Betriebssystem mit den Anwendungen nicht mehr kompatibel ist, weil ein Update die Betriebssystem-API geändert hat. Dies kann zu Funktionsausfällen bei der Anwendungssoftware führen und zu entsprechendem Aufwand bei der Fehlersuche. Da die Schnittstelle aufgrund der Marktmacht großer Anbieter in der Regel durch den Hersteller des Betriebssystems vorgegeben wird (siehe oben Ziffer 1 lit. d), sind die Hersteller der Anwendungssoftware gezwungen, ihre Programme an die geänderte Schnittstelle des Betriebssystems anzupassen. Die vorgenannten Aufwände zahlt schlussendlich der Nutzer, indem er teure Pflegeverträge mit den Herstellern der Anwendungssoftware abschließt oder sogar aktualisierte Anwendungssoftware erwerben muss.

Im Ausgangsfall (Abschnitt II.) trägt das Unternehmen U. folglich das Risiko des Systemausfalls und die Kosten der Anpassung der Anwendungen an das geänderte Betriebssystem.

---

15 Siehe zum Beispiel BGH, 5. 6. 2014 – VII ZR 276/13, K&R 2014, 601 – Mängel-Darlegungslast beim Softwarevertrag, BGH, 28. 5. 2014 – VIII ZR 94/13, BB 2014, 1999 – Akustische Einparkhilfe; OLG Hamburg, 16. 8. 2013 – 9 U 41/11, BeckRS 2013, 19747, OLG Düsseldorf, 14. 3. 2014 – I-22 U 134/13 und jüngst OLG Hamm, 22. 3. 2016 – 28 U 44/15, BeckRS 2016, 10945 (dort Rn. 59, 60).

16 https://www.wired.com/story/its-not-a-bug-its-a-feature/.

17 https://thehackernews.com/2017/08/pacemakers-hacking.html.

18 Beispielhaft: https://thehackernews.com/2019/03/microsoft-edge-ie-zero-days.html.

19 http://heartbleed.com/.

20 https://www.geeksforgeeks.org/why-strcpy-and-strncpy-are-not-safe-to-use/.

21 https://en.wikipedia.org/wiki/Stagefright_(bug).

Erhebliche Kosten können auch entstehen, wenn zwar die Schnittstellendefinition richtig ist, jedoch die API selbst eine Fehlfunktion aufweist.

3. Beheben von Softwarefehlern

Eine Behebung von Softwarefehlern erfordert stets, den fehlerhaften Programmcode zu korrigieren und den Anwendern diese korrigierte Software als Update oder Upgrade zur Verfügung zu stellen. In Einzelfällen existieren Work-Arounds, d. h. Nutzungswege, bei denen der Fehler nicht auftritt oder nicht stört. Ein Work-Around für die bei Audi-Dieseln verbaute Schummelsoftware wäre z. B. die Anforderung, nach Starten des Motors auf dem Prüfstand das Lenkrad einmal voll in beide Richtungen einzuschlagen, das würde den Schummelmodus deaktivieren und zu realen Testergebnissen führen.[22] Die fehlerhafte Software ist dann weiterhin im System enthalten, allerdings ohne Effekt.

4. Vermeiden von Softwarefehlern

Sofern Fehler nicht mutwillig, wie bei der Schummelsoftware,[23] eingebracht werden, verhindern mehrere einfache Qualitätssicherungsmaßnahmen Programmierfehler:

- Dokumentation des Codes, die auch gelesen wird, ist geeignet, insbesondere Logikfehler aufzudecken,
- Code-Reviews, bei der ein zweiter Entwickler den Programmcode analysiert und nachprüft,
- Coding-Standards, die z. B. die Nutzung unsicherer Funktionen verbieten,
- automatische Tests, die Extremwerte, Normalwerte und auch Werte früherer Fehler eingeben und die Ergebnisse prüfen,
- manuelle Tests auf die gewünschte Funktionalität sowie Penetrationstests,
- Dokumentation von Fehlern zur Vermeidung ihrer Wiederholung.

Nur wenige Softwareanbieter nutzen diese Möglichkeiten.[24] Dass sie erfolgversprechend sind, demonstriert OpenBSD seit 20 Jahren – mit nur zwei entfernt nutzbaren Sicherheitslücken.[25]

IV. Anspruch auf Kompatibilität?

Die Praxis ist hohen Aufwendungen ausgesetzt, um die Kompatibilität verschiedener Systeme nach einem Update herzustellen (siehe oben Ziffer III.). Nachfolgend wird geprüft, ob diese Aufwendungen aus rechtlicher Sicht letztlich durch den IT-Anwender zu tragen sind. Dazu sind zunächst die vertraglichen und gesetzlichen Pflichten im Ausgangsfall zu betrachten (Ziffer 1), bevor hieraus mögliche Ansprüche des U. abgeleitet werden können (Ziffer 2).

1. Einordnung der vertraglichen Beziehungen und Grundlage gesetzlicher Pflichten

Die vertraglichen Beziehungen werden getrennt für das Betriebssystem, das ERP-System und die Produktionsmaschine des U. betrachtet. Zudem können sich Ansprüche aus gesetzlichen Rechtsgrundlagen ergeben.

a) Betriebssystem

Im Ausgangsfall (oben Abschnitt II.) nutzt der IT-Anwender U. ein Betriebssystem, das er auf einem Server des Betriebssystemherstellers B. „in der Cloud" betreibt. B. überlässt dem U. dabei für die vereinbarte Dauer die Betriebssystem-Software nebst der Speicherkapazität auf einem Server, die zum Betrieb der Anwendungen nötig ist. Derartige Gebrauchsüberlassungen folgen dem Mietrecht.[26] Der Vermieter (Cloud-Provider) ist gemäß § 535 Abs. 1 S. 2 BGB verpflichtet, dem Mieter (Cloud-Nutzer)

- die Mietsache in einem zum vertragsgemäßen Gebrauch geeigneten Zustand zu überlassen und
- sie während der Mietzeit in diesem Zustand zu erhalten.
- Für Mängel, die der Vermieter zu vertreten hat, hat er Schadensersatz zu leisten (§ 536 a Abs. 1 Var. 2 BGB).[27]

Die Verträge der großen Hersteller von Betriebssystemen und Cloud Computing Anbieter lassen sich nur schwer in diese Vorgaben einordnen. Zum Beispiel lautet eine Regelung der Microsoft „Online Services Terms" (unter anderem maßgeblich für die Online-Nutzung von Produkten dieses Herstellers):[28]

„Microsoft ist berechtigt, Updates oder Ergänzungen zu dieser Software zu empfehlen oder mit oder ohne Ankündigung auf das Gerät des Kunden herunterzuladen."

Im Ausgangsfall ist somit zu erwarten, dass U. Updates für sein Betriebssystem erhält, ohne hiervon informiert zu werden.

b) ERP

Die möglichen Vertragskonstellationen für ERP-Systeme sind vielfältig. Oft gibt es einen Kauf- und Anpassungsvertrag über ERP-Standardsoftware. Updates werden aufgrund eines Pflegevertrags bereitgestellt. Dieser ist auch maßgeblich für die Frage, ob der ERP-Hersteller das ERP-System des U. im Ausgangsfall an das Update des Betriebssystems anzupassen hat. Der Pflegevertrag kann je nach Inhalt dem Dienst- oder Werkvertragsrecht unterliegen oder ein Sukzessivkaufvertrag sein.[29]

---

22 https://www.nzz.ch/wirtschaft/diesel-skandal-des-vw-konzerns-audi-betruegt-auch-in-deutschland-mit-illegaler-abschaltsoftware-ld.1299027.

23 Zum Dieselskandal siehe oben Fn. 14 und 22.

24 *Taeger* hat die Mängel in der Softwareerstellung bereits 1995 festgestellt (*Taeger*, Außervertragliche Haftung für fehlerhafte Computerprogramme, S. 61, 1995); ebenso mit dem Fokus auf Softwareanwendungen 4.0 *Deusch/Eggendorfer*, K&R 2016, 152, 157; dem folgend: *Schneider*, Handbuch EDV-Recht, 5. Aufl. 2017, Kap. R Rn. 342; zu Penetrationstests als Stand der Technik *Deusch/Eggendorfer*, K&R 2018, 223, 226.

25 https://www.openbsd.org/.

26 Zum ASP: BGH, 15. 11. 2006 – XII ZR 120/04, K&R 2007, 91 ff. = NJW 2007, 2394 Rn. 11 - 21 und OLG Hamburg, 15. 12. 2011 – 4 U 85/11, MMR 2012, 740, 741; *Schneider* (Fn. 24), Kap. M Rn. 472, Kap. R Rn. 499 und Kap. U Rn. 60 f.; 63 (die dort vorgenommene Differenzierung, wonach der Provider auch einen werkvertraglichen Leistungserfolg schulden kann, zum Beispiel eine Quartalsabrechnung, ist nach Auffassung der Autoren für den vorliegenden Fall der Betriebssystemnutzung nicht relevant); für die Anwendung von Mietrecht ebenso *Strittmatter*, in: Auer-Reinsdorff/Conrad, IT- und Datenschutzrecht, 2. Aufl. 2016, § 22 Rn. 29; *Ernst*, in: Ulmer/Brandner/Hensen, AGB-Recht, 12. Aufl. 2016, Teil 2, Softwareverträge Rn. 41; *Prinz zu Löwenstein*, in: Heussen/Hamm, Beck'sches Rechtsanwalts-Handbuch, 11. Aufl. 2016, Teil B § 40 Rn. 26; *Söbbing*, Cloud Computing und Virtualisierung in: Leible/Sosnitza, Onlinerecht 2.0: Alte Fragen – neue Antworten, 2010, S. 35, 47.

27 *Schneider* (Fn. 24), Kap. R Rn. 590.

28 https://www.microsoft.com/en-us/licensing/product-licensing/products#PT, dort unter „Online Services Terms" – German, Seite 7.

29 *Schneider* (Fn. 24), Kap. M Rn. 108 - 110 und Kap. S Rn. 10.12 zur rechtlichen Einordnung des Pflegevertrags; zur ERP-Pflege ebenso *Conrad/Schneider*, in: Auer-Reinsdorff/Conrad (Fn. 26), § 14 Rn. 9. ERP-Software wird aber auch als Cloud-Dienst angeboten.

c) Produktionsmaschine

Für die Steuerung der Produktionsmaschine des U. im Ausgangsfall gelten die Ausführungen zum ERP-System entsprechend. Meist gibt es einen Kaufvertrag nebst Wartung bzw. Pflege. Oftmals treffen die Hersteller derartiger Maschinen Aussagen zur Kompatibilität ihrer Produkte mit bestimmten Betriebssystemen. Beispielsweise hat der Hersteller Siemens für seine Software zur Programmierung von Steuerungen seiner Geräte Aussagen zur Kompatibilität mit dem Betriebssystem Windows 10 getroffen und gibt Hilfestellung zur Regulierung von Betriebssystemupdates. Teilweise werden Produkte „abgekündigt", die mit aktuellen Betriebssystemen nicht mehr kompatibel sind.[30]

d) Gesetzliche Rechtsgrundlagen für Updates und Kompatibilität

aa) Deliktsrecht

Software-Hersteller eröffnen durch die Verbreitung eines IT-Systems eine Gefahrenquelle, die dazu verpflichtet, dafür Sorge zu tragen, dass hieraus niemand zu Schaden kommt. Diese allgemeine Verkehrssicherungspflicht verlangt von Software-Herstellern zuvorderst eine sorgfältige Softwareerstellung (siehe dazu oben Abschnitt III. Ziffer 4 und unten Ziffer 2). Zudem müssen Hersteller von IT-Systemen Updates bereitstellen zur Schließung von Sicherheitslücken, die sie selbst festgestellt haben oder die anderweitig bekannt wurden. Weiter sind die betroffenen Nutzer zu informieren und zur Update-Installation zu instruieren. Dies gilt unabhängig davon, ob die Updates aufgrund eines Pflegevertrags vergütet werden oder ob das Update Sach- oder Rechtsmängel zur Erfüllung vertraglicher Gewährleistungspflichten beseitigt.[31]

Neben den Herstellern sind auch die IT-Nutzer zur Verkehrssicherung verpflichtet. Sie müssen die bereitgestellten Updates nach den Instruktionen des Herstellers installieren. Unternehmerischen Nutzern wird sogar zugemutet, ihre Systeme regelmäßig kostenpflichtig upzudaten oder ein neues Softwareprodukt zu erwerben.[32]

bb) Kompatibilität deliktsrechtlich geschuldeter Updates

Die Frage, ob derartige Updates auch mit anderen Systemen kompatibel sein müssen, ist soweit ersichtlich bislang nur vereinzelt behandelt worden. *Spindler* lehnt eine deliktische Rechtsgutverletzung am eingerichteten Gewerbebetrieb ab, wenn Updates zu Inkompatibilitäten mit anderen Computerprogrammen führen. Betroffen sei nicht das Integritätsinteresse, sondern lediglich das Äquivalenzinteresse, das nur durch vertragliche Rechtspositionen geschützt werden könne.[33] Demzufolge wäre die Erhaltung der Kompatibilität zweier Systeme nach einem Update nur geschuldet, wenn dies zuvor ausdrücklich vertraglich vereinbart worden ist. Ob diese Beurteilung (noch) zutrifft oder modifiziert zu betrachten ist, wird zu prüfen sein (unten Ziffer 2).

cc) Kompatibilität gemäß der Richtlinie über digitale Inhalte

Die Richtlinie über digitale Inhalte[34] schreibt in Art. 8 Abs. 1 vor, dass digitale Inhalte (wozu auch Software zählt) nur dann vertragsgerecht sind, wenn sie soweit mit Hard- oder Software kompatibel sind, wie dies bei digitalen Inhalten derselben Art üblich ist oder vernünftigerweise erwartet werden kann. Nach Art. 8 Abs. 2 sind die digitalen Inhalte zu aktualisieren („upzudaten"). Gemäß Art. 8 Abs. 4 sind müssen die Aktualisierungen ebenso vertragsgerecht sein wie die ursprünglichen Inhalte. Dies schließt die Kompatibilität ein.

Insoweit könnte sich aus dieser – noch umzusetzenden – Richtlinie ein gewisser Kompatibilitätsanspruch ergeben. Die Richtlinie gilt aber nur für Verbraucher und nicht für unternehmerische Softwarenutzer. Sie nimmt nicht den Softwarehersteller, sondern den Vertragspartner des Verbrauchers in die Pflicht (Art. 3 Abs. 1). Zahlreiche Einzelfragen sind noch ungeklärt, so dass zumindest derzeit noch keine hinreichend rechtssichere Aussage dazu getroffen werden kann, unter welchen näheren Voraussetzungen ein Anspruch auf Kompatibilität entstehen kann. Insbesondere ist offen, welche Auslegungen die unbestimmten Rechtsbegriffe in Rechtsprechung, Lehre und Vertragspraxis erfahren werden.

2. Ansprüche des Unternehmen

Die Konstellation im Ausgangsfall (Ziffer II., oben) ist typisch für die Vernetzung von Systemen in der „IoT"- oder „4.0"-Welt:[35] Beim Anwender werden verschiedene Systeme unterschiedlicher Hersteller miteinander verknüpft. Da sie regelmäßig aus gesonderten Bezugsquellen stammen, sieht sich der Anwender mit gesonderten Verträgen zum Betriebssystem und zu den weiteren Anwendungen konfrontiert.

a) Ausgangssituation für Unternehmen und Verwendungsrisiko des Anwenders

Im Ausgangsfall unterhält Unternehmen U. einen Cloud-Computing-Vertrag mit B. über die Bereitstellung und Nutzung des Betriebssystems, der dem Mietrecht unterliegt (oben Ziffer 1. lit. a).

Die Verträge über die Anschaffung des ERP-Systems und der Produktionsmaschine unterliegen dem Kaufrecht mit einem Pflegevertrag, der dem Dienst- oder Werkvertragsrecht folgt (oben Ziffer 1. lit. b und c).

Der Anwender U. ist somit für jedes der drei Systeme an unterschiedliche Verträge gebunden, mit gesonderten Vertragspartnern, die getrennt voneinander haften. Eine ver-

30 https://support.industry.siemens.com/cs/document/109749283/welche-inkompatibilit%C3%A4ten-bestehen-zwischen-windows-und-wincc-(tia-portal)-?dti=0&lc=de-WW; https://support.industry.siemens.com/cs/document/109754089/erh%C3%B6hung-der-verf%C3%BCgbarkeit-von-wincc-anlagen-durch-einsatz-eines-wsus-(microsoft-windows-server-update-service)-zur-geplanten-installation-ausgew%C3%A4hlter-windows-updates?dti=0&lc=de-WW; zur Abkündigung: https://de.wikipedia.org/wiki/STEP_7.

31 *Taeger* (Fn. 24), S. 225 ff., *Spindler*, Verantwortlichkeiten von IT-Herstellern, Nutzern und Intermediären, 2007, S. 60; *Spindler*, MMR 1998, 23, 24 f.; *Raue*, NJW 2017, 1841, 1844, *Wiebe*, NJW 2019, 625, 630; *Reusch*, BB 2019, 904, 905 f.

32 *Spindler* (Fn. 31), S. 164 - 166.

33 *Spindler* (Fn. 31), S. 56. *Taeger* (Fn. 24, dort S. 128) hatte in seiner Untersuchung 12 Jahre zuvor die Frage der Lauffähigkeit eines (nachgebesserten) Computerprogramms auf (IBM-) Hardware untersucht.

34 Siehe oben Fn. 8.

35 IoT = Internet of Things (https://de.wikipedia.org/wiki/Internet_der_Dinge); zur Vernetzung von Produktion und IT durch „Industrie 4.0" siehe https://www.bmwi.de/Redaktion/DE/Dossier/industrie-40.html.

tragliche Gesamtschuld besteht nicht.[36] Für den hier zu prüfenden „Anspruch auf Kompatibilität" des U. gibt es deshalb drei Beteiligte, deren Anspruchsverpflichtung getrennt zu beurteilen ist.

In der Fallkonstellation (Ziffer II. oben) besteht das Problem für U. darin, dass er die erworbenen (bzw. ihm überlassenen) Systeme aufgrund des Updates des B. nicht mehr verwenden kann, jedenfalls nicht ohne diese anderen Systeme anzupassen und den damit verbundenen Aufwand zu tragen. Im Grundsatz liegt dieses Risiko bei U. Der Mieter/Käufer einer Sache trägt stets das Risiko, den Vertragsgegenstand so einsetzen zu können, dass sich seine Erwartungen erfüllen, zum Beispiel an den geplanten Einsatzzweck oder den dabei geplanten Gewinn.[37]

Zu Lasten des IT-Anwenders wurde sogar folgendes entschieden: Der Auftragnehmer einer Softwareinstallation haftet – ohne ausdrückliche vertragliche Regelung – nicht für die Kompatibilität der zu installierenden Software mit dem Betriebssystem des Anwenders. Es sei die Angelegenheit des Anwenders, festzustellen und dem Auftragnehmer vorzugeben, auf welchem Betriebssystem mit welcher Version er die Software betreiben wolle.[38]

Zu prüfen ist daher, ob und falls ja welche Umstände es geben kann, von dem Grundsatz des Verwendungsrisikos abzuweichen.

b) Ansprüche gegen den Hersteller des Betriebssystems B.

Zur Abgrenzung, ob die Inkompatibilität zum Verwendungsrisiko des Anwenders gehört oder vom Softwarehersteller zu tragen ist, sind im Ausgangsfall die vertraglichen Ansprüche des U. zu untersuchen.

aa) Mangelfreie Software als vertragliche Hauptpflicht

Gemäß § 535 Abs. 1 S. 2 Alt. 1 BGB hat der Vermieter dem Mieter die Sache *in einem zum vertragsgemäßen Gebrauch geeigneten Zustand* zu überlassen.

Bisherige Beurteilungen zum vertragsgemäßen Zustand einer Software stellten darauf ab, ob sie alle zugesagten Funktionalitäten ausführt. Softwarefehler wie in Abschnitt III. dargestellt, werden bzw. wurden nur als Sachmangel gewertet, wenn sie sich auf die Softwarefunktionen auswirken.[39]

Diese Beurteilung kann nach Auffassung der Autoren nicht mehr aufrecht erhalten bleiben. Insbesondere aufgrund der Verknüpfung von Gegenständen und Softwareanwendungen verschiedenster Art im Zuge der Digitalisierung der Gesellschaft bedeutet jeder Softwarefehler ein Sicherheitsrisiko, aus dem gravierende Folgen erwachsen können. Dies gilt unabhängig davon, ob die Software die für den Anwender zugesagten Funktionen ausführt.[40]

Somit hat U. jedenfalls einen Anspruch auf eine fehlerfreie Betriebssystem-Software gegen den Hersteller B. Nachbesserungen durch Updates wären bei Erfüllung dieses Anspruchs überflüssig bzw. zumindest soweit möglich reduziert.

bb) Mangelfreie API mit Dokumentation als Hauptpflicht bei der Überlassung von Betriebssystem-Software

Mit Blick auf die Kompatibilität ist die Überlassung von Betriebssystem-Software im Ausgangsfall (Abschnitt II.) von einer Besonderheit geprägt: Der technische Zweck eines Betriebssystems ist es, mit Anwendungssoftware so zu kommunizieren, dass die Anwendung die Hard- und Softwareressourcen eines IT-Systems nutzen kann.[41]

Deshalb stellt sich die Frage, ob die API nebst Dokumentation für den vertragsgemäßen Gebrauch gemäß § 535 Abs. 1 S. 2 Alt. 1 BGB zwingend sind. Ohne anderweitige Vereinbarung schuldet der Vermieter einen Zustand, der den üblichen Gebrauch der Mietsache ermöglicht. Maßgeblich sind dabei Inhalt, Zweck und Umstände des Vertrags und die Standards, die sich nach der Verkehrsanschauung herausgebildet haben.[42]

Da ein Betriebssystem ohne API und zugehöriger Dokumentation nicht mit anderer Software kommunizieren kann, gehören diese Komponenten zu seinen Hauptbestandteilen. Dies gilt umso mehr, als sich in der Praxis die Veröffentlichung dieser Komponenten durch den Hersteller durchgesetzt hat. Die Hersteller der Betriebssysteme haben de facto selbst einen entsprechenden Standard geschaffen (siehe oben Abschnitt III. Ziffer 1. lit. d).

Der Anwender kann somit vom Hersteller eines Betriebssystems eine mangelfreie API nebst der Dokumentation verlangen, die nötig ist, um die Kompatibilität mit seinen Anwendungen herzustellen. Auf dieser Grundlage entwickeln die Hersteller der Anwendungssoftware ihre Computerprogramme, damit sie mit dem Betriebssystem betrieben werden können.

Diese Überlegungen gelten nicht nur für Betriebssysteme, sondern für alle Softwaresysteme mit einer veröffentlichten Programmierschnittstelle.

Damit hat der Anwender aber noch keinen Kompatibilitätsanspruch. Er ist selbst dafür verantwortlich, nach Erhalt der Betriebssystem-Software die Kompatibilität der hierauf beruhenden weiteren Systeme zu schaffen (im Ausgangsfall: ERP- und SPS-System). Grundlage hierfür ist jedoch die – mangelfreie – API des Betriebssystems. Zur Mangelfreiheit gehört die vollständige und ordnungsgemäße Dokumentation der API.

---

36 Obwohl jeder Softwarehersteller verkehrssicherungspflichtig ist (siehe oben Ziffer 1. lit. d), ist auch keine Gesamtschuld gemäß § 844 BGB gegeben, wenn die Programme verschiedener Hersteller verbunden werden. Denn im Ausgangsfall ist die Inkompatibilität durch das Update des Betriebssystems verursacht, für das nur B verantwortlich ist und nicht die anderen Softwarehersteller.

37 Für das Mietrecht z. B. BGH, 7. 10. 2015 – VIII ZR 247/14, NJW 2015, 3780, Rn. 26; BGH, 13.7. 2011 – XII ZR 189/09, NJW 2011, 3151, Rn. 9; BGH, 25. 11. 2015 – XII ZR 114/14, NZM 2016, 98, Rn. 33; für das Kaufrecht z. B. BGH, 16. 6. 2004 – VIII ZR 303/03, NJW 2004, 2301, 2302.

38 KG Berlin, 4. 11. 2010 – 2 U 116/05 – juris, Rn. 19, 20.

39 Nachweise siehe Fn. 15.

40 *Deusch/Eggendorfer*, K&R 2016, 152, 157; dem folgend: *Schneider* (Fn. 24), Kap. R Rn. 342; ähnlich *Raue*, NJW 2017, 1841, 1843, der jedoch eine Haftung für Sicherheitslücken ausschließen will, die (beim Kauf) nach Gefahrübergang entstanden sind, weil Angriffstechniken weiterentwickelt wurden. Da (auch neue) Angriffstechniken oftmals auf Softwarefehlern basieren, die bereits in dem angegriffenen Programm enthalten sind, bleiben für einen Haftungsausschluss des Softwareherstellers nur Fälle, in denen ein Angriff ohne Softwarefehler erfolgt oder wenn trotz Qualitätssicherung Softwarefehler aufgetreten sind. Viele Angriffe nutzen Standardverfahren aus, die wohlbekannt sind und leicht verhinderbar. Hat der Entwickler keine geeigneten Schutzmaßnahmen ergriffen, sind das Mängel.

41 Siehe oben Abschnitt III. Ziffer 1 lit. f und Fn. 10.

42 *Palandt*, Kommentar zum BGB, 78. Aufl. 2019, § 535 Rn. 17; BeckOK BGB, 49. Ed. 2019, § 535 Rn. 343.

cc) Erhaltung der API nebst Dokumentation als vertragliche Hauptpflicht: spezifizierter Kompatibilitätsanspruch des Anwenders

Gemäß § 535 Abs. 1 S. 2 BGB muss der Vermieter die Mietsache während der Mietzeit in vertragsgemäßem Zustand erhalten.

Wenn die Überlassung der API nebst Dokumentation bei Vertragsbeginn zur vertraglichen Hauptpflicht eines Cloud Computing Vertrags über ein Betriebssystem gehört, ist auch die Pflicht zur Erhaltung (mithin Aufrechterhaltung) der API eine vertragliche Hauptpflicht. Der Anwender hat das (miet-) vertragliche Recht, sich darauf zu verlassen, dass die API nicht geändert wird. Andernfalls ist die Kommunikationsfähigkeit des Betriebssystems mit den Anwendungen in Gefahr, die der Nutzer mit dem Betriebssystem betreibt.

Insofern kann von einem „spezifizierten Kompatibilitätsanspruch" des Anwenders einer Betriebssystem-Software gesprochen werden. Da der Hersteller eines Betriebssystems nicht absehen kann, welche Anwendungen ein Nutzer mit dem System kombinieren will, kann es ohne gesetzliche Regelung keinen pauschalen Anspruch auf Kompatibilität gegen Softwarehersteller geben. Der Hersteller eines Betriebssystems muss jedoch die Voraussetzungen dafür schaffen und – jedenfalls mietvertraglich – auch erhalten, dass Kompatibilitäten, die aufgrund einer von ihm veröffentlichten API hergestellt wurden, erhalten bleiben, und zwar durch Aufrechterhaltung der API.

Im Ausgangsfall (Abschnitt II.) bedeutet dies: Wenn der Betriebssystem-Hersteller B. bei seinem Update die API geändert hat, hat er seine vertragliche Hauptpflicht verletzt.[43]

Dieser „spezifizierte Kompatibilitätsanspruch" gilt nicht nur für Betriebssysteme, sondern für alle Softwaresysteme, die eine Programmierschnittstelle bereitstellen.

dd) Spezifizierte Kompatibilität und E-Privacy Verordnung

Außerhalb des Mietrechts kann der spezifizierte Kompatibilitätsanspruch im Rahmen der (noch immer nicht verabschiedeten) E-Privacy Verordnung relevant werden. Art. 8 Abs. 1 lit. e des Verordnungsentwurfs in der Fassung vom 4. 2. 2019 sieht vor, dass Verarbeitungs- und Speicherfunktionen von Endeinrichtungen auch ohne Einwilligung des betroffenen Nutzers für „Sicherheitsupdates" genutzt werden dürfen.[44] Derartige „Zwangsupdates" können Inkompatibilitäten beim Nutzer verursachen. Somit muss auch außerhalb des Mietrechts für Sicherheitsupdates der E-Privacy Verordnung gelten: Veröffentlichte Schnittstellen dürfen durch die Sicherheitsupdates grundsätzlich nicht geändert werden.

ee) Informationspflicht des Herstellers bei Änderung der API

Fraglich sind Fälle, in denen der Softwarehersteller aus technischen Gründen durch ein Update die API ändern muss, obwohl er vertraglich zu ihrer Aufrechterhaltung verpflichtet ist. Bei einem derart notwendigen Update dürfte es sich in der Regel um die Beseitigung eines Sachmangels handeln, für die der Vermieter unter den Voraussetzungen des § 536 a BGB Schadensersatz zu leisten hat (dazu siehe unten lit. ff).

Um bei einer Schnittstellenänderung im Rahmen der Mangelbeseitigung mögliche Inkompatibilitäten für den Anwender soweit wie möglich zu reduzieren, muss dieser wissen,

- wann das Update des Betriebssystems implementiert wird und
- welche Bereiche des Betriebssystems (bzw. der Schnittstelle) genau durch das Update in welcher Weise verändert werden.

Diese Informationen müssen dem U. so detailliert und so rechtzeitig vor der Update-Installation vorliegen, dass er (bzw. die von ihm beauftragten Dritten) in der Lage ist, die Kompatibilität des ERP-Programms und der Steuerung seiner Produktionsmaschine aufrechtzuerhalten.

Da die API nebst Dokumentation sowie deren Erhaltung zur vertraglichen Hauptpflicht des Cloud-Providers gehören (siehe oben lit. bb und cc), ist die Information über die API-Änderung ebenfalls Teil der Hauptpflicht. Die Information kann somit durch den Anwender als selbständiger Anspruch eingeklagt werden, im Gegensatz zu einer vertraglichen Nebenpflicht gemäß § 242 BGB.[45]

ff) Anpassungskosten als Schadensersatz?

Gemäß § 536 a Abs. 1 Var. 1 und 2 BGB muss der Vermieter Schadensersatz leisten für Mängel, die bei Beginn des Mietverhältnisses vorhanden waren und für solche, die später eintreten wegen eines Umstandes, den er zu vertreten hat. Für anfängliche (Software-) Mängel kann die mietvertragliche Garantiehaftung gemäß § 536 a Abs. 1 Var. 1 BGB in AGB ausgeschlossen werden. Für verschuldete Mängel (auch anfängliche) bleibt der Vermieter jedoch vollständig verantwortlich.[46] Hiernach ist der Hersteller des Betriebssystems B. als Vermieter jedenfalls gemäß § 536 a Abs. 1 BGB verpflichtet, die Schäden und Aufwendungen zu ersetzen, die bei U. infolge vermeidbarer Updates entstehen.

Versteht man die Bereitstellung und Aufrechterhaltung der API nebst Schnittstellendokumentation als vertragliche Hauptpflicht (siehe oben lit. bb und cc), so stellt sich die Änderung der Schnittstelle als Mangel dar. Für dessen Folgen kann der Mieter gemäß § 536 a BGB Schadensersatz verlangen. Der Cloud-Nutzer des Betriebssystems ist daher so zu stellen, wie er ohne die API-Änderung stünde.

Folglich gehört nicht die Herstellung jeglicher Kompatibilität zum Ersatzanspruch gemäß § 536 a BGB; zu erset-

43 Außerhalb des Mietrechts könnte sich der spezifizierte Kompatibilitätsanspruch aus der deliktischen Verkehrssicherungspflicht gemäß § 823 Absatz (1) BGB des Betriebssystemherstellers ergeben. Dabei wäre der Betriebssystemhersteller im Rahmen seiner deliktischen Updatepflicht gleichzeitig zur Aufrechterhaltung der API verpflichtet. Wie sich insofern die Unterschiede des Deliktsrechts zum Mietrecht auswirken, wäre indes gesondert zu prüfen.

44 Zum derzeitigen Stand des Gesetzgebungsverfahrens siehe http://www.cr-online.de/45742.htm; der aktuelle Kompromissvorschlag des Ratsvorsitzes vom 4. 2. 2019 ist abrufbar unter http://www.cr-online.de/ST_8537_2018_INIT_EN(1).pdf.

45 Zur Information als vertragliche Nebenpflicht: *Böttcher*, in: Erman, BGB, 15. Aufl. 2017, § 242, Rn. 93; *Pfeiffer*, in: Herberger/Martinek/Rüßmann/Weth/Würdinger, jurisPK-BGB, 8. Aufl. 2017, § 242 BGB Rn. 44.

46 Die Vertragspraxis setzt diese zulässige Option regelmäßig um, siehe dazu *Schneider* (Fn. 24), Kap. R Rn. 594; *Roth*, in: Auer-Reinsdorff/Conrad, IT- und Datenschutzrecht (Fn. 26), Teil C, § 13 Rn. 203 - 209. Zur Unwirksamkeit der Haftungsfreizeichnung des Vermieters für verschuldete, auch anfängliche Mängel siehe *Christensen*, in: Ulmer/Brandner/Hensen, AGB-Recht (Fn. 26), Teil 2, Mietverträge Rn. 40.

zen sind nur die Kosten des Nutzers, die bei der Anpassung seiner Anwendungen an die geänderte API entstehen. Insofern setzt sich der spezifizierte Kompatibilitätsanspruch beim Schadensersatz fort.

Im Ausgansfall muss der Cloud-Provider B. des Betriebssystems aber nur Schadensersatz leisten, sofern er die API-Änderung zu vertreten, das heißt vorsätzlich oder fahrlässig zu verschulden hat. Ein Verschulden ist anzunehmen, wenn der Softwarehersteller nicht alle gebotenen Qualitätsanforderungen bei der Softwareentwicklung erfüllt hat (siehe dazu oben Abschnitt III. Ziffer 4). Da diese Umstände im Herrschafts- und Einflussbereich des „Vermieters" B. liegen, muss er beweisen, wie er diese Anforderungen erfüllt hat.[47] Andernfalls kann der Softwarenutzer U. von B. die Kosten ersetzt verlangen, die zur Herstellung der Kompatibilität der Anwendungen mit der geänderten Betriebssystem-API nötig sind (siehe dazu oben Abschnitt III. Ziffer 3. lit. f).

Fraglich ist, ob sich dieser Befund durch § 439 Abs. 3 BGB ändert (in Kraft seit 1. 1. 2018). Hiernach hat der Verkäufer die Ein- und Ausbaukosten zu ersetzen, die dem Käufer dadurch entstehen, dass er die mangelhafte durch die nachgelieferte mangelfreie Sache austauschen muss, wenn die Kaufsache gemäß ihrer Art und ihrem Verwendungszweck in eine andere Sache eingebaut oder angebracht ist. Gemäß § 453 BGB gilt dies auch für Rechte und sonstige Gegenstände, mithin für Software. Ob die Begriffe „Ein- und Ausbau" auch die Aufwendungen für die Deinstallation der mangelhaften Software und die Installation der mangelfreien Software sowie der Neukonfiguration von Systemen erfasst, war im Gesetzgebungsverfahren streitig und ist bislang nicht geklärt.[48] Je nach Entscheidung dieser Frage wäre B. somit zum Ersatz der Anpassungskosten verpflichtet, wenn U. das Betriebssystem gekauft hätte.

Für das Mietrecht hat der Gesetzgeber jedoch keine vergleichbare Regelung beschlossen. Aus den Gesetzesmaterialien ist nicht ersichtlich, ob das im Kaufrecht erkannte Problem für das Mietrecht (relevant für Cloud Computing) übersehen oder bewusst nicht geregelt wurde. Die Regelung beruht auf einem EuGH-Urteil, in dem es um den Kauf und den Einbau von Fliesen und einer Spülmaschine ging.[49] Der EuGH hat dem Verbraucher die Erstattung dieser Kosten zugesprochen. Durch § 439 Abs. 3 BGB wollte der Gesetzgeber diese Rechtslage auch im unternehmerischen Rechtsverkehr herstellen. Diese Historie lässt keinen Raum für einen Umkehrschluss, dass der Ersatz der Ein- und Ausbaukosten auf das Kaufrecht begrenzt sei, bei einem Mietmangel aber nicht ersetzt werden soll.

c) Ansprüche des Unternehmens in Bezug auf das ERP-Programm und die Maschinensteuerung

Aus den Ausführungen gemäß lit. a und b folgt für die Vertragsbeziehungen des U. in Bezug auf das ERP-System und die Maschinensteuerung:

Auch hier gibt es ohne vertragliche Abrede keinen generellen Kompatibilitätsanspruch. Da die Hersteller dieser Systeme jedoch Aussagen zur Kompatibilität treffen, sollte U. im Ausgangsfall darauf achten, diese als vertragliche Zusagen zu dokumentieren.[50]

Im Ausgangsfall (Abschnitt II.) ist dem Unternehmen U. zu raten, die Vertragspartner des ERP- und des SPS-Systems um eine gesonderte Erfassung der Kompatibilitätskosten zu bitten, die aufgrund des Betriebssystem-Updates entstanden sind. Denn es ist zu unterstellen, dass die Hersteller der Anwendungssysteme für ihre Leistungen zur Schaffung und Aufrechterhaltung der Kompatibilität entweder durch einen erhöhten Kaufpreis oder einen kostenpflichtigen Pflegevertrags eine entsprechende Vergütung fordern.

V. Thesen und Fazit

- Das Updating einer Software, die mit anderen Systemen verbunden ist, kann zu Inkompatibilitäten führen, v. a. wenn Programmierschnittstellen geändert werden. Dies kann für den Anwender zu hohen Aufwendungen führen.
- Konsequente Qualitätssicherung in der Softwareentwicklung reduziert die Anlässe für Updates und damit die Risiken einer Inkompatibilität.
- Bei verbundenen Software-Systemen haftet jeder Softwarehersteller grundsätzlich nur für sein eigenes System.
- Der Käufer und Mieter einer Software trägt das Verwendungsrisiko hierfür. Es obliegt dem Anwender, die Kompatibilität zwischen verschiedenen Software-Systemen selbst auf eigene Kosten herzustellen, die er von verschiedenen Anbietern erworben hat. Deshalb gibt es keinen pauschalen Anspruch des IT-Anwenders auf Kompatibilität beliebiger von ihm erworbener Software.
- Die Überlassung fehlerfreier Software mit einem Qualitätsniveau, das das Bugfixing durch Updates möglichst vermeidet, ist eine vertragliche Hauptpflicht des Softwareanbieters.
- Hersteller von Softwaresystemen, die eine Programmierschnittstelle bereitstellen (zum Beispiel Betriebssysteme) schulden die Überlassung einer mangelfreien API nebst Dokumentation als vertragliche Hauptpflicht. Auf dieser Grundlage obliegt es dem Anwender, die Kompatibilität seiner Systeme herzustellen.
- Beim Updating von Softwaresystemen, die eine Programmierschnittstelle bereitstellen, gibt es jedenfalls im Rahmen des Mietrechts einen spezifizierten Kompatibilitätsanspruch des Anwenders: Die Programmierschnittstellen des Softwaresystems und die Dokumentation müssen unverändert aufrechterhalten bleiben.
- Falls dennoch Programmierschnittstellen eines Softwaresystems geändert werden müssen, trifft den Anbieter eine vertragliche Informationspflicht und eine Pflicht zum Ersatz der Kosten zur Beseitigung diesbezüglicher Inkompatibilitäten.

47 *Palandt* (Fn. 42), § 535 Rn. 11.

48 Zweifelnd die Stellungnahme des Bundesrats, BT-Drs. 18/8486, S. 83; dafür *Faust*, in: BeckOK BGB (Fn. 42), § 439 Rn. 91.

49 BT-Drs. 18/8486, S. 1, EuGH, 16. 6. 2011 – C 65/09 und C 87/09.

50 Bei Verträgen, die mit Verbrauchern außerhalb von Geschäftsräumen oder als Fernabsatzvertrag abgeschlossen werden, muss der Unternehmer gemäß § 312 d BGB i. V. m. Art. 246 a Nr. 15 EGBGB unaufgefordert auf Beschränkungen der Interoperabilität und Kompatibilität hinweisen.