17

SOFTWARE BLUES

Software bläht sich immer weiter auf, sodass man dauernd neue Hardware kaufen muss. Das nervt. Ein wichtiges Programm der ZEIT WISSEN-Redaktion bleibt allerdings seit Jahren äußerst bescheiden. Unser Autor besuchte die Programmierer, um sich zu bedanken – und zu fragen: Wie macht ihr das?

Text **Max Rauner** Fotos **Janus Engel Rasmussen**

Es gibt ein Wettrennen auf der Welt, das fast jeder Mensch mitfinanziert. Es wird nicht vom Fernsehen übertragen und auch nicht im Internet, aber wir alle sind Zeugen. Zwei Parteien beteiligen sich an diesem Rennen. Mal liegt die eine vorn, mal die andere, das Publikum merkt den Unterschied ziemlich schnell. Es ist das Wettrennen Hardware gegen Software.

Die Hardware: immer mehr Speicherplatz, immer schnellere Mikroprozessoren, immer dünnere Smartphones, immer kleinere SD-Karten. Die Hardware steigert sich exponentiell, Ingenieure nennen es das Mooresche Gesetz. Wenn die Hardware das Rennen anführt, werden unsere Geräte schneller, und ein Gefühl von Fortschritt stellt sich ein. Die Software: immer neue Updates, immer größere Betriebssysteme, immer mehr Apps, mehr Features, mehr Speicherplatzbedarf. Wenn Software die Hardware überholt, werden die Geräte langsamer, bis man irgendwann neue braucht. Das ist teuer. Aus Euphorie wird Frust.

Dieses Wettrennen ist durchaus im Sinne der Erfinder. Apple, eine Firma, die sowohl Software als auch Hardware herstellt, ist heute der wertvollste Aktienkonzern der Welt. Platz zwei: Saudi Arabian Oil, Platz drei bis sechs: Microsoft, Amazon, Alphabet (Google), Facebook. Jeder Kauf eines Computers oder Smartphones, jeder Einkauf im App- oder Play-Store finanziert das Rennen. Wir alle bezahlen dafür. Am frustrierendsten ist, dass die meiste Software sich unnötig breitmacht. Sie ist schlampig programmiert. Fachleute sprechen von »Bloatware«. *Bloat* ist englisch für aufblähen. Hätten Züge sich im selben Tempo weiterentwickelt wie Hardware, würden sie die Strecke Hamburg–Kopenhagen heute nicht in viereinhalb Stunden, sondern in weniger als 60 Sekunden zurücklegen. Diesen Fortschritt würde man spüren. Warum fühlt es sich bei Computern nicht so an? Weil jeder Fortschritt der Hardware durch Software-Aufblähung neutralisiert wird. Das Mooresche Gesetz macht Programmierer übermütig. Sie handeln nach der Maxime: Wir rennen schon mal vor, die Hardware wird uns schon einholen. »Software ist wie ein Gas«, sagte 1997 der Chief Technical Officer von Microsoft, Nathan Myhrvold, in einem Vortrag. »Sie dehnt sich immer aus, um den Behälter zu füllen, der ihr zur Verfügung steht.« Der Mann war ein Prophet.

Seit einigen Jahren beobachte ich auf meiner Festplatte eine Ausnahme, die mich

zunehmend fasziniert: Seit 2009 produzieren wir den ZEIT WISSEN-Podcast, und dafür nutzen wir fast von Beginn an ein Audio-Schnittprogramm mit dem merkwürdigen Namen Hindenburg. Es wird von einer dänischen Firma verkauft und hat unter Hörfunkjournalistinnen und Podcastern auf der ganzen Welt zahlreiche Fans. Hindenburg ist intuitiv bedienbar, nicht überladen mit Funktionen, schlicht im Design. Die Praktikantinnen in der Redaktion spielen zwei Stunden lang damit herum, dann schneiden sie halbstündige Podcast-Episoden. Nun gibt es einige gut designte Programme, jeder Mensch hat seine Lieblinge. Beeindruckender ist etwas anderes.

Hindenburg macht bei dem irren Wettrennen Software gegen Hardware einfach nicht mit. Diese Software will nicht immer mehr. Sie fordert nicht Jahr für Jahr Geld für neue Versionen. Sie bläht sich nicht unverschämt auf. Sie meldet sich mit Updates, das schon, aber während die meisten Programme auf meiner Festplatte sich breiter und breiter machen, ist Hindenburg einmal sogar geschrumpft. Die Version 1.85 gibt sich heute mit mageren zehn Megabyte zufrieden. Zehn Megabyte! So viel beansprucht ein einziges Foto meiner Digitalkamera. Es ist so, als wäre Hindenburg in den Hungerstreik getreten.

An einem dieser heißen Tage im August sitze ich im IC nach Kopenhagen, um die Leute kennenzulernen, die Hindenburg programmieren. Es ist immer interessant, wenn jemand etwas ganz anders macht als alle anderen. Außerdem reden gerade alle von Postwachstumsökonomie und Ressourcenschonung. Hindenburg ist kein Open-Source-Programm, aber es widersetzt sich der Wachstumslogik. Wie machen die das? Ist das die Zukunft? Und schließlich, nun ja, will ich endlich mal Danke sagen. Danke, liebe Programmierer, dass ihr meine Festplatte nicht zumüllt wie alle anderen!

Nick Dunkerley hat als Treffpunkt eine Adresse am Wasser genannt. Er steht als Creative Director und Gründer von Hindenburg Systems auf der Website, so heißt die Firma. Ich stelle mir vor, wie er von seinem Penthouse aus auf die Ostsee blickt, aber Google Maps führt mich zum Kopenhagener Jachthafen, wo weit und breit keine Häuser zu sehen sind. Dunkerley wohnt hier am Steg Nummer 1 auf einem 14 Meter langen Stahlschiff, das schon bessere Zeiten gesehen hat. Der Vorbesitzer umsegelte damit die Welt. Nun ragt anstelle des Masts ein Ofenrohr in den Himmel. Im Hafenbecken planscht ein Junge mit Taucherbrille, das ist Dunkerleys 15-jähriger Sohn, der den Autoschlüssel seines Vaters sucht. Gleich nebenan ragt ein Müllheizkraftwerk in den Himmel, dessen schräges Dach als Skipiste mit Kunstrasen dient, an der Seite gibt es eine Kletterwand, alles sehr Kopenhagen. Dunkerley kommt barfuß vom Duschen, ein braun gebrannter Typ mit T-Shirt, Tweedweste, Sonnenbrille. Er renoviert gerade das Nachbarschiff.

Im Silicon Valley wird Software in Garagen geboren, in Dänemark auf Booten. Ich hätte gleich darauf kommen können.

Im Silicon Valley wird Software in Garagen geboren, in Dänemark auf Booten. Ich hätte es mir denken können

Neben Dunkerley ist ein schlaksiger Mann an Bord, der eine Zigarette raucht und eher blass ist: Preben Friis, Mitgründer von Hindenburg Systems, Chief Technical Officer und das Gehirn hinter der Software. Der Coder. Er wohnt eine ÖPNV-Stunde entfernt, wo er sich ein Haus nach eigenen Angaben mit mehr als zehn Gitarren, fünf oder sechs Bässen und 13 Keyboards teilt. Die Lautsprecher und Verstärker wohnen in der Garage. Er beschreibt sich als »ewiger Junggeselle« und ist »nicht gerade ein Partytyp«, was ihm die Freiheit gibt, auch nachts und am Wochenende zu programmieren, ohne dass sich jemand beschwert.

Wir gehen Pasta essen im Hafenbistro, und es stellt sich heraus, dass das Geheimnis schlanker Software-Programmierung in diesem Fall mit drei Dingen zu tun hat: erstens mit der Freundschaft von zwei sehr unterschiedlichen Männern, Nick Dunkerley und Preben Friis, die zusammen eine Band hatten. Nick war der Leadsänger, Preben saß am Mischpult. Io hieß die Band, wie der Mond des Jupiter. Zweitens mit einer gewissen Sturheit der Deutschen Telekom. Und drittens mit einer Kleinstadt in Afrika.

Itezhi Tezhi liegt vier Autostunden westlich von Sambias Hauptstadt Lusaka. Hier sollte Nick Dunkerley 2007 mit Geldern des dänischen Außenministeriums ein Community-Radio mit aufbauen. Eigentlich wollte er vor allem seine Frau begleiten, die für eine dänische Hilfsorganisation nach Sambia ging, und auf den zweijährigen Sohn aufpassen, aber als gelernter Toningenieur und ehemaliger Hörfunkreporter kannte er sich mit Radiotechnik aus. »Innerhalb von 14 Tagen hatte ich einen Job«, sagt er. Überall im Land standen damals Radiosender, die von Hilfsorganisationen gesponsert worden waren. Ein Schiffscontainer mit Technik, eine Antenne auf einem Hügel, fertig. *»White elephants«,* sagt Dunkerley. Das ist der englische Begriff für Investitionsruinen. »Da war Technik drin, um Musik abzuspielen. Die Leute wollten aber im Radio keine Musik hören, sondern über ihre Angelegenheiten sprechen.«

Nick Dunkerley und die Menschen von Itezhi Tezhi machten es anders. Sie bauten ein Studio in der Nähe des Rathauses auf. Zentral gelegen und gut erreichbar für potenzielle Gesprächspartner. Als Hardware dienten einfache PCs, und als Aufnahmegeräte sollten die Lokalreporter ihre Nokia-Handys verwenden, die damals schon eine Aufnahmefunktion hatten. Das war zumindest der Plan. Das Problem war die Software. Die existierenden Programme waren zu teuer, zu komplex, zu aufgebläht. Dunkerley suchte nach einer Alternative – ohne Erfolg. Aber er hatte Glück: Sein Freund Preben besuchte ihn in Sambia.

Männerfreundschaften in Bands sind manchmal etwas seltsam. Man tourt jahrelang durch die Welt und weiß trotzdem nur ungefähr, was der andere beruflich so macht. Vielleicht ist es auch das tragische Schicksal von Informatikern, dass sich niemand wirklich im Detail für ihre Arbeit interessiert, bis man sie dringend braucht. Jedenfalls saßen Preben und Nick während des abendlichen Stromausfalls in Lusaka auf der Terrasse, und Nick erzählte von seiner verzweifelten Suche nach einem Audio-Schnittprogramm. Du kennst dich doch ein bisschen mit Software aus, habe er zu Preben gesagt. Sehr gut sogar, antwortete dieser. Nick: Ich brauche einen Audio-Editor für das Community-Radio, er muss simpel, zuverlässig und günstig sein. Kann man das auch selbst machen? Preben: »Wenn es nur Software ist, kann man alles machen.« Dieses Gespräch ist der Gründungsmythos von Hindenburg.

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Generator: Adobe Illustrator 15.0.2, SVG Export Plug-In . SVG Version: 6.00 Build 0)  -->
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd" [
	<!ENTITY ns_extend "http://ns.adobe.com/Extensibility/1.0/">
	<!ENTITY ns_ai "http://ns.adobe.com/AdobeIllustrator/10.0/">
	<!ENTITY ns_graphs "http://ns.adobe.com/Graphs/1.0/">
	<!ENTITY ns_vars "http://ns.adobe.com/Variables/1.0/">
	<!ENTITY ns_imrep "http://ns.adobe.com/ImageReplacement/1.0/">
	<!ENTITY ns_sfw "http://ns.adobe.com/SaveForWeb/1.0/">
	<!ENTITY ns_custom "http://ns.adobe.com/GenericCustomNamespace/1.0/">
	<!ENTITY ns_adobe_xpath "http://ns.adobe.com/XPath/1.0/">
]>
<svg version="1.1" id="Layer_1" xmlns:x="&ns_extend;" xmlns:i="&ns_ai;" xmlns:graph="&ns_graphs;"
	 xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" x="0px"
y="0px" width="16px" height="16px"
	 viewBox="0 0 16 16" enable-background="new 0 0 16 16" xml:space="preserve">
<switch>
	<foreignObject requiredExtensions="&ns_ai;" x="0" y="0" width="1" height="1">
		<i:pgfRef  xlink:href="#adobe_illustrator_pgf">
		</i:pgfRef>
	</foreignObject>
	<g i:extraneous="self">
		<path fill="#00A8FF" d="M14,1H2H1v13v1h1h13v-1V1H14z M6,14v-2h1v2H6z M11,14v-4H7H6H5v4H2
V2h2v5h8V2h2v12H11z"/>
	</g>
</switch>
<i:pgf  id="adobe_illustrator_pgf">
	<![CDATA[
	eJzsvXuPXLmRJ/oJzneo+8cA9mJVe/gmjcUClVlVs17IdsOvtTEYNDRqua0ZldRQq+3r++mXEfGL
IM/JzJJa0sx676poq7OiTpI8fATj8YvgP/w/X/3myc03b/7lxZNwvV4t//APx7cvnr178/ZnV0y9
+vmrVz98/+4tkX7y659euXS99odufl6/xoO/f/H2+5dvXv+M/3Tt+x/v6ds/+eXL5/92dfvD6397
8fbVi7/99OonP+1/+u3Ld69e9D/+5tlfXlx//5dvf6oN9hpun73rf3L5v6zlv7hI1Xlq6vjs9V+e
ff/9y/+P/xhq6LTDmx9ef/Py9beHN/9vJ149cak/3v/T//TfX/76xfeP/P32zfMfHl68fvfV2zfP
X3z//fHNqzdvv//Z1fFvz15f/eLZt/0vz5b+eunr+5evXvQ3eXj27so5eumbnzv/9eGHl6+++eUP
D//yor9jaI3I4Wuu5Xff96/3mugzkcvXP3/olN+8ePeud6W3QW/z6388zC13Ipef/NOvX3z7kse5
j8Q//xTVvn3z3cOzt/9G3+29z/39r1b5029fPHz3qg8Yv2G97i9H/+gHPNPfgP/+xBdHfwm1Xvla
+sfQ8MwYjhd/efnirz+7+uWb1y9kAG7evvuNjHqM6yr/yl9+/cOrF29/9/rlu96xTKQmI/CLN9+8
eNWft+/fv3rGL87FjX/lgd8+e/vti3d9ut68+uEdL6GqLfQRfvrsby9oZpw08KvvXrz+7Zvfcx+f
hHDVrnxfI9XVq5pC/8CVP3G8cKK15ca/UjFVQ5Vo7aXPyVd9ln719uW3L1//7Ilv9epJXKNM4D++
ffnNmD+3Tv+jBq5rim31Zz80X2Jypx/kbfrAvHv34jXerq+r4y+mdbJe/+I3vV93r785vnmg2fme
1nxfIK/72nn15lv5m33mv/Sv//CdvCP//nWfyK/evnxNdS6/5L/Ur7969UP/0z++fPDz9//ac3
y09kj//27bPn/bGrX/3Lv754/q5vVRDGp9/88PLdi+tnL7/76aMV9dd6++JK/ti/zL/qf9//7dsX
f+r7bXxdqHev//Li1ZvvpmqN8uz1N1f/89nb795f9Vevnr1+9vaK6Vbz05d/6X951sdo1D1oH1Bp
X0Hf9QHjr/AjuwYeeWD60wc09Ozdn68Or168/uZ7q1t+3XZcaO+v7zfPafW9vTq8/eH7P1/99s2b
V1bt9k9WO8hMpef/Ptr4ir/w+levZYBOW8ID+5Y6Z/q7a6U/fbmF/se/59qPz169evnt22fN1G
83MNnPm7tSR/+zEL628P//Lm1cvvH8Z6mihfPXv77uXzVy9+87fv3714eKw2Go0/vXz9TV+IzN1G
b988fEfyw9Vv/vzsuxdc57s/3/OTv7EK09edA8889smTR5hvPzAOr6e//+PbZ9+87Hy9S0L3z755
cfXuD/Zyl+Y236tPLbfaWju0Y7tr9zfrjbsJ
N/Em3eSbetNubm4ON8eb25u7m/vDenAH37sfD+mQD+VQD/1rB/o59jk63B3uj2sv7tiH4hiO8ZiO
vTPHcuxNHnv9x17TsT97lJ/bXu56uZdyu6I4KcutRwkocVcS//89ZflvfX573+4Ot71Q8/pzg1K5
lP5Kub9Y6g9Hc+T7y/Zyc8/lrg8Bd78PSLtpSx+b3Esf/z5a4cbfrO2+lzt6yz4oN31MSy+5j3Do
```

Ein Beispiel für aufgeblähte Software: Wer das kleine Symbol einer Diskette (rechts neben diesem Text) mit Adobe Illustrator zeichnet und abspeichert, erzeugt den oben gezeigten Code, der sich über weitere 50 Seiten erstreckt. Die Datei hat 3000 Zeilen und ist 225 Kilobyte groß. Für die Darstellung des Diskettensymbols reicht aber der rot markierte Code aus: sieben Zeilen, 237 Byte. Ein Tausendstel. Den großen Rest nutzt nur das Adobe-Programm

Nick Dunkerley (vorn) und **Preben Friis** programmierten ihre Software für alte Computer in Sambia. Dann stellte sich heraus, dass Anwender in reichen Ländern ganz ähnliche Bedürfnisse haben

Das Community-Radio von Itezhi Tezhi gibt es heute noch, aber Nick Dunkerley konnte bei der Einweihung nicht mit dabei sein. Seine Frau kam bei einem Autounfall in Afrika ums Leben. Er kehrte nach Dänemark zurück und zog mit seinem Sohn auf das Boot. Die Idee für die Software nahm er mit. Und dann gründeten Nick Dunkerley und Preben Friis ihre Firma, um ein Audio-Schnittprogramm für Entwicklungsländer zu erfinden.

Wie wurde Preben Friis zu einem Informatiker, der anders arbeitet als viele Kollegen? Man kann es nicht anders sagen: Er wurde dazu gezwungen, und zwar ausgerechnet von der Deutschen Telekom. Am Anfang war nämlich auch Friis ein Aufbläher. In seinem ersten Job nach dem Informatikstudium entwickelte er Software für einen dänischen Glasfaserhersteller. Das Mooresche Gesetz wird uns retten, hoffte man in der Firma. »Die Computer würden viermal so schnell und die Speicher viermal so groß sein, wenn wir fertig sind«, sagt Friis, »das war das Mindset meiner Kollegen und meines Chefs.«

Die Deutsche Telekom jedoch hatte bei der Firma eine Software bestellt, die auf den damals verbreiteten Laptops der Servicetechniker laufen sollte. Die Laptops waren eher langsam. »Es war unmöglich«, sagt Friis, »unser Programm war zu groß.« So kam er zu seiner ersten Verschlankungsmission. Er teilte die Software in ein kleines Hauptprogramm und mehrere Module auf, die nur bei Bedarf ausgeführt werden. Später hatte er noch einen ähnlichen Einsatz bei Nokia in Kopenhagen, wo das Betriebssystem für die billigsten und kleinsten Nokia-Handys geschrumpft werden musste. Preben Friis wurde ein Meister darin, aus Software die Luft rauszulassen.

Nick Dunkerley und Preben Friis veröffentlichten die erste Version ihrer Software im Oktober 2010. Hindenburg 1.0 brauchte 1,5 Megabyte Speicherplatz und passte damit in die Besenkammer jeder Festplatte. Große Marketingstrategen waren die beiden allerdings nicht. Zuerst tauften sie ihre Firma Nsaka, was niemand aussprechen kann. So heißen in Sambia die Hütten, in denen sich die Dorfgemeinschaft versammelt. Der Name Hindenburg wiederum ist eine Anspielung auf eine berühmte Live-Reportage vom Absturz des Zeppelins *Hindenburg* am 6. Mai 1937. Nette Idee, aber wenn man Hindenburg googelt, geht es seitenweise erst mal um Nazis, Hitler und Luftschiffe. Und dann wollten sie ihr Programm auch noch verschenken, finanziert

von einer Stiftung, das war der Plan. Es sollte ja Journalisten in Afrika dienen.

Verschenken? Das sei der schlechteste Businessplan, den er jemals gesehen habe, erklärte ihnen ein Bekannter, der sich mit Geld auskennt. Zumal auch Journalisten und Radiostationen in reichen Ländern an der Software Gefallen fanden. Und diese Leute sind nicht arm, oder? Der Mann, Chris Mottes, ist heute der dritte Eigentümer. Das neue Geschäftsmodell: an Reiche verkaufen, an Arme verschenken. Seit 2010 habe Hindenburg Systems 40.000 Lizenzen vergeben, schätzt Mottes, davon etwa ein Viertel kostenlos oder mit Rabatt an gemeinnützige Organisationen. Das Unternehmen hat siebeneinhalb Vollzeitstellen und macht 760.000 Euro Jahresumsatz. »Wir haben keine Investoren«, sagt Nick Dunkerley, »wir verschulden uns nicht, wir müssen keine Ziele erreichen. Wir machen einfach das, was wir wollen.« Reich werde man davon nicht. Er fährt einen Škoda.

Ist das jetzt subversiv? Nachhaltig? Idealistisch? »Unsere politische Absicht ist allenfalls, die technische Spaltung zu überwinden«, sagt Dunkerley. »Wir wollen die Welt ein kleines bisschen demokratischer machen.« Postwachstumsökonomie? Preben Friis sagt: *»It's just how I code.«* So programmiere er nun mal.

In der Hinterhofetage des Unternehmens in der Kopenhagener Altstadt zeigt mir Preben Friis ein Beispiel für Blähsoftware, wie sie sich auf jedem Computer versteckt. Es geht um die zahlreichen kleinen Bildchen und Symbole der Programmoberflächen. Das Briefumschlagsymbol im Mailprogramm. Den Radiergummi in der Bildverarbeitung. Die Ordnersymbole im Windows Explorer. Jede Anwendung ist voll davon, auch Hindenburg. Wenn die Entwickler nicht aufpassen, blähen diese Grafiken die Software auf.

Das Bild der Weltkugel, das Preben Friis verwenden wollte, war so hoch aufgelöst, dass man hineinzoomen und Florida erkennen konnte. Unnötig. Niemand muss Florida auf dem Icon für den Internet-Upload erkennen. Ein gedankenloser Programmierer würde das Bild genau so verwenden und dadurch Speicherplatz verschwenden. Nicht so der Kilobyte-Geizhals Preben Friis. Er reduzierte die Auflösung, sodass man allein Afrika in Umrissen erkennt. Außerdem schrieb er ein Hilfsprogramm, das alle in Hindenburg gezeigten Symbole nach überflüssigem Code untersucht und auf ihren Kern reduziert (siehe Seite 39).

Warum gibt es nicht mehr Software von dieser Sorte? Das fragen sich auch die Experten. Seit 15 Jahren arbeite er als Programmierer, schreibt der in München lebende Software-Ingenieur Nikita Prokopov in einem Essay auf *tonsky.me,* »und es macht mich fertig, dass unsere Branche keinen Wert auf Effizienz, Einfachheit und Qualität legt.« Viele Programme könnten mindestens hundertmal schneller laufen, wenn sie schlanker programmiert wären, schätzt er. Stattdessen verschwenden sie Speicherplatz und Rechenpower. »Würden Sie ein Auto kaufen, das auf 100 Kilometern 100 Liter schluckt? Oder 1000? Mit Computern passiert das jeden Tag.«

Manche Experten deuten die Mängel von Software als Vorzeichen einer Technologiekrise

Prokopov traf einen Nerv. Sein Essay wurde in sieben Sprachen übersetzt und tausendfach kommentiert. Eines der Probleme sei, sagt er, dass immer neue Programmierer an einer Software arbeiten und dabei Schicht für Schicht Code anhäufen, ohne dass mal jemand gründlich aufräumt. Niemand blickt mehr durch, aber es funktioniert ja irgendwie. Und die Kunden beschweren sich nicht. »Wir sagen ihnen: Wenn das Gerät sich aufhängt, mach einen Neustart. Und sie machen es. Sie haben keine Wahl, denn es gibt keine Konkurrenz. Jeder entwickelt dieselben langsamen, aufgeblähten und unzuverlässigen Produkte.«

Der Spieleentwickler Jonathan Blow vergleicht die Qualitätsmängel von Software sogar mit dem Niedergang von Hochkulturen in der Antike. Die alten Römer hatten Kristallgläser mit raffinierten Lichteffekten, die Byzantiner konstruierten Hightech-Flammenwerfer, die alten Griechen bauten analoge Rechenmaschinen. Das war Technologie auf höchstem Niveau, aber ihre Reiche sind trotzdem untergegangen. »Wir leben in einer sehr privilegierten Zeit, in der die Technologie schon seit Langem in einem guten Zustand ist«, sagte Blow im vergangenen Jahr auf der Entwicklerkonferenz Devgamm. »Wir halten es für selbstverständlich, dass Technologie immer besser wird und dass der Untergang dieser Zivilisationen kleine Ausrutscher der Vergangenheit waren. Aber das waren keine Ausrutscher, sondern es ist der normale Verlauf der Weltgeschichte, dass große technologische Errungenschaften völlig verloren gehen: weil es den Zivilisationen, die sie hervorgebracht haben, nicht gelungen ist, das Wissen mit in die Zukunft zu nehmen.«

Elon Musk hat es in einem Interview so formuliert: »Technologie wird nicht automatisch immer besser. Sie wird nur besser, wenn eine Menge Leute sehr hart daran arbeiten, sie zu verbessern.«

Das klingt alles ein bisschen deprimierend. Doch zum einen war Elon Musk vor Kurzem in Brandenburg und feierte das Richtfest für seine Batteriefabrik. Der Mann baut Raumschiffe und will zum Mars, aber er hat die Erde offensichtlich noch nicht aufgegeben. Und zum anderen gibt es ja Menschen wie Preben Friis, die hart daran arbeiten, Technologie besser zu machen.

Hindenburg könnte jetzt am Ziel sein, aber die Wahrheit ist natürlich: Keine Software ist jemals fertig. Es gibt kein Ziel. Das Rennen geht immer weiter. Für Ende dieses Jahres hat Apple eine kleine Revolution angekündigt: Statt mit Mikrochips von Intel werden Macbooks künftig mit Prozessoren der Firma ARM laufen. Die Hardware sprintet nach vorn.

Die Gitarren, Bässe und Synthesizer in Prebens Haus haben daher mal wieder einen neuen Mitbewohner: einen Mac-Prototyp mit ARM-Chip inside. Damit bereitet sich Preben Friis auf den Tag X vor. Denn natürlich kommt im Windschatten der neuen Hardware auch ein neues Betriebssystem. Es heißt »Big Sur«. Preben Friis muss Hindenburg dafür fit machen. Er hofft, dass er seine Software langfristig unter 50 Megabyte halten kann. Apple schreibt auf seiner Website: »Big Sur bringt das fortschrittlichste Computer-Betriebssystem der Welt auf ein neues Level von Leistung und Design.« Für mich klingt es wie eine Drohung. —

Max Rauner *hat einen 15-jährigen Sohn, der ihm regelmäßig von den neuen Features bestimmter Smartphones vorschwärmt. Die beiden werden sich nicht einig, ob das Fortschritt ist oder Abzocke*