TEXTE
105/2018

Entwicklung und Anwendung von Bewertungsgrundlagen für ressourceneffiziente Software unter Berücksichtigung bestehender Methodik

Abschlussbericht

Für Mensch & Umwelt

Umwelt Bundesamt

TEXTE 105/2018

Umweltforschungsplan des
Bundesministeriums für Umwelt,
Naturschutz und nukleare Sicherheit

Forschungskennzahl 3715 37 601 0
UBA-FB 002725

Entwicklung und Anwendung von Bewertungsgrundlagen für ressourceneffiziente Software unter Berücksichtigung bestehender Methodik

von

Dipl.-Ing. Jens Gröger, Dr. Andreas Köhler
Öko-Institut e.V., Freiburg

Prof. Dr. Stefan Naumann, Andreas Filler, M.Sc., Achim Guldner, M.Sc., Eva Kern, M.Sc.
Institut für Softwaresysteme, Hochschule Trier, Umwelt-Campus Birkenfeld

Prof. Dr. Lorenz M. Hilty, Yuliyan Maksimov, M.Sc.
Forschungsgruppe Informatik und Nachhaltigkeit, Universität Zürich

Im Auftrag des Umweltbundesamtes

Impressum

Herausgeber:
Umweltbundesamt
Wörlitzer Platz 1
06844 Dessau-Roßlau
Tel: +49 340-2103-0
Fax: +49 340-2103-2285
info@umweltbundesamt.de
Internet: www.umweltbundesamt.de

/umweltbundesamt.de
/umweltbundesamt

Durchführung der Studie:
Öko-Institut e.V.
Postfach 17 71
79017 Freiburg

Institut für Softwaresysteme in Wirtschaft, Umwelt und Verwaltung Hochschule Trier,
Umwelt-Campus Birkenfeld
Postfach 1380
55761 Birkenfeld

Institut für Informatik, Universität Zürich
Binzmühlestrasse 14
CH-8050 Zürich

Abschlussdatum:
Juli 2018

Redaktion:
Beratungsstelle nachhaltige Informations- und Kommunikationstechnik (Green-IT)
Marina Köhn

Publikationen als pdf:
http://www.umweltbundesamt.de/publikationen

ISSN 1862-4359

Dessau-Roßlau, Dezember 2018

Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autorinnen und Autoren.

Kurzbeschreibung

In dem Forschungsprojekt werden Bewertungsgrundlagen für ressourceneffiziente Software erforscht. Dazu wird ein Wirkungsmodell entwickelt, das den Zusammenhang zwischen der Nutzung von Software und dem Energieverbrauch von Hardware sowie Hardware-Inanspruchnahme aufzeigt. Durch eine Klassifikation von Softwareprodukten in lokale Anwendungen, Anwendungen mit entfernter Datenhaltung und Datenverarbeitung sowie in Serverdienste wird aufgezeigt, dass der Energie- und Ressourcenaufwand an unterschiedlichen Orten auftreten kann.

Aufbauend auf bestehenden Erkenntnissen zur ressourcenschonenden Softwareentwicklung und zum Softwaredesign wird eine Bewertungsmethodik entwickelt, die Software-Eigenschaften den Wirkungsbereichen *Ressourceneffizienz*, *potenzielle Hardware-Nutzungsdauer* und *Nutzungsautonomie* zuordnet. Es wird ein Kriterienkatalog mit insgesamt 25 Kriterien und 76 Indikatoren zur Überprüfung der Umweltauswirkung von Softwareprodukten aufgestellt.

Anhand von mehreren Fallbeispielen (2 Textverarbeitungsprogramme, 3 Internetbrowser, 3 Content-Management-Systeme, 3 Datenbanksysteme) wird aufgezeigt, dass die Bewertungsmethode und die entwickelten Kriterien in der Praxis anwendbar sind und relevante Unterschiede von Softwareprodukten gleicher Funktionalität offenlegen.

Im Ergebnis wird die Empfehlung ausgesprochen, ein Umweltzeichen „Blauer Engel“ für ressourceneffiziente Software zu entwickeln. Um die Anwendung der Bewertungsmethode zu erleichtern, werden außerdem drei ergänzende Bausteine entwickelt: ein Referenzsystem zur Durchführung von Messungen an Büro-Software, ein Softwaretool zur Auswertung von Hardwareauslastung und Energieverbrauch und ein Softwaretool zur Erhebung der Bewertungskriterien (Datenerfassung). In den Schlussfolgerungen werden Empfehlungen für die Anwendung der Bewertungsmethodik als produktpolitisches Steuerungsinstrument gegeben und es wird weitergehender Forschungsbedarf identifiziert.

Abstract

In this research project, the evaluation basis for resource-efficient software is being researched. To this end, an impact model is being developed that shows the link between the use of software and the energy consumption of hardware as well as the use of hardware. By classifying software products into local applications, remote data storage and processing applications as well as server services it is shown that energy and resources consumption may take place at different locations.

Based on existing findings on resource-saving software development and software design, an evaluation methodology assigning software properties to the impact areas of resource efficiency, potential useful life of hardware and autonomy of use is being developed. Furthermore, a catalogue of criteria with a total of 25 criteria and 76 indicators for assessing the environmental impact of software products is being drawn up.

Several case studies (two word processing programs, three internet browsers, three content management systems, three database systems) show that the evaluation method and the criteria developed are applicable in practice, and reveal relevant differences of software products with the same functionality.

As a result, the recommendation to develop a "Blue Angel" eco-label for resource-efficient software is being formulated. In order to facilitate the application of the evaluation method, three supplementary modules are also being developed, i.e. a reference system for carrying out measurements on office software, a software for evaluating hardware utilization and energy consumption, and a recording tool for obtaining the evaluation criteria. The conclusions will provide recommendations for the application of the evaluation methodology as controlling tool for product policy. Furthermore, they will identify further-going research needs.

Inhaltsverzeichnis

Abbildungsverzeichnis 10
Tabellenverzeichnis 12
Glossar 13
Zusammenfassung 15
Summary 22
1 Hintergrund und Zielsetzung des Vorhabens 28
2 Bearbeitungskonzept 28
3 Entwicklung von Bewertungsgrundlagen für Software 30
3.1 Präzisierung der Zielsetzung 30
3.2 Wirkungsmodell von Software 32
3.3 Klassifikation von Softwareprodukten 35
3.4 Bestandsaufnahme 37
3.5 Bewertung und Ergänzung der bestehenden Methoden und Modelle 37
3.6 Integration und Komplexitätsreduktion 38
3.7 Entwicklung eines Kriterienkataloges für nachhaltige Software 38
4 Anwendung der Bewertungsmethodik anhand von Fallbeispielen 39
4.1 Beschreibung des Vorgehens 39
4.1.1 Standardnutzungsszenarien 39
4.1.2 Vorgehensweise zur Messung von Hardware-Inanspruchnahme und Energieverbrauch 41
4.1.2.1 Messaufbau 43
4.1.2.2 Messablauf 45
4.1.3 Vorgehensweise zur Anwendung der weiteren Kriterien und Indikatoren 46
4.2 Auswahl der untersuchten Software 48
4.3 Exemplarische Anwendungsergebnisse der Kriterien und Indikatoren 49
4.3.1 Exemplarische Messergebnisse der Kriterien 1.1.3, 1.1.4 und 1.2 49
4.3.2 Exemplarische Mess- und Prüfergebnisse weiterer Kriterien 59
4.4 Weitere Hinweise zur Anwendbarkeit des Kriterienkatalogs 67
4.4.1 Hinweise zur vorgeschlagenen Mess- und Prüfmethodik 67
4.4.2 Festlegung von Referenzsystemen 67
4.4.3 Festlegung der Standardkonfiguration 68
4.5 Bewertung der Kriterien und Indikatoren hinsichtlich Aussagekraft und Umsetzbarkeit 68
5 Handlungsempfehlungen für die Entwicklung eines Umweltzeichens 70
6 Bausteine zur weiteren Verwendung der Bewertungsmethodik 72

6.1 Referenzsystem „Arbeitsplatzcomputer für Büro-Software“ ... 73
6.1.1 Anforderungen an das Referenzsystem ... 73
6.1.2 Auswahl eines Referenzsystems ... 74
6.1.3 Beschreibung des Referenzsystems ... 76
6.2 Auswertungssoftware zu Hardwareauslastung und zum Energieverbrauch ... 78
6.3 EXCEL-Tool zur Erfassung der Bewertungskriterien ... 82
7 Schlussfolgerungen und weiterer Forschungsbedarf ... 85
8 Quellenverzeichnis ... 87
Anhang 1 Kriterienkatalog für nachhaltige Software ... 92
1 Ressourceneffizienz ... 92
1.1 Hardwareeffizienz ... 92
1.1.1 Empfohlene Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte) ... 95
1.1.2 *Minimale* Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte) ... 96
1.1.3 Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration ... 96
1.1.4 Hardware-Inanspruchnahme bei normaler Nutzung unter der Annahme einer Standardkonfiguration und eines Standardnutzungsszenarios ... 97
1.1.5 Sparsame Hardwarenutzung durch Anpassbarkeit und Unterstützung der Nutzenden bei der Anpassung des Softwareprodukts ... 98
1.1.6 Online-Auslieferung ... 98
1.2 Energieeffizienz ... 98
1.3 Ressourcenmanagement ... 99
1.3.1 Anpassung der beanspruchten Kapazitäten an den Bedarf ... 99
1.3.2 Anpassung des Bedarfs an die verfügbaren Kapazitäten ... 100
1.3.3 Ressourcenschonende Standardeinstellungen ... 100
1.3.4 Feedback zur Beanspruchung von Hardwarekapazitäten und Energie ... 100
2 Potenzielle Hardware-Nutzungsdauer ... 101
2.1 Abwärtskompatibilität ... 101
2.2 Plattformunabhängigkeit und Portabilität ... 102
2.3 Hardwaresuffizienz ... 102
3 Nutzungsautonomie ... 103
3.1 Transparenz und Interoperabilität ... 103
3.1.1 Transparenz der Datenformate und Datenportabilität ... 103
3.1.2 Transparenz und Interoperabilität der Programme ... 104
3.1.3 Kontinuität des Softwareproduktes ... 104

3.1.4 Transparenz des Prozessmanagements 104

3.2 Deinstallierbarkeit 105

3.2.1 Deinstallierbarkeit der Programme 105

3.2.2 Löschbarkeit der Daten 105

3.3 Wartungsfunktionen 105

3.3.1 Datenwiederherstellbarkeit 106

3.3.2 Selbstreparaturfähigkeit 106

3.4 Unabhängigkeit von Fremdressourcen 106

3.4.1 Offlinefähigkeit 106

3.5 Qualität der Produktinformation 106

3.5.1 Verständlichkeit und Überschaubarkeit der Produktdokumentation, Lizenz- und Nutzungsbedingungen 106

3.5.2 Ressourcenrelevanz der Produktinformation 107

Anhang 2 Beispielhafter Ablauf eines Standardnutzungsszenarios für Textverarbeitungssoftware 108

Anhang 3 Operationalisierung für die Aufnahme der Indikatoren 110

Anhang 4 Ergebnisse der Kriterienerfassung im Detail 125

Anhang 5 XML-Schema „ssd-info“ 151

Abbildungsverzeichnis

Abbildung 1: Vergleich der Energieverbräuche des lokalen Geräts (SUT(Client)) während der Ausführung des Standardnutzungsszenarios 17

Abbildung 2: Hardware-Auslastung (CPU) im Leerlauf dreier Internetbrowser 18

Abbildung 3: Aufteilung der Arbeitspakete im Projektteam 29

Abbildung 4: Lebenszyklen von Hardware und Software (horizontale Dimension) und Beanspruchung von Ressourcen (vertikale Dimension) 32

Abbildung 5: Durchfluss von Energie und Hardware durch eine Organisation 33

Abbildung 6: Wirkungsmodell für Zusammenhänge zwischen Software-Eigenschaften, Nutzerverhalten/Organisation, Hardware-Inanspruchnahme und Ressourcenaufwand 35

Abbildung 7: Klassifikation von Anwendungssoftware bezüglich der Softwarearchitektur 36

Abbildung 8: Beispiel für die Struktur einzelner Kriterien im Kriterienkatalog 39

Abbildung 9: Exemplarischer Messaufbau Energie- und Ressourceneffizienzmessung von Software 43

Abbildung 10: Labor für Umwelt- und Nachhaltigkeitsinformatik mit SUT (Client), SUT (Server), Datenerfassungsstation und Janitza-Leistungsmessgerät 44

Abbildung 11: Hardware-Auslastung (CPU und RAM) im Leerlauf zweier Textverarbeitungsprogramme 50

Abbildung 12: Hardware-Auslastung (CPU) im Leerlauf dreier Internetbrowser 51

Abbildung 13: Hardware-Auslastung (CPU und RAM) im Leerlauf dreier Content Management Systeme (CMS) auf einem Server 52

Abbildung 14: Vergleich der Inanspruchnahme von Prozessor und Arbeitsspeicher 53

Abbildung 15: Vergleich der Inanspruchnahme von Permanentspeicher (HDD) und übertragener Datenmenge (Traffic) der Content Management Systeme auf SUT (Client) und SUT (Server) 54

Abbildung 16: Aus 30 Wiederholungen gemittelte Leistungsaufnahme der Messung des Standardnutzungsszenarios für eine Textverarbeitungssoftware 55

Abbildung 17: Vergleich der Textverarbeitungssoftware: Zeitlicher Verlauf 55

Abbildung 18: Vergleich der Energieverbräuche des lokalen Geräts (SUT(Client)) während der Ausführung des Standardnutzungsszenarios 56

Abbildung 19: Durch den Datenverkehr verursachter Energieverbrauch im Übertragungsnetz zur Ausführung des Standardnutzungsszenarios (Schätzung) 57

Abbildung 20: Energieverbrauch durch die entfernte Speicherung und Verarbeitung in Servern 58

Abbildung 21: Darstellung aller Indikatoren des Kriteriums 1.2 im Verhältnis zueinander, exemplarisch für die Browsermessungen bzw. -schätzungen 59

Abbildung 22: Vergleich minimaler Systemvoraussetzungen (RAM und HDD) 60

Abbildung 23: Wartezeit bis zur Aktivierung des Energiesparmodus der Software bei Standardeinstellungen 61

Abbildung 24: Abwärtskompatibilität der Content Management Systeme nach Betriebssystem 63

Abbildung 25: Typischer jährlicher Energieverbrauch (TEC) der Referenzsystem-Computer 77

Abbildung 26: Ausstattung der Referenzsystem-Computer mit Arbeitsspeicher (RAM) 78

Abbildung 27: Übersicht über die Zusammenarbeit und Funktionalitäten der einzelnen Bestandteile für eine Messung (System Under Test (SUT), Messgerät, OSCAR-Auswertungssoftware) 80

Abbildung 28: Beispiel-Screenshot der Auswertungssoftware OSCAR 81

Abbildung 29: Visualisierung der Struktur des XML-Schema „ssd-info“ 82

Abbildung 30: Screenshot des Erfassungstools: Hinweise zum zugrunde liegenden Projekt und zur Bearbeitung, zu Erfassungsmöglichkeit von Informationen zum Produkt sowie zur Messung. 83

Abbildung 31: Screenshot des Erfassungstools: Informationen zu den zu erfassenden Indikatoren werden über einen Kommentar angezeigt. 83

Abbildung 32: Screenshot des Erfassungstools: Auswahlmöglichkeiten bei vorgegebenen Ergebniswerten 84

Abbildung 33: Export der Excel-Datei in eine XML-Datei basierend auf dem XML-Schema "ssd-info" 84

Abbildung 34: Beispielhafter Messzyklus zur Bestimmung der Hardware-Auslastung 95

Tabellenverzeichnis

Tabelle 1: Systematik der Bewertungskriterien 16
Tabelle 2: Liste ausgewählter Architekturen und Softwareprodukte 16
Tabelle 3: Software-Kriterien zur potenziellen Anwendung in einem Umweltzeichen 19
Tabelle 4: Systematik der Bewertungskriterien 37
Tabelle 5: Überblick der Standardnutzungsszenarien für die ausgewählten Softwareprodukte 41
Tabelle 6: Ausstattung des Messlabors 42
Tabelle 7: Liste ausgewählter Architekturen und Softwareprodukte 49
Tabelle 8: Betriebssystemkompatibilität der Softwareprodukte 63
Tabelle 9: Offenlegung des Quellcodes und Lizenzmodell 64
Tabelle 10: Zeitraum, für den der Hersteller Sicherheitsupdates garantiert 65
Tabelle 11: Sicherheitsrisiken in den Softwareprodukten der Fallbeispiele 65
Tabelle 12: Bereitstellung differentieller Updates 66
Tabelle 13: Verbleibende Dateien und Registry-Einträge nach Deinstallation 66
Tabelle 14: Einschätzung der Anwendbarkeit und Aussagekraft der entwickelten Kriterien 68
Tabelle 15: Software-Kriterien zur potenziellen Anwendung in einem Umweltzeichen 71
Tabelle 16: Beschreibung des Referenzsystems als Zeitreihe 2010 – 2013 75
Tabelle 17: Beschreibung des Referenzsystems als Zeitreihe 2014 – 2017 75
Tabelle 18: Differenzierung der Hardwarekapazitäten in zwei Dimensionen 93
Tabelle 19: Grundlegende Definitionen für die Messung der Kriterien 1.1.3 und 1.1.4 94
Tabelle 20: Ergebnisse der Kriterienerfassung für die Fallbeispiele „Textverarbeitung (TVP)“ und „Browser (B)“ für die Fallbeispiele „Textverarbeitung“, „Browser“, „Content Management Systeme“ und „Datenbanken“ 125
Tabelle 21: Ergebnisse der Kriterienerfassung für die Fallbeispiele „Content Management Systeme (CMS)“ und „Datenbanken (DB)“ 136

Glossar

Content Management System	Ein Content Management System (CMS) ist ein Softwareprodukt zur Erstellung von Inhalten (Content) durch eine Gruppe von Autoren. Dies kann beispielsweise der Inhalt einer Internetseite sein oder ein unternehmensinternes Informationssystem.
Datenträger-Image	Ein Abbild eines Datenträgers oder einer Partition eines Datenträgers, das in einer Datei gespeichert wird.
Energieeffizienz	Menge an „nützlicher Arbeit“ dividiert durch den dabei anfallenden Energieaufwand. Die Steigerung der Energieeffizienz bedeutet damit die rationellere Verwendung von Energie. Im Kontext dieses Dokuments wird „nützliche Arbeit“ als erfolgreiche Ausführung von Standardnutzungsszenarien operationalisiert.
Hardware	Gesamtheit der für die Ausführung von Programmen, die Speicherung oder den Transport von Daten benötigten Sachgüter.
Hardwarekapazität	Quantifizierbare Eigenschaft eines Hardwaresystems, die die Grenze seiner Leistungsfähigkeit auf einer gegebenen Leistungsdimension darstellt (z. B. Arbeitsspeicherkapazität, Rechenkapazität, Bandbreite).
Hardwaresystem	Abgrenzbare Einheit von Hardware, die definierte Funktionen erbringt.
Indikator	Eine empirisch bestimmbare Größe, die Aufschluss über einen nicht direkt messbaren Sachverhalt gibt. Das Skalenniveau der in diesem Dokument vorgeschlagenen Indikatoren ist unterschiedlich. In einigen Fällen wird eine qualitative Ordinalskala angenommen (z. B. „ungenügend”, „genügend”, „gut”, „sehr gut” oder auch nur „erfüllt“, „nicht erfüllt“). Damit soll vermieden werden, dass durch die Verwendung einer Kardinalskala eine nicht vorhandene Präzision suggeriert wird.
Lasttreiber	Ein Lasttreiber simuliert eine Arbeitsleistung auf einem Computer, um Software zu testen. Ein Lasttreiber wird i.d.R. durch Automatisierungssoftware oder ein Benchmark-Programm realisiert.
Log-Datei	Von einem Programm automatisch erstellte Datei, die während des Ablaufs einer Messung (z. B. eines Standardnutzungsszenarios) die durchgeführten Aktionen inklusive Zeitstempel protokolliert.
Nutzungsmuster	Abstrahierte Form einer Sequenz von Interaktionen mit einem gegebenen Softwareprodukt.
Nutzungsszenario	Beschreibung eines Nutzungsmusters, in der Regel maschinell ausführbar.
Plugin	Ein Softwaremodul, das in ein Softwareprodukt eingebunden werden kann, um dessen Funktionalität zu erweitern.
Rechenleistung	Unter (maximaler) Rechenleistung einer Central Processing Unit (CPU) wird im Rahmen dieses Vorhabens vereinfacht das Produkt aus Taktfrequenz, Anzahl der CPU-Kerne und Datenbusbreite verstanden. Ein anteiliger Wert der Rechenleistung (in Prozent) wird erreicht, indem die Abarbeitung von Befehlen zeitlich gestaffelt, mit reduzierter Taktfrequenz, unter Nutzung von weniger als der verfügbaren Prozessorkerne und/oder geringerer Datenbusbreite erfolgt.

Referenzsystem	Ein Hardwaresystem, das hinsichtlich seiner wichtigsten Kapazitäten (z. B. Arbeitsspeicher, Rechenleistung) während einer festgelegten Zeitperiode (z. B. ein Jahr) als allgemein üblich definiert wird. Das Referenzsystem dient dazu, Indikatoren wie z. B. „minimaler lokaler Arbeitsspeicher" relativ zu einer Referenzgröße (der „aktuell üblichen" Arbeitsspeicherkapazität) ausdrücken zu können.
Ressource	Im Kontext dieses Dokuments eine natürliche Ressource, insbesondere ein Rohstoff, eine Energieform oder auch die Absorptionsfähigkeit eines Umweltmediums für Emissionen. Zur Abgrenzung gegen technische Ressourcen, insbesondere Hardwareressourcen, werden letztere hier präziser als „Hardwarekapazitäten" bezeichnet. Da die Beanspruchung von Hardwarekapazitäten stets zur Beanspruchung natürlicher Ressourcen führt, ist diese Abgrenzung (die in letzter Konsequenz auf eine definitorisch schwierige Grenzziehung zwischen Ökosphäre und Technosphäre hinausläuft) hier nicht von entscheidender Bedeutung.
Ressourceneffizienz	Allgemein die Menge an „nützlicher Arbeit" dividiert durch die dabei beanspruchten Ressourcen. Im Kontext dieses Dokuments wird „nützliche Arbeit" als erfolgreiche Ausführung von Standardnutzungsszenarien operationalisiert.
Software	Ausführbare Programme und die dazugehörigen Daten in digitaler Form.
Softwareprodukt	Eine abgrenzbare Einheit von Programmen und Daten, die zur Ausführung und Verarbeitung definierter Aufgaben auf einem Hardwaresystem bestimmt sind.
Standardkonfiguration	Eine als Referenz definierte Menge von Bedingungen, unter denen ein gegebenes Softwareprodukt betrieben wird. Sie umfasst die am Softwareprodukt während der Installation oder des Betriebs vorgenommenen Parametereinstellungen, die bereitgestellte Systemsoftware, ggf. weitere zum Betrieb benötigte Softwareprodukte sowie auf Hardwareebene das Referenzsystem.
Standardnutzungsszenario	Ein Nutzungsszenario, das zum Testen eines Softwareprodukts verwendet wird und möglichst repräsentativ für den üblichen Anwendungsfall sein soll.
System Under Test (SUT)	Ein Hardwaresystem, dessen verbrauchte Energie und verwendete Hardwarekapazitäten gemessen werden (z. B. während des Ablaufs eines Standardnutzungsszenarios). Beinhaltet neben der Hardware auch sämtliche zum Betrieb des Softwareprodukts notwendige Software (z. B. Betriebssystem, Laufzeitumgebungen, etc.).
Zeitstempel	Datum und Uhrzeit in einem definierten digitalen Format, das den genauen Zeitpunkt der Aufnahme eines Messwerts oder anderen Ereignisses (z. B. einer Aktion eines Standardnutzungsszenarios) festhält.

Zusammenfassung

Das Forschungsprojekt „Entwicklung und Anwendung von Bewertungsgrundlagen für ressourceneffiziente Software“ hat die methodischen Grundlagen erarbeitet, um die Ressourcen-Inanspruchnahme durch Software zu ermitteln, Softwareprodukte untereinander zu vergleichen und Effizienzanforderungen an diese zu stellen. Software hat einen messbaren Einfluss auf den Energieverbrauch von Hardware und kann durch steigende Hardware-Inanspruchnahme dazu beitragen, dass Hardware vorzeitig unbrauchbar wird (Obsoleszenz). Die Wirkungszusammenhänge zwischen der Nutzung von Software und der Hardware-Inanspruchnahme sind sehr komplex. Mit dem Forschungsprojekt ist es gelungen, diese Komplexität zu reduzieren und Unterschiede zwischen Softwareprodukten gleicher Funktionalität aufzuzeigen.

Das Forschungsprojekt wurde von einer Kooperationsgemeinschaft bestehend aus dem Öko-Institut e.V., der Hochschule Trier, Umwelt-Campus Birkenfeld, Institut für Softwaresysteme in Wirtschaft, Umwelt und Verwaltung und der Universität Zürich, Institut für Informatik, Forschungsgruppe Informatik und Nachhaltigkeit durchgeführt.

Die Bearbeitung der Aufgabenstellung erfolgte in folgenden Arbeitspaketen:

- Arbeitspaket 1: Methodenkonzept – Erarbeitung eines methodischen Ansatzes zur Bewertung der Umweltwirkung von Software
- Arbeitspaket 2: Anwendung der Methodik anhand von Fallbeispielen
- Arbeitspaket 3: Handlungsempfehlungen für die Entwicklung eines Umweltzeichens
- Arbeitspaket 4: Leitfaden für die Beschaffung nachhaltiger Software
- Arbeitspaket 5: Projektorganisation inklusive Projektbesprechungen und Berichte
- Arbeitspaket 6: Referenzsystem „Arbeitsplatzcomputer für Büro-Software“
- Arbeitspaket 7: Auswertungssoftware zu Hardwareauslastung und Energieverbrauch
- Arbeitspaket 8: Erfassungstool zur Erhebung der Bewertungskriterien

Aufbauend auf bestehenden Erkenntnissen zur ressourcenschonenden Softwareentwicklung und zum Softwaredesign wurde in Arbeitspaket 1 eine Bewertungsmethodik entwickelt, mit der die Software-Eigenschaften unterschiedlichen Wirkungsbereichen zugeordnet werden können (siehe Abschnitt 3 *Entwicklung von Bewertungsgrundlagen für Software*). Diese Wirkungsbereiche sind *Ressourceneffizienz*, *Potenzielle Hardware-Nutzungsdauer* und *Nutzungsautonomie*. Es wurde ein Kriterienkatalog mit insgesamt 25 Kriterien für die jeweiligen Wirkungsbereiche aufgestellt. Mit den Kriterien werden die Anforderungen an die Ressourceneffizienz von Softwareprodukten formuliert, die durch insgesamt 76 Indikatoren quantitativ und qualitativ überprüft werden können. Mit dem Wirkungsbereich *Ressourceneffizienz* soll das Ausmaß der Inanspruchnahme von Hardwareressourcen und der benötigten Energie ausgewiesen werden. Der Wirkungsbereich *Potenzielle Hardware-Nutzungsdauer* stellt den Einfluss der Software auf den Hardware-Erneuerungszyklen dar und die *Nutzungsautonomie* adressiert den Grad der Eigenständigkeit der Nutzenden im Umgang mit dem Softwareprodukt. In Tabelle 1 wird die Systematik der Bewertungskriterien dargestellt.

Tabelle 1: Systematik der Bewertungskriterien

1 Ressourceneffizienz	2 Potenzielle Hardware-Nutzungsdauer	3 Nutzungsautonomie
1.1 Hardwareeffizienz 1.2 Energieeffizienz 1.3 Ressourcenmanagement	2.1 Abwärtskompatibilität 2.2 Plattformunabhängigkeit und Portabilität 2.3 Hardwaresuffizienz	3.1 Transparenz und Interoperabilität 3.2 Deinstallierbarkeit 3.3 Wartungsfunktionen 3.4 Unabhängigkeit von Fremdressourcen 3.5 Qualität der Produktinformation

Quelle: Kriterienkatalog für nachhaltige Software, siehe Anhang 1

Der Kriterienkatalog wurde in Arbeitspaket 2 in der Praxis erprobt und auf seine Anwendbarkeit hin untersucht (siehe Abschnitt 4 *Anwendung der Bewertungsmethodik anhand von Fallbeispielen*).

Zunächst wurden für den Praxistest vier verschiedene Software-Produktgruppen ausgewählt, anhand derer der Kriterienkatalog erprobt werden sollte (siehe Abschnitt 4.2 *Auswahl der untersuchten Software*). Insgesamt wurden 11 verschiedene Softwareprodukte untersucht: zwei Textverarbeitungsprogramme, drei Internetbrowser, drei Content Management Systeme (CMS) und drei Datenbanksysteme. Weitere Angaben zu den Produktgruppen sind in Tabelle 2 aufgeführt.

Tabelle 2: Liste ausgewählter Architekturen und Softwareprodukte

#	Produktgruppe	Architektur	Plattform	Einsatzfeld	Produkte
1	Textverarbeitung	Lokale Anwendung	Desktop/ Mobile	Privat & Geschäftlich	Es wurden zwei Textverarbeitungsprogramme gewählt (TVP1 und TVP2). Bei TVP1 handelt es sich um ein proprietäres Produkt, TVP2 ist quelloffen.
2	Browser	Anwendung mit entfernter Verarbeitung	Desktop/ Mobile	Privat & Geschäftlich	Es wurden drei Internetbrowser (B1, B2 und B3) gewählt. B1 und B2 sind quelloffen, B3 ist proprietär.
3	Content Management System	Anwendung mit entfernter Verarbeitung	Desktop/ Server	Privat & Geschäftlich	Es wurden drei CMS (CMS1, CMS2 und CMS3) gewählt. Alle CMS sind quelloffen.
4	Datenbank	Serverdienst	Server	Geschäftlich	Es wurden drei Datenbanksysteme (DB1, DB2 und DB3) gewählt. DB1 und DB2 sind quelloffen, DB3 ist proprietär.

Quelle: Hochschule Trier

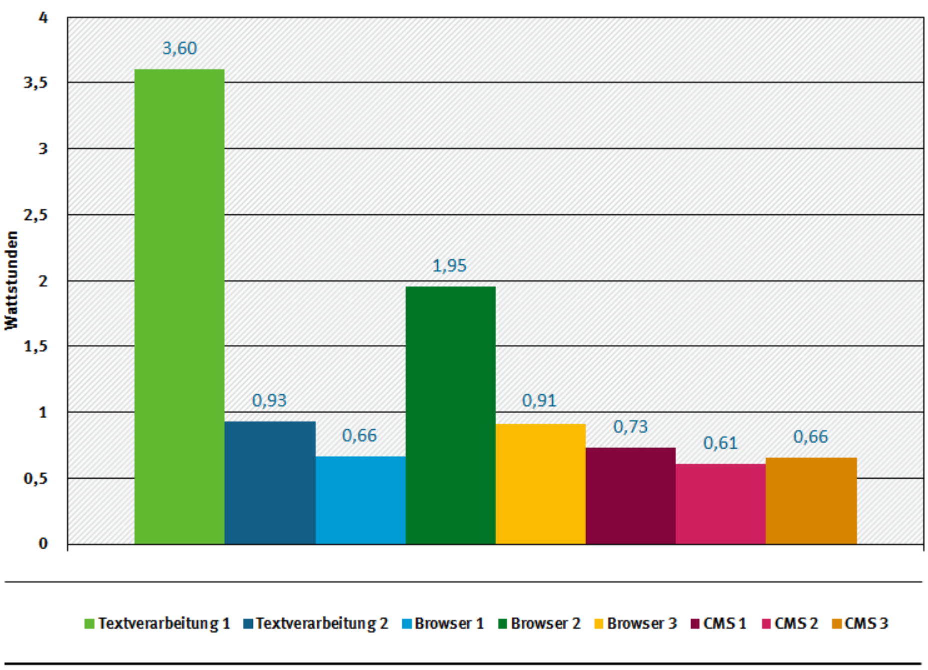
Für die zu untersuchenden Software-Produktgruppen wurden jeweils Standardnutzungsszenarien festgelegt (siehe Abschnitt 4.1.1 *Standardnutzungsszenarien*). Ein Standardnutzungsszenario beschreibt eine möglichst repräsentative Nutzung des jeweiligen Softwareprodukts über einen festgelegten Zeitraum. Konkret besteht das Standardnutzungsszenario aus einer definierten Abfolge von Befehlen und Nutzerinteraktionen (z.B. Tastatureingaben, Datenabfragen, Speichervorgängen), die für die Nutzung des Softwareproduktes typisch sind. Die Durchführung dieser Abfolge muss unabhängig vom konkreten Softwareprodukt für alle Anwendungen einer Produktgruppe möglich sein. In der Praxis wird das Standardnutzungsszenario mithilfe einer Automatisierungssoftware während des Messzyklus mehrfach (hier: 30-fach) durchlaufen und ermöglicht damit eine Reproduzierbarkeit der Messungen und Nivellierung der Messungenauigkeiten. Das Standardnutzungsszenario bildet die Bezugseinheit für alle durchgeführten Messungen des Energieverbrauchs und der Hardware-Inanspruchnahme.

Für die Durchführung der Messung der Standardszenarien wurde ein Messsystem (System Under Test) bestimmt, auf dem das zu untersuchende Softwareprodukt installiert werden konnte (siehe Abschnitt 4.1.2 *Vorgehensweise zur Messung von Hardware-Inanspruchnahme und Energieverbrauch*). Das Messsystem ist mit geeigneten Messgeräten zur präzisen Messung des Energieverbrauchs sowie mit Datenloggern zur Messung der Inanspruchnahme von Hardwarekapazitäten ausgestattet. Je nachdem, ob das zu untersuchende Softwareprodukt eine lokale Anwendung (z.B. Textverarbeitungsprogramm) oder eine Server-Anwendung (z.B. Datenbanksystem) ist, kommt als Messsystem ein Desktop-PC oder ein Server zum Einsatz. Bei einigen Anwendungen (z.B. Content-Management-Systeme) wurden die Auswirkungen auf beiden Messsystemen (client- und serverseitig) untersucht.

Die Ergebnisse der Anwendung der Bewertungsmethodik in der Praxis zeigen, dass es deutliche Unterschiede beim Energieverbrauch zwischen Softwareprodukten mit gleicher Funktionalität gibt (siehe Abschnitt 4.3 *Exemplarische Anwendungsergebnisse der Kriterien und Indikatoren*).

Für die Ausführung des Standardnutzungsszenarios für Textverarbeitungsprogramme unterscheiden sich die untersuchten Softwareprodukte beispielsweise deutlich im Energieverbrauch (siehe Abschnitt 4.3.1 *Exemplarische Messergebnisse der Kriterien 1.1.3, 1.1.4 und 1.2*). Während die Ausführung des Standardnutzungsszenarios mit dem Textverarbeitungsprogramm 1 einen Energieverbrauch von 3,6 Wattstunden auf dem lokalen Computer verursacht, sind es beim Textverarbeitungsprogramm 2 nur 0,93 Wattstunden. Obwohl beide Programme die gleichen Aufgaben erfüllen, benötigt Programm 2 nur rund ein Viertel an elektrischer Energie und ist damit deutlich energieeffizienter. In den anderen Produktgruppen zeigen sich ebenfalls Unterschiede im Energieverbrauch (Kriterium 1.2) zwischen den Programmen, die zusammenfassend in Abbildung 1 dokumentiert sind.

Abbildung 1: Vergleich der Energieverbräuche des lokalen Geräts (SUT(Client)) während der Ausführung des Standardnutzungsszenarios



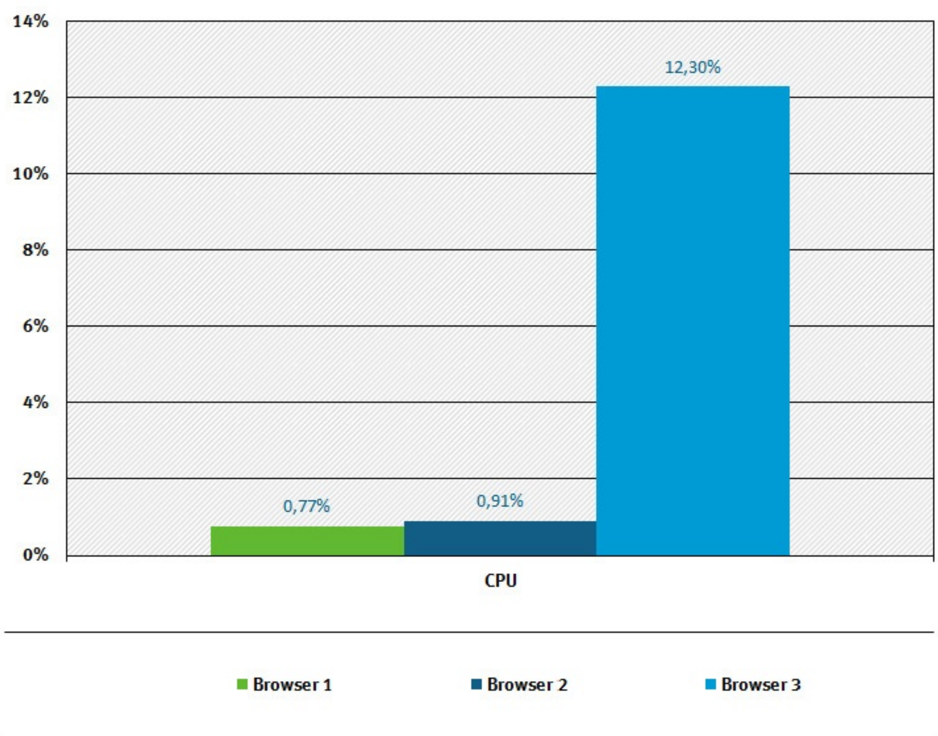
Quelle: Eigene Darstellung, Hochschule Trier

Die Ergebnisse der Messungen zeigen, dass es auch bei der Hardwareeffizienz (Prozessorauslastung, Arbeitsspeicher, Permanentspeicher, Bandbreite für Netzzugang) erkennbare Unterschiede zwischen den Softwareprodukten gibt (siehe Abschnitt 4.3.1 *Exemplarische Messergebnisse der Kriterien 1.1.3, 1.1.4 und 1.2*). Dies ist vor allem vor dem Hintergrund relevant, dass die übermäßige Beanspruchung von Hardware dazu führt, dass die Programmausführung zu lange dauert und Unternehmen bzw. öffentliche Verwaltungen und auch Privatpersonen diese vermeintlich langsame Hardware ausmustern und neue, schnellere Hardware anschaffen. Noch deutlicher ist der Effekt der übermäßigen Hardware-Inanspruchnahme, wenn Software auf bestehenden Hardwaresystemen gar nicht mehr lauffähig ist, weil die Leistungsfähigkeit bestehender Hardware zu gering ist. In beiden Fällen führt Software bzw. deren Updates zu einer Hardware-Obsoleszenz, d.h. zu einem vorzeitigen Ersatz von Geräten und damit zu einem Mehrverbrauch an Ressourcen für deren Herstellung.

Im Kriterienkatalog für nachhaltige Software (siehe Anhang 1) wurden unter *1.1 Hardwareeffizienz* verschiedene Kriterien entwickelt, mithilfe derer die Hardware-Inanspruchnahme gemessen werden kann. Beim Kriterium *1.1.3 Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration* wird beispielsweise überprüft, wie hoch die Hardwareressourcen in Anspruch genommen werden, wenn sich ein Softwareprodukt im Leerlauf befindet. Der Leerlauf beschreibt dabei den Zustand, nachdem die Software gestartet wurde, jedoch keine Nutzerinteraktion stattfindet oder Berechnungen durchgeführt werden.

Die Ergebnisse der Messungen des Kriteriums *1.1.3 a) Messung der mittleren Prozessorauslastung* sind in Abbildung 2 für drei verschiedene Internetbrowser dargestellt. Im Leerlauf belasten die Browser 1 und 2 den Prozessor (CPU) zusätzlich zur Grundlast des Messsystems mit rund 1 Prozent. Der Leerlauf von Browser 3 führt dagegen zu einer Mehrauslastung des Prozessors von 12 Prozent. Browser 3 nimmt damit die zwölffache Menge an Hardwareressourcen (bezogen auf die CPU-Auslastung) in Anspruch.

Abbildung 2: Hardware-Auslastung (CPU) im Leerlauf dreier Internetbrowser



Quelle: Hochschule Trier

Ausgehend von den Erkenntnissen des Praxistests wurde in Arbeitspaket 3 (siehe Abschnitt 5 *Handlungsempfehlungen für die Entwicklung eines Umweltzeichens*) ein reduzierter Kriterienkatalog entwickelt, der als Grundlage für mögliche Vergabekriterien für ein Umweltzeichen für Softwareprodukte herangezogen werden sollte. Vergabekriterien eines Umweltzeichens zeichnen sich insbesondere durch folgende Eigenschaften aus:

1. Vergabekriterien adressieren die wesentlichen Umweltauswirkungen eines Produktes entlang dessen Produktlebensweges.
2. Kriterien müssen richtungssicher sein, d.h. die Erfüllung der Kriterien muss Vorteile (für Mensch und Umwelt) bieten.
3. Nur solche Produkteigenschaften werden abgefragt, die zu einer Unterscheidung von Produkten beitragen, nicht solche, die von allen Produkten gleichermaßen erfüllt werden.
4. Die Anforderungen müssen mit überprüfbaren Indikatoren hinterlegt sein, die das Kriterium bestätigen (z.B. Überprüfung des Kriteriums Energieeffizienz durch Messung des Energieverbrauchs als Indikator).
5. Zur Quantifizierung der Indikatoren muss auf Prüfvorschriften verwiesen werden, die eine unabhängige und reproduzierbare Überprüfung ermöglichen (z.B. Verweis auf eine Norm oder Vorgabe einer Prüfvorschrift).
6. Formulierung einer Nachweisregelung, in welcher Form die Einhaltung gegenüber einer Vergabestelle nachgewiesen werden muss (z. B. externer Labortest oder Eigenerklärung).

Diese unter den vorgenannten Gesichtspunkten reduzierte Anzahl an Kriterien wird in Tabelle 3 dargestellt. Dabei wurde die Systematik beibehalten, dass aus allen drei Wirkungsbereichen *Ressourceneffizienz*, *Potenzielle Hardware-Nutzungsdauer* und *Nutzungsautonomie* jeweils die relevantesten Kriterien ausgewählt wurden, die den Ansprüchen an Vergabekriterien und Kriterien für eine Umweltkennzeichnung genügen. Die Kriterien unterscheiden sich darin, dass sie teilweise auf Messergebnissen beruhen (z.B. Energieeffizienz, Hardware-Auslastung), teilweise Herstellerangaben sind (z.B. Transparenz der Datenformate, Plattformunabhängigkeit) und teilweise durch Augenscheinprüfung ermittelt werden können (z.B. Verständlichkeit und Überschaubarkeit der Produktdokumentation).

Tabelle 3: Software-Kriterien zur potenziellen Anwendung in einem Umweltzeichen

Kriterium
1 Ressourceneffizienz
1.1.2 Minimale Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)
1.1.3 Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration
1.1.4 Hardware-Inanspruchnahme bei normaler Nutzung unter der Annahme einer Standardkonfiguration und eines Standardnutzungsszenarios
1.2 Energieeffizienz
2 Potenzielle Hardware-Nutzungsdauer
2.1 Abwärtskompatibilität
2.2 Plattformunabhängigkeit und Portabilität
2.3 Hardwaresuffizienz
3 Nutzungsautonomie
3.1.1 Transparenz der Datenformate und Datenportabilität
3.1.2 Transparenz und Interoperabilität der Programme
3.1.3 Kontinuität des Softwareproduktes

Kriterium
3.2.1 Deinstallierbarkeit der Programme
3.4.1 Offlinefähigkeit
3.5.1 Verständlichkeit und Überschaubarkeit der Produktdokumentation, Lizenz- und Nutzungsbedingungen

Für alle Kriterien gilt, dass für ein Umweltzeichen ein geeignetes Ambitionsniveau festgelegt werden muss. D.h. es muss festgelegt werden, ab welchem Erfüllungsgrad oder unterhalb welchen Schwellenwertes bezogen auf die einzelnen Kriterien ein Softwareprodukt als ressourceneffizient bzw. nachhaltig bezeichnet werden kann.

Die Untersuchung empfiehlt, dass zur Entwicklung von Vergabekriterien für ein Umweltzeichen „Blauer Engel“ für Softwareprodukte noch folgende Vorarbeiten geleistet werden sollten:

- Entwicklung von Standardnutzungsszenarien.
  Abhängig von der zu untersuchenden bzw. mit einem Umweltzeichen zu kennzeichnenden Software müssen geeignete Standardnutzungsszenarien entwickelt werden, die eine typische Nutzung der Software beschreiben. Diese Szenarien sollten idealerweise zusammen mit Software-Entwicklern und Software-Anwendern in Stakeholder-Dialogen entwickelt werden.
- Weitere Festlegungen zur Vereinheitlichung der Messmethoden zur Bestimmung der Hardware-Inanspruchnahme und des Energieverbrauchs von Hardware und Datenübertragung.
- Überprüfung einer größeren Anzahl an Software-Anwendungen anhand des Kriterienkataloges und Ableitung von Mindestanforderungen bzw. Benchmarks für ressourceneffiziente Software.

Für individuell programmierte Software, die beispielsweise bei der öffentlichen Hand als Fachanwendungen zum Einsatz kommt, besteht ebenfalls ein dringender Bedarf, auf deren Ressourceneffizienz zu achten. Um dies bereits kurzfristig zu ermöglichen, wurde in Arbeitspaket 5 ein Leitfaden für die Beschaffung nachhaltiger Software erstellt. Der Leitfaden richtet sich an Beschaffer von Software und erläutert die wichtigsten Kriterien aus dem Kriterienkatalog, die bereits jetzt bei der Software-Beschaffung als Leistungsanforderungen festgelegt werden können. Der Beschaffungsleitfaden soll als separate Publikation beim Umweltbundesamt veröffentlicht werden.

Da die Bewertungsmethodik nach Abschluss des Forschungsvorhabens auch durch Dritte genutzt werden soll, wurden bereits im Rahmen des Forschungsprojektes in den Arbeitspaketen 6 – 8 Hilfsmittel entwickelt, die eine Anwendung erleichtern sollen (siehe Abschnitt *6 Bausteine zur weiteren Verwendung der Bewertungsmethodik*). Diese weiteren Bausteine, die durch interessierte Software-Entwickler, Forschungseinrichtungen und Prüflabore genutzt werden können, sind:

- Referenzsystem zur Durchführung von Messungen an Büro-Software (siehe Abschnitt 6.1),
- Software zur Auswertung von Hardwareauslastung und Energieverbrauch (siehe Abschnitt 6.2; das Tool bereitet die Rohdaten von durchgeführten Messungen auf) und
- Erfassungstool zur Erhebung der Bewertungskriterien (siehe Abschnitt 6.3; durch die standardisierte Ein- und Ausgabe sowie die Entwicklung eines strukturierten XML-Schemas wird die Erfassung und Verarbeitung der Ausprägungen der Kriterien für ein konkretes Softwareprodukt vereinfacht).

Abschließend kam das Projekt zu dem Ergebnis, dass sich die Forschung im Bereich des Energie- und Ressourcenbedarfs von Software noch in einem frühen Stadium befindet (siehe Abschnitt *7 Schlussfolgerungen und weiterer Forschungsbedarf*). Trotz der wachsenden Bedeutung von Software steckt die umweltbezogene Forschung in diesem Bereich noch immer in den Kinderschuhen. Die Forschung zu nachhaltiger Software sollte deshalb insgesamt deutlich intensiviert werden. In dem Projekt wurde weiterer Forschungsbedarf identifiziert, der kurzfristig in der Entwicklung von Anforderungen für ein Umweltzeichen „Blauer Engel“ für Softwareprodukte gesehen wird. Weiterhin wird ein Forschungsbedarf bei der Bewertung des Nutzens und der Funktionalität von Software gesehen. Der Betrachtungsrahmen der Untersuchungen sollte hin zum Prozess der Softwareentwicklung inklusive der sozialverträglichen Herstellung ausgedehnt werden. Vor dem Hintergrund der Relevanz von Datenübertragung über Netzwerke sollte der Energie- und Ressourcenverbrauch innerhalb von Netzwerken weitergehend erforscht und quantifiziert werden. Software wird nicht nur in Geräten verwendet, die im Alltag als Computer wahrgenommen werden, sondern zunehmend auch in vielen anderen Geräten des täglichen Lebens (z.B. Smartphones, Haushaltsgeräte, „smarte Dinge“), die Mikroprozessoren enthalten, aber aufgrund der eingeschränkten Funktionalität nicht als Computer wahrgenommen werden. Die Bewertungsmethodik für Software sollte auch für weitere Geräte anwendbar gemacht werden. Aufgrund der zunehmenden Relevanz des „Internet of Things“ und der Blockchain-Technologie wird außerdem empfohlen, weitere Forschungen im Bereich des Energie- und Ressourcenverbrauchs durch die Nutzung solcher Technologie mit dezentraler Datenspeicherung und -verarbeitung durchzuführen.

Summary

The research project " Development and application of criteria for resource-efficient software products with consideration of existing methods" has developed the methodological basis for determining the use of resources by software, comparing software products with each other and making efficiency demands on them. Software has a measurable influence on the energy consumption of hardware and can contribute to hardware becoming prematurely due for replacement (obsolescence) because of increasing hardware consumption. The interdependencies between the use of software and the use of hardware are very complex. The research project has succeeded in reducing this complexity and revealing differences between software products with the same functionality.

The research project was carried out by a cooperation group consisting of Oeko-Institut e.V., Hochschule Trier, Umwelt-Campus Birkenfeld, Institute for Software Systems in Business, Environment and Administration and the University of Zurich, Department of Informatics, Research Group Informatics and Sustainability.

The task was processed in the following work packages:

- Work package 1: Methodological concept – Development of a methodical approach to evaluate the environmental impact of software
- Work package 2: Application of the methodology based on case studies
- Work package 3: Recommendations for the development of an eco-label
- Work package 4: Guide for the procurement of sustainable software
- Work package 5: Project organization including project meetings and reports
- Work package 6: Reference system "Desktop computer for office software"
- Work package 7: Evaluation software for hardware utilization and energy consumption
- Work package 8: Collecting tool for assessing the evaluation criteria

Based on existing findings on resource-saving software development and software design, an evaluation methodology was developed in work package 1 by means of which the software properties can be assigned to different areas of impact (see section 3), namely *resource efficiency, potential hardware life and user autonomy*. A catalogue of criteria with a total of 25 criteria was drawn up for the individual impact areas. The criteria are used to formulate the requirements for the resource efficiency of software products, which can be quantitatively and qualitatively verified on the basis of a total of 76 indicators. The *resource efficiency* area of impact is intended to identify the extent to which hardware resources are used and energy is required. The *potential hardware useful life* area of impact represents the influence of the software on the hardware renewal cycles, and *user autonomy* addresses the degree of autonomy of the user in dealing with the software product. Table 1 shows the classification of the evaluation criteria.

Table 1: Classification of evaluation criteria

1	Resource efficiency	2	Potential hardware useful life	3	User autonomy
1.1	hardware efficiency	2.1	backward compatibility	3.1	transparency and interoperability
1.2	energy efficiency	2.2	platform independence and portability	3.2	uninstallability
1.3	resource management	2.3	hardware sufficiency	3.3	maintenance functions
				3.4	independence of outside resources
				3.5	quality of product in formation

Source: Criteria catalogue for sustainable software, see Anhang 1

The criteria catalogue was tested in reality in work package 2. Furthermore, its applicability was examined (see section 4) in the framework of this work package.

First of all, four different software product groups were selected for the practical test with the aim of testing the catalogue of criteria (see section 4.2). In total, 11 different software products were examined: two word processing programs, three Internet browsers, three content management systems and three database systems. Further information on the product groups is included in the following table.

Table 2: List of selected architectures and software products

#	Product group	Architecture	Platform	Application area	Products
1	text processing	local application	desktop/ mobile	private & business	Two text processing programs have been selected (TVP1 and TVP2). TVP1 is a proprietary product, while TVP2 is an open source system.
2	browser	application with remote processing	desktop/ mobile	private & business	Three Internet browsers (B1, B2 and B3) have been selected. B1 and B2 are open source systems, B3 is a proprietary product.
3	content management system	application with remote processing	desktop/ server	private & business	Three CMS programs (CMS1, CMS2 und CMS3) have been selected. All CMS are open source systems.
4	database	server service	server	business	Three database systems (DB1, DB2 und DB3) have been selected. DB1 und DB2 are open source systems, DB3 is a proprietary product.

Source: Hochschule Trier

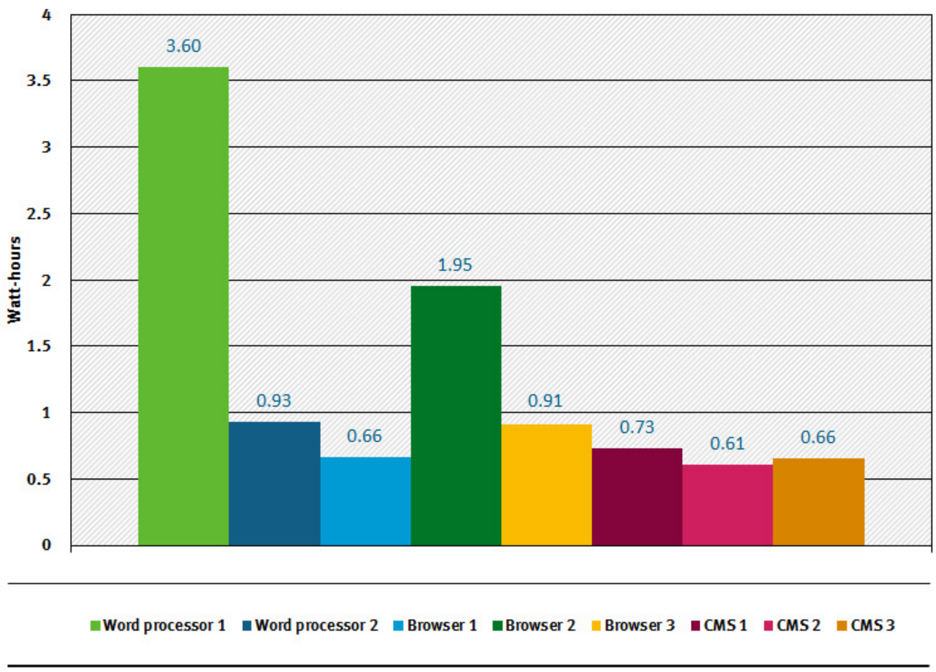
Standard usage scenarios were defined for the software product groups to be examined (see section 4.1.1). A standard usage scenario describes the most representative use of the respective software over a defined period of time. In concrete terms, the standard usage scenario consists of a defined sequence of commands and user interactions (e.g. keyboard entries, data queries, storage processes) that are typical for the use of the software product. It must be possible to apply this sequence for all applications of a product group, regardless of the specific software product. In practice, the standard usage scenario is run through several times (here: 30 times) with the aid of automation software during the measurement cycle, thus enabling reproducibility of the measurements and levelling of the measurement inaccuracies. The standard usage scenario is the reference unit for all measurements of energy consumption and hardware usage.

A measurement system (System Under Test) was specified on which the software to be tested could be installed (see section 4.1.2). The measuring system is equipped with suitable measuring devices for the precise measurement of energy consumption and data loggers for measuring the use of hardware capacities. Depending on whether the software to be examined is a local application (e.g. word processing program) or a server application (e.g. database system), a desktop PC or a server is used as the measuring system. In some applications (e.g. content management systems), the effects on both measuring systems (client and server side) were examined.

In actual operation, the results of the application of the evaluation methodology point to clear differences in energy consumption between software products with the same functionality (see section 4.3).

When executing the standard use scenario for word processing programs, for example, the software products examined differ significantly in energy consumption (see section 4.3.1). While the execution of the standard usage scenario with word processing program 1 results in an energy consumption of 3.6 watt hours on the local computer, it reveals only 0.93 watt hours when using word processing program 2. Although both programs perform the same tasks, program 2 requires only about a quarter of the electrical energy and is therefore significantly more energy-efficient. The other product groups also show differences in energy consumption (criterion 1.2) between the programs, which are summarised in Figure 1.

Figure 1: Comparison of energy consumption of the local device (SUT(Client)) during the execution of the standard usage scenario



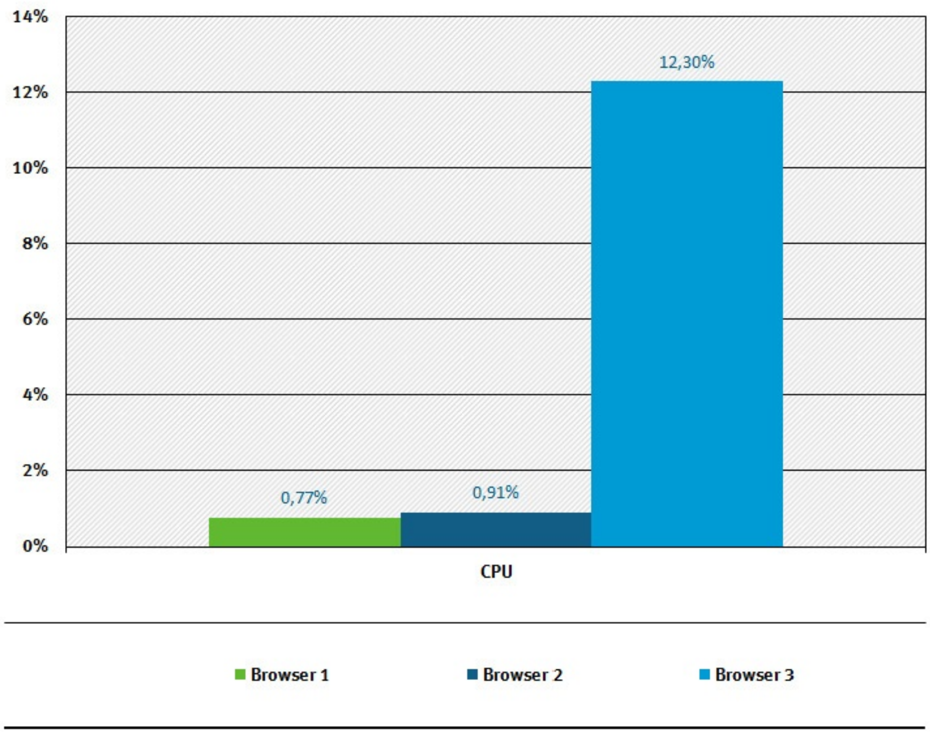
Source: Hochschule Trier

The results of the measurements show that there are also discernible differences between the software products in terms of hardware efficiency (processor utilization, working memory, permanent storage, bandwidth for network access) (see section 4.3.1). This is particularly relevant against the background that the excessive use of hardware leads to program execution taking too long, and companies, administrations or private users taking this supposedly slow hardware out of service and purchasing new, faster hardware. The effect of excessive use of hardware is even more pronounced when software can no longer run on existing hardware systems because the performance of existing hardware is too low. In both cases, software and its updates lead to hardware obsolescence, i.e. premature replacement of devices and thus to an increased consumption of resources used for their production.

In the criteria catalogue for sustainable software (see Annex 1), various criteria have been developed under *1.1 Hardware efficiency*, which can be used to measure hardware usage. For example, criterion *1.1.3 Hardware utilization in idle mode assuming a standard configuration* checks the utilization levels of hardware resources if a software is in idle mode. Idle describes the state after the software has been started, but in which no user interaction takes place or calculations are performed.

The results of the measurements of criterion 1.1.3 *(a) Measurement of average processor utilization* are shown in Figure 2 for three different web browsers. When idle, browsers 1 and 2 increase the processor load (CPU) by around 1 percent in addition to the base load of the measurement system. The idle mode of Browser 3, on the other hand, leads to an additional utilization of the processor of 12 percent. Browser 3 uses twelve times the amount of hardware resources (based on CPU utilization).

Figure 2: Hardware Utilization (CPU) of three web browsers in idle mode



Source: Own illustration, Hochschule Trier

Based on the findings of the practical test, a reduced catalogue of criteria was developed in work package 3 (see section 5), which should be used as a basis for possible criteria for the award of an eco-label for software. Criteria for the award of an eco-label are particularly characterised by the following characteristics:

1. Award criteria address the significant environmental impacts of a product along its life cycle.
2. Criteria must provide strategic direction, i.e. the fulfilment of the criteria must offer advantages (for human and environment).
3. Only those product characteristics are queried which contribute to a differentiation of products, not those which are equally fulfilled by all products.
4. The requirements must be accompanied by verifiable indicators confirming the criterion (e.g. verification of the energy efficiency criterion by measuring energy consumption as an indicator).
5. For the quantification of indicators, reference shall be made to test specifications which enable independent and reproducible testing (e.g. reference to a standard or specification of a test regulation).
6. Formulating a verification rule in which compliance must be proven to an awarding body (e.g. external laboratory test or self-declaration).

These criteria – the number of which has been reduced in accordance with the considerations mentioned above – are presented in Table 3. The overall approach of selecting the most relevant criteria from each of the three impact areas, i.e. *resource efficiency, potential hardware operating life* and *user autonomy* has been maintained, the criteria having to comply with the specifications laid down for award criteria and Ecolabel criteria. The criteria differ in that they are partly based on measurement results (e.g. energy efficiency, hardware utilization), partly manufacturer information (e.g. transparency of data formats, platform independence) and partly can be determined by visual inspection (e.g. comprehensibility and manageability of product documentation).

Table 3: Software criteria for the potential application in an eco-label

Criterion
1 Resource efficiency
1.1.2 Minimum system requirements and resulting hardware requirements (incl. peripheral devices)
1.1.3 Hardware utilization in idle mode assuming a standard configuration
1.1.4 Hardware utilization during normal use assuming a standard configuration and a standard usage scenario
1.2 Energy efficiency
2 Potential useful life of hardware
2.1 Backward compatibility
2.2 Platform independence and portability
2.3 Hardware sufficiency
3 User autonomy
3.1.1 Transparency of data formats and data portability
3.1.2 Transparency and interoperability of the programs
3.1.3 Continuity of the software product
3.2.1 Uninstallability of programs
3.4.1 Offline capability
3.5.1 Comprehensibility and manageability of product documentation, licensing conditions and terms of use

For all criteria, an appropriate level of ambition must be set for an eco-label. This means that it must be determined from which degree of fulfilment or below which threshold value in relation to the individual criteria a software product can be described as resource-efficient or sustainable.

The study recommends that the following preliminary work should be carried out to develop criteria for the award of the "Blue Angel" eco-label to software products:

- Development of standard usage scenarios.
  Depending on the software to be examined or to be labelled with an eco-label, suitable standard usage scenarios depicting typical software utilization must be developed. Ideally, these scenarios should be developed in stakeholder dialogues in collaboration with software developers and software users.
- Further specifications to standardize the measurement methods for determining the hardware utilization and the energy consumption of hardware and data transmission.
- Review of a larger number of software applications based on the catalogue of criteria and derivation of minimum requirements or benchmarks for resource-efficient software.

There is also an urgent need to pay attention to resource efficiency for individually programmed software, which is used, for example, by the public sector as special applications. In order to make this possible at short notice, a guideline for the procurement of sustainable software was drawn up in work package 5. The guide is aimed at purchasers of software and explains the most important criteria from the criteria catalogue, which can already be defined as performance requirements in software procurement. The procurement guideline is to be published separately at the Federal Environment Agency.

Since the evaluation methodology will also be used by third parties after completion of the research project, tools designed to facilitate its application have already been developed as part of the research project in the framework of work packages 6-8 (see section 6). These other modules, which can be used by interested software developers, research institutions and testing laboratories, are:

- Reference system for the implementation of measurements on office software (see section 6.1),
- Software for the evaluation of hardware utilization and energy consumption (see section 6.2) and
- Collecting tool for assessing the evaluation criteria (see section 6.3)

Finally, the study concluded that research into the energy and resource requirements of software is still at a very early stage (see section 7). Despite the growing importance of software, environmental research in this area is still in its infancy. The overall research into sustainable software should therefore be stepped up to a considerable extent. The project identified further research needs which, in the short term, are deemed necessary for developing requirements for a "Blue Angel" eco-label for software products. Furthermore, it is considered necessary to push forward the assessment of the use and functionality of software. The scope of the investigations should be extended to the process of software development including the social standards and working conditions for designing and programming. Against the background of the relevance of data transmission via networks, energy and resource consumption within networks should be further researched and quantified. Software is not only used in so called computers, but of course also in many other devices of daily life (e.g. smartphones, household appliances, "smart things"). The evaluation methodology for software should also be made applicable to other devices. Due to the increasing relevance of the *internet of things* and *blockchain technology*, the execution of further research is also recommended for the field of energy and resource consumption by using these technologies with the help of decentralized data storage and processing.

1 Hintergrund und Zielsetzung des Vorhabens

Mit dem Forschungsprojekt „Entwicklung und Anwendung von Bewertungsgrundlagen für ressourceneffiziente Software unter Berücksichtigung bestehender Methodik“ setzt das Umweltbundesamt seine anwendungsbezogene Forschung im Bereich der nachhaltigen Software fort. Diese wurde in den Jahren 2011 bis 2013 mit dem Forschungsvorhaben „Grüne Software: Ermittlung und Erschließung von Umweltschutzpotenzialen der Informations- und Kommunikationstechnik (Green IT)“ begonnen und durch ein Fachgespräch „Nachhaltige Software“ im November 2014 weiter verstetigt. Das vorliegende Forschungsprojekt „Entwicklung und Anwendung von Bewertungsgrundlagen für ressourceneffiziente Software unter Berücksichtigung bestehender Methodik“ baut auf den vorangehenden Arbeiten auf und entwickelt eine Methodik zur Bewertung von Software.

Bei der Ermittlung der Umweltauswirkungen von Informations- und Kommunikationstechnik (IKT) liegt der Fokus der Forschung, der öffentlichen Diskussion und der Herstelleraktivitäten bislang schwerpunktmäßig auf der Hardware und deren Energie- und Ressourceninanspruchnahme. So gibt es eine Reihe von Umweltkennzeichnungen (z. B. EnergyStar, EPEAT, Blauer Engel), freiwillige Selbstverpflichtungen (z. B. Code-of-Conducts) und gesetzliche Mindestanforderungen (z. B. Ökodesign-Richtlinien), die auf energiesparende und teilweise auf ressourcenschonende IKT-Hardware abzielen. Für nachhaltige oder effiziente Software dagegen gibt es bislang wenig Aktivitäten, keine einheitliche Definition oder allgemein akzeptierte Standards. Dennoch ist sich die Expertengemeinschaft einig, dass Software von ebenso großer Bedeutung für die Nachhaltigkeit eines IKT-Systems ist wie die Hardware. Wachsende Datenmengen und immer komplexere Programme erfordern die ständige Neuanschaffung von Computern und mobiler IKT sowie den beständigen Ausbau von Übertragungsnetzen, Datenspeichern und Rechenzentren.

Das Forschungsvorhaben soll daher den Fokus auf Software und deren Einfluss auf den Energieverbrauch und die Inanspruchnahme von Ressourcen legen. Hierzu werden zunächst vorhandene methodische Ansätze zur Bewertung von Software systematisiert und der Einfluss der Software auf Energieverbrauch und Hardwareinanspruchnahme ermittelt. Im Ergebnis des Forschungsvorhabens sollen Kriterien benannt werden, nach denen Software auf ihre Umweltverträglichkeit hin untersucht und bewertet werden kann. Perspektivisch sollen diese Kriterien dazu geeignet sein, die Anforderungen eines Umweltkennzeichens, beispielsweise eines Blauen Engels für Software, abzuleiten. Im Forschungsvorhaben werden erste Vorschläge für Kriterien und deren Nachweisregelung entwickelt. Anhand eines Leitfadens werden die Erkenntnisse des Vorhabens als Arbeitshilfe für Beschaffung von energie- und ressourceneffizienter Software zusammengestellt.

2 Bearbeitungskonzept

Das Forschungsprojekt „Entwicklung und Anwendung von Bewertungsgrundlagen für ressourceneffiziente Software unter Berücksichtigung bestehender Methodik“ wird von einer Kooperationsgemeinschaft bestehend aus dem Öko-Institut e.V., der Hochschule Trier, Umwelt-Campus Birkenfeld, Institut für Softwaresysteme in Wirtschaft, Umwelt und Verwaltung und der Universität Zürich, Institut für Informatik, Forschungsgruppe Informatik und Nachhaltigkeit durchgeführt.

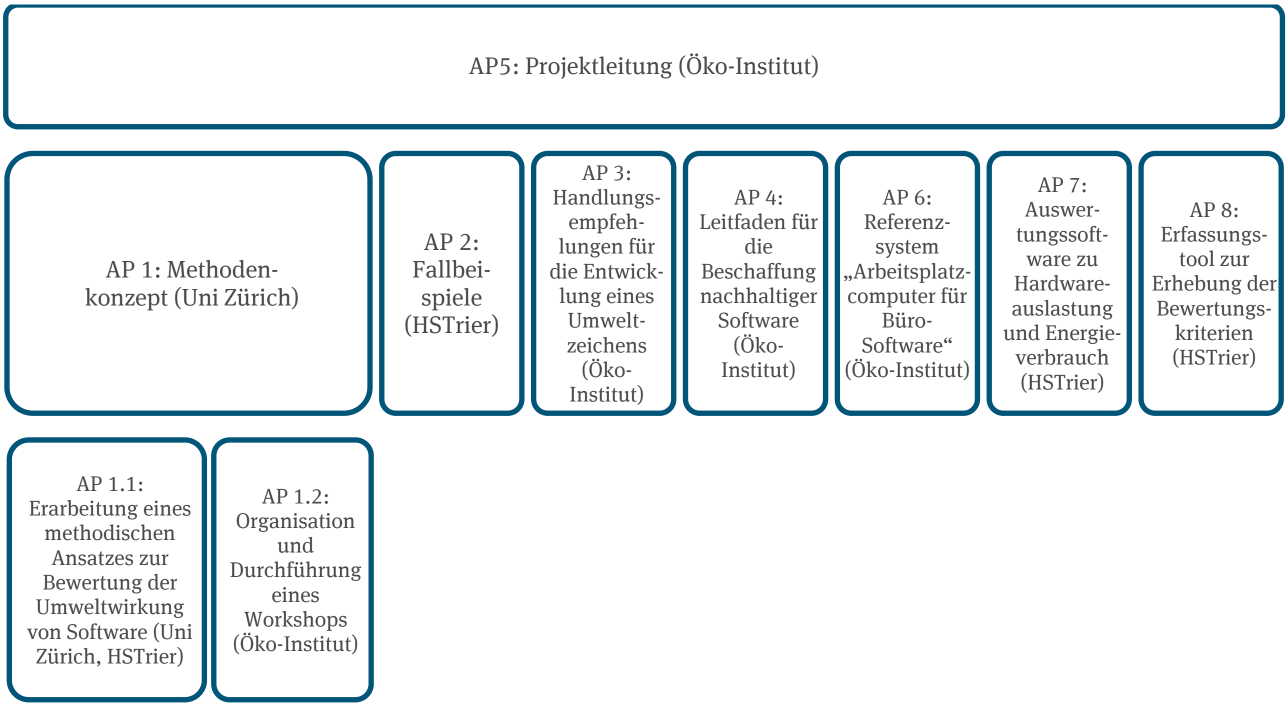
Die Bearbeitung der Aufgabenstellung erfolgt in folgenden Arbeitspaketen:

- Arbeitspaket 1.1: Methodenkonzept – Erarbeitung eines methodischen Ansatzes zur Bewertung der Umweltwirkung von Software
- Arbeitspaket 1.2: Methodenkonzept – Organisation und Durchführung eines Workshops
- Arbeitspaket 2: Anwendung der Methodik anhand von Fallbeispielen
- Arbeitspaket 3: Handlungsempfehlungen für die Entwicklung eines Umweltzeichens

- Arbeitspaket 4: Leitfaden für die Beschaffung nachhaltiger Software
- Arbeitspaket 5: Projektorganisation inklusive Projektbesprechungen und Berichte
- Arbeitspaket 6: Referenzsystem „Arbeitsplatzcomputer für Büro-Software“
- Arbeitspaket 7: Auswertungssoftware zu Hardwareauslastung und Energieverbrauch
- Arbeitspaket 8: Erfassungstool zur Erhebung der Bewertungskriterien

Die Aufteilung der Arbeitspakete innerhalb der Kooperationsgemeinschaft ist aus der nachfolgenden Abbildung 3 ersichtlich:

Abbildung 3: Aufteilung der Arbeitspakete im Projektteam



Der Bearbeitungszeitraum des Forschungsprojekts lag zwischen Juli 2015 und Februar 2018.

Die Entwicklung des Methodenkonzeptes (Arbeitspaket 1.1) ist in Abschnitt 3 *Entwicklung von Bewertungsgrundlagen für Software* dokumentiert. Die Erkenntnisse aus dem Workshop (Arbeitspaket 1.2) sind in die Überarbeitung des Kriterienkatalogs (siehe Anhang 1) eingeflossen.

Die Ergebnisse der Überprüfung des Kriterienkataloges (Arbeitspaket 2) ist in Abschnitt 4 *Anwendung der Bewertungsmethodik anhand von Fallbeispielen* festgehalten.

Die Schlussfolgerungen für ein Umweltzeichen (Arbeitspaket 3) sind in Abschnitt 5 *Handlungsempfehlungen für die Entwicklung eines Umweltzeichens* dokumentiert.

Ein exemplarisches Referenzsystem (Arbeitspaket 6) wird in Abschnitt 6.1 Referenzsystem „Arbeitsplatzcomputer für Büro-Software“ vorgestellt.

Die beiden Software-Werkzeuge werden in den *Abschnitten* 6.2 *Auswertungssoftware zu Hardwareauslastung und zum* Energieverbrauch und 6.3 *EXCEL-Tool zur Erfassung der Bewertungskriterien* kurz dargestellt.

Der Leitfaden für die Beschaffung nachhaltiger Software (Arbeitspaket 4) sowie die beiden Software-Werkzeuge (Arbeitspakete 7 und 8) werden darüber hinaus als Anhänge zum Forschungsbericht veröffentlicht.

3 Entwicklung von Bewertungsgrundlagen für Software

Die Erarbeitung einer Methodik zur Bewertung von Software erfolgt in folgenden Bearbeitungsschritten:

1. Präzisierung der Zielsetzung
2. Wirkungsmodell von Software
3. Klassifikation von Softwareprodukten
4. Bestandsaufnahme
5. Bewertung und Ergänzung der bestehenden Methoden und Modelle
6. Integration und Komplexitätsreduktion
7. Entwurf eines Kriterienkataloges für nachhaltige Software

3.1 Präzisierung der Zielsetzung

Zur Präzisierung der Zielsetzung wurden durch das Projektteam folgende Forschungsfragen formuliert:

1. *Wie lassen sich auf Basis der Fachliteratur und des erarbeiteten „Wirkungsmodells für Softwareprodukte" Kriterien zur Beurteilung der Ressourcenbeanspruchung durch Softwareprodukte definieren?*

2. *Wie lassen sich diese Kriterien operationalisieren, d.h. durch Indikatoren ergänzen, die sich systematisch durch Messungen am Softwareprodukt oder auf andere Weise einschätzen lassen?*

3. *Wie lassen sich die Kriterien zu einem Bewertungsmodell kombinieren, das zur vergleichenden oder absoluten Beurteilung des von einem Softwareprodukt verursachten Ressourcenverbrauchs, insbesondere für die Vergabe eines Labels, geeignet ist?*

Im Rahmen des hier durchgeführten Forschungsprojektes stehen die einzelnen Bewertungskriterien für nachhaltige Software zunächst gleichrangig und nicht gewichtet nebeneinander. Das heißt, es wird nicht versucht, eine Aggregation der Bewertungskriterien, beispielsweise in Form von Bewertungspunkten, vorzunehmen. Hierfür müssten noch eine Vielzahl an weiteren methodischen Fragestellungen geklärt werden, wie beispielsweise die Abwägung unterschiedlicher Schutzgüter untereinander (z. B. Klima- und Ressourcenschutz vs. Datenschutz und Nutzungsautonomie) oder die Unterscheidung verschiedener Anwendungsszenarien der Software.

Weiterhin wurden auf der Grundlage von Literaturrecherchen sowie Diskussionsprozessen im Projektteam und mit dem Umweltbundesamt folgende Einschränkungen und Präzisierungen des Untersuchungsrahmens vorgenommen:

1. Die oben formulierten Fragen werden, soweit immer möglich, aus einer Perspektive beantwortet, die räumlich und zeitlich über den momentanen und lokalen Einsatz des Softwareprodukts hinausgeht. Berücksichtigt werden also auch, zumindest in sehr groben Abschätzungen, die Beanspruchung von Ressourcen durch ausgelösten Datenverkehr im Internet und die in entfernten Hardwarekomponenten (insbesondere Servern) verursachten Lasten. Anderenfalls wäre eine Erfüllung der Kriterien durch Auslagerung von Ressourcenbedarf in „die Cloud" leicht möglich. Ebenso soll nach Möglichkeit die zeitliche Entwicklung des Softwareprodukts, z. B. die Vermeidung von „Software Bloat" (Aufblähen des Produkts von Version zu Version) durch den Hersteller, berücksichtigt werden. Der beispielsweise durch Updates oder neueren Versionen von Softwareprodukten motivierte Ersatz funktionierender Hardware durch neue Hardware ist einer der wichtigsten

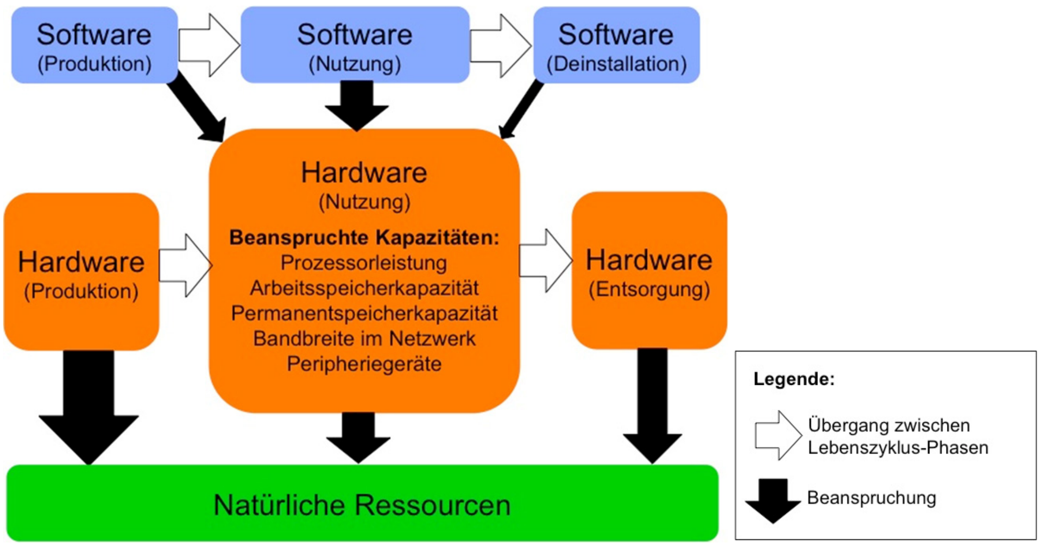
Wirkungspfade der durch Software verursachten Ressourcenverbräuche und soll deshalb nicht vernachlässigt werden, selbst wenn die Kriterien zum Teil qualitativ bleiben.

2. Im Zentrum der Betrachtung stehen a) die für den Betrieb des betrachteten Softwareprodukts bereitzuhaltenden Hardwarekapazitäten und b) deren Energieverbrauch während der Nutzung. Die Phase der Softwareentwicklung ist insofern relevant, als Empfehlungen für die Entwicklung aus den Kriterien abgeleitet werden können. Bei der Anwendung der Kriterien wird jedoch ausschließlich die Nutzungsphase des Produkts betrachtet. Zur Beurteilung des Softwareprodukts werden also nur dessen Eigenschaften im laufenden Betrieb herangezogen.
3. Es wird die Ressourcenbeanspruchung im Sinne von Aufwand (Effekte erster Ordnung) betrachtet. Auswirkungen des vom Softwareprodukt erbrachten Nutzens werden nicht beurteilt. Aspekte aus dem Feld „Green by IT“ (Effekte zweiter Ordnung) werden somit nicht in die Erarbeitung der Kriterien mit einbezogen. Direkt vergleichbar werden dadurch nur Softwareprodukte mit ähnlicher Funktionalität. Das schließt nicht aus, dass der Auftraggeber beispielsweise für die Ableitung von Kriterien für ein Umweltzeichens zusätzlich zum entwickelten Kriterienkatalog ethische Kriterien berücksichtigt, die Software für bestimmte Anwendungen ausschließen (z. B. wegen Verletzung sozialer Standards).
4. Das Projekt ist auf Anwendungssoftware für Standardaufgaben fokussiert, Systemsoftware und speziell entwickelte Individualsoftware liegen somit nicht im Fokus der Kriterienentwicklung.
5. Softwaresysteme mit starker Hardware-Software-Verschränkung, beispielsweise Firmware von Druckern oder anderer Peripheriegeräte, werden nicht betrachtet.
6. Eine Quantifizierung des Einsparpotenzials an natürlichen Ressourcen in physikalischen Einheiten durch Einhaltung der Kriterien ist nicht vorgesehen.
7. Es wird nur die Auswirkung von Anwendungssoftware für Standardaufgaben auf ausgewählte Referenzhardware betrachtet. Gegebenenfalls zur Ausführung benötigte Dienste anderer Softwareprodukte und entfernter Hardware (bei Client-Software insbesondere auf Server-Seite) werden hierbei nur grob abgeschätzt. Dies gilt ebenfalls für die Netzwerk-Ebene.
8. Als Verbrauch der Hardware in der Nutzungsphase wird nur der direkte Energieverbrauch betrachtet, nicht-energetische Verbräuche, insbesondere durch Peripheriegeräte (wie z. B. Papierverbrauch durch Drucker) werden nur ausnahmsweise in die Betrachtung einbezogen.
9. Der Ressourcenverbrauch des Entwicklungsprozesses wird nicht betrachtet.
10. Es wird keine quantitative Analyse mit dem Ziel durchgeführt, zu identifizieren, welche Teile eines Softwareprodukts, welche Aspekte der Softwareentwicklung, welche Architekturmodelle oder welche Softwareprodukte die größten Potenziale bieten, durch gezielte Veränderung Ressourcen einzusparen.
11. Das Bewertungsmodell, das die Indikatoren und Kriterien verknüpft, wird im Rahmen des Projekts nur der Form nach und exemplarisch definiert. Die für die Vergabe eines Labels geltende Gewichtung der Kriterien und die genauen Aggregationsvorschriften sind vom Auftraggeber festzulegen.
12. Die in Arbeitspaket 4 (Leitfaden für die Beschaffung nachhaltiger Software) zu entwickelnden Handlungsempfehlungen richten sich an die Beschaffer von Software. Die für die Beschaffung von Software anwendbaren Kriterien werden erläutert, damit sie als Mindestanforderungen an Lieferanten und Entwickler von Software weitergereicht werden können.

3.2 Wirkungsmodell von Software

Umweltwirkungen eines Produkts entstehen generell durch die Beanspruchung natürlicher Ressourcen[1] entlang des Lebenszyklus eines Produkts (z. B. Computer). Dieser Lebenszyklus umfasst die Gewinnung von Rohstoffen, den Transport, die Produktion, die Nutzung des Produkts sowie die Entsorgung bzw. die Abfallbehandlung. Im Rahmen dieses Vorhabens wird diese Lebenszyklusperspektive auch in Bezug auf Softwareprodukte eingenommen. Dabei wird insbesondere berücksichtigt, dass die für den Betrieb eines Softwareprodukts benötigte Hardware produziert, mit elektrischer Energie versorgt und am Ende ihrer Nutzungsdauer entsorgt werden muss (siehe Abbildung 4). Jedes Softwareprodukt beansprucht somit einen quantifizierbaren Anteil am Lebenszyklus aller zu seiner Entwicklung und seinem Betrieb nötigen physischen Hardwareprodukte (programmierbare Geräte in jeglicher Form, Peripheriegeräte und Datenträger) sowie deren Energie- und Rohstoffverbrauch bei der Nutzung.

Abbildung 4: Lebenszyklen von Hardware und Software (horizontale Dimension) und Beanspruchung von Ressourcen (vertikale Dimension)



Quelle: Eigene Darstellung, Universität Zürich

Aufgrund der Lebenszyklusperspektive ist dieser Ansatz grundsätzlich auch für eine Erweiterung in Richtung der sozialen Aspekte der Rohstoffgewinnung und der Arbeitsbedingungen in der Hardwareproduktion oder -entsorgung geeignet. Der Fokus im vorliegenden Vorhaben liegt jedoch auf den Umweltaspekten.

Auf Softwareebene wird die Perspektive im Folgenden bewusst auf die Nutzungsphase der Software begrenzt. Ziel der hier zu definierenden Kriterien wird es sein, ein Softwareprodukt aufgrund von Eigenschaften zu bewerten, die in der Nutzungsphase beobachtbar sind. Sei es durch die Nutzenden selbst oder durch Personen, die spezielle Tests durchführen. Eine zusätzliche Berücksichtigung der Produktionsphase der Software wäre aber durch Erweiterung des Ansatzes prinzipiell möglich.

[1] Definitionen von „Ressource“ und anderen zentralen Begriffen sind im Glossar zu finden. Der Begriff „Ressource“ steht in diesem Forschungsvorhaben für natürliche Ressourcen und nicht für den technischen Begriff der „Hardwareressourcen“. Letztere werden hier präziser als „Hardwarekapazitäten“ bezeichnet.

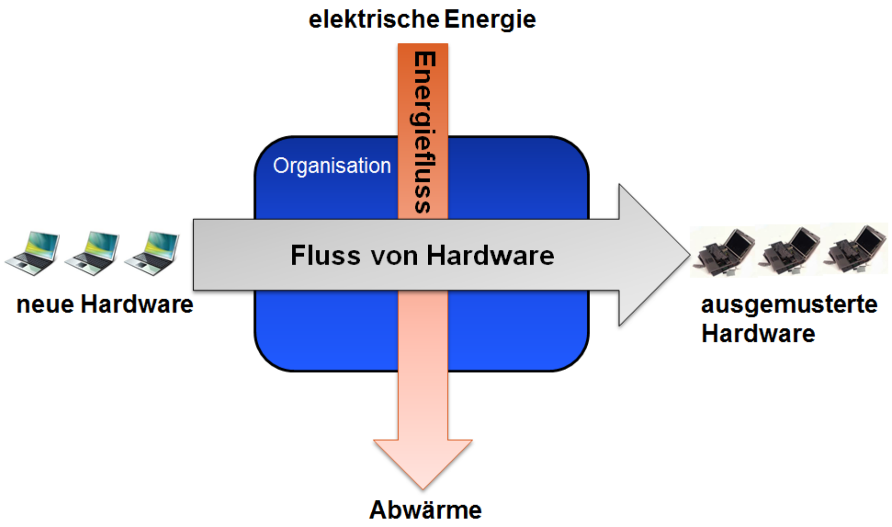
Bei der Bewertung von Softwareprodukten ist grundsätzlich nicht nur eine Momentaufnahme, sondern die Nutzung des Softwareprodukts über längere Zeiträume (insbesondere auch die Nutzung mehrerer Versionen) zu betrachten. Erst aus dieser Perspektive wird z. B. die Frage der durch Software induzierten Beschaffung von Hardware relevant (insbesondere durch Software bedingte Obsoleszenz von Hardware).

Abstrakt formuliert, stehen zwei durch ein Softwareprodukt verursachte materielle Flüsse (an natürlichen Ressourcen bzw. den daraus gewonnenen Produkten) im Vordergrund:

- der Durchfluss von Hardware durch die nutzende Organisation, z.B. ein Unternehmen oder einen Privathaushalt (Hardwarefluss: funktionstüchtige Hardware wird zu Abfall),
- der Durchfluss von Energie durch die Hardware (Energiefluss: Elektrizität wird zu Abwärme).

Dieser Durchfluss von Hardware und Energie wird in Abbildung 5 illustriert. Die Ränder der in der Abbildung als „Organisation“ bezeichnete Betrachtungseinheit (z.B. eine Firma, eine Verwaltung, ein Büro, ein Haushalt) stellen die Systemgrenze im Sinne einer Ökobilanz dar. Die genutzte Software hat einen Einfluss darauf, in welcher Menge neue Hardware und elektrische Energie durch die Betrachtungseinheit fließt. Wenn ein Softwareprodukt im Vergleich zu Konkurrenzprodukten mit ähnlicher Funktionalität einen deutlich geringeren Hardware- und Energiefluss verursacht, kann es als (relativ) „nachhaltig“ gelten.[2]

Abbildung 5: Durchfluss von Energie und Hardware durch eine Organisation



Quelle: Eigene Darstellung, Universität Zürich

Der Fluss von Hardware kann mit Methoden der Ökobilanzierung von Produkten (Life Cycle Assessment, LCA) hinsichtlich seiner Ressourcenbeanspruchung eingeschätzt werden. Hierfür gibt es Lebenszyklusinventare zur Produktion und Entsorgung der wichtigsten Hardwarekomponenten, die im Rahmen dieses Vorhabens nicht weitergehend spezifiziert, sondern als zugänglich vorausgesetzt werden. Ebenso kann der Energiefluss mit Methoden der Ökobilanzierung bewertet werden. Auch Metho-

[2] Die Funktionalität eines Softwareprodukts und damit sein *Nutzen* werden hier nicht bewertet. Ziel ist es ausschließlich, den ausgelösten *Aufwand* (an Ressourcen) einzuschätzen und zu bewerten. Bei gegebenem, insbesondere über verschiedene Softwareprodukte äquivalentem Nutzen lässt sich dieser zum Aufwand ins Verhältnis setzen, um die *Effizienz* zu ermitteln.

den zur Bewertung der Stromerzeugung sind ausreichend bekannt, weshalb die entsprechenden Daten (z. B. CO_2-Emissionen pro Kilowattstunde elektrischer Energie) als gegeben vorausgesetzt werden.

Aus den vorgenannten Gründen ist es ausreichend, mit den hier entwickelten Bewertungskriterien den Einfluss von Software auf die benötigten Hardwarekapazitäten sowie auf deren Energieverbrauch zu adressieren. In dieser Studie werden die Hardwareinanspruchnahme und deren Stromverbrauch bei der Nutzung der Software betrachtet, denn nur dieser ist für den Untersuchungsgegenstand spezifisch.[3] Für die Bewertung der Hardware- und Energieflüsse werden Lebenszyklusinventare für Hardwarekomponenten und Methoden der ökologischen Bewertung benötigt, die bereits existieren und deshalb nicht Teil dieses Vorhabens sind.

Um Software hinsichtlich ihrer Nachhaltigkeit in Bezug auf den ausgelösten Hardware- und Energiefluss zu beurteilen, sind operationalisierbare Kriterien notwendig. Diese Kriterien können dann z. B. zur Information der Verantwortlichen für Softwareentwicklung, Softwarebeschaffung oder zur Vergabe eines Umweltkennzeichens eingesetzt werden.

Der hier entwickelte Kriterienkatalog konzentriert sich auf Umweltwirkungen, die aus dem *Betrieb* eines Softwareprodukts resultieren. Dies schließt nicht aus, dass für die Vergabe eines Umweltkennzeichens weitere Kriterien zur Anwendung kommen, die sich auf den Prozess der Softwareentwicklung (z. B. Einhaltung von ILO[4]-Standards beim Outsourcing von Programmierarbeiten), auf die Funktionalität der Software (z. B. Barrierefreiheit oder Ausschluss bestimmter Kategorien wie „Ballerspiele“) oder weitere Aspekte beziehen. Es erscheint uns jedoch wichtig, die hier behandelten *Auswirkungen von Softwareeigenschaften auf die Beanspruchung natürlicher Ressourcen* als klar definierten Forschungsgegenstand zu behandeln und nicht schon im Ansatz mit anderen Fragestellungen zu vermischen.

Die Zusammenhänge zwischen Software-Eigenschaften, Nutzerverhalten, Hardware-Inanspruchnahme und Ressourcenaufwand werden durch ein Wirkungsmodell beschrieben (siehe Abbildung 6). Das Wirkungsmodell beschreibt die Zusammenhänge rein indikativ und macht keine quantitative oder qualitative Aussage über diese Wirkungsprinzipien. Das Modell dient im Rahmen dieses Vorhabens als Grundlage und Orientierungsrahmen für die Diskussion der Auswirkungen unterschiedlicher Softwareprodukte.

Anhand des Wirkungsmodells kann gezeigt werden, dass sich Software-Eigenschaften auf unterschiedlichen Ebenen und über verschiedene Wirkungspfade auf materielle Flüsse auswirken. Als direkte Wirkungen kann man die Hardware-Inanspruchnahme und den mit der Ausführung der Software verbundenen Energieverbrauch bezeichnen. Indirekt wirkt die Software über das Verhalten der Nutzer und der nutzenden Organisationen. So tragen beispielsweise die Standardeinstellungen (beispielsweise von Energiesparmodi oder Grafikkartenbeschleunigung) zu den typischen Nutzungsszenarien bei. Durch die Weiterentwicklung von Software (Updates und Versionssprünge) trägt diese ebenfalls entweder direkt zur veränderten Hardware-Inanspruchnahme oder im ungünstigsten Fall zur Obsoleszenz der eingesetzten Hardware bei, indem neue Software-Generationen auf älteren Hardware-Plattformen nicht mehr lauffähig sind. Dies hat direkte Ressourcenflüsse bei der Entsorgung alter Hardware und für die Produktion der sie ersetzenden neuen Hardware zur Folge.

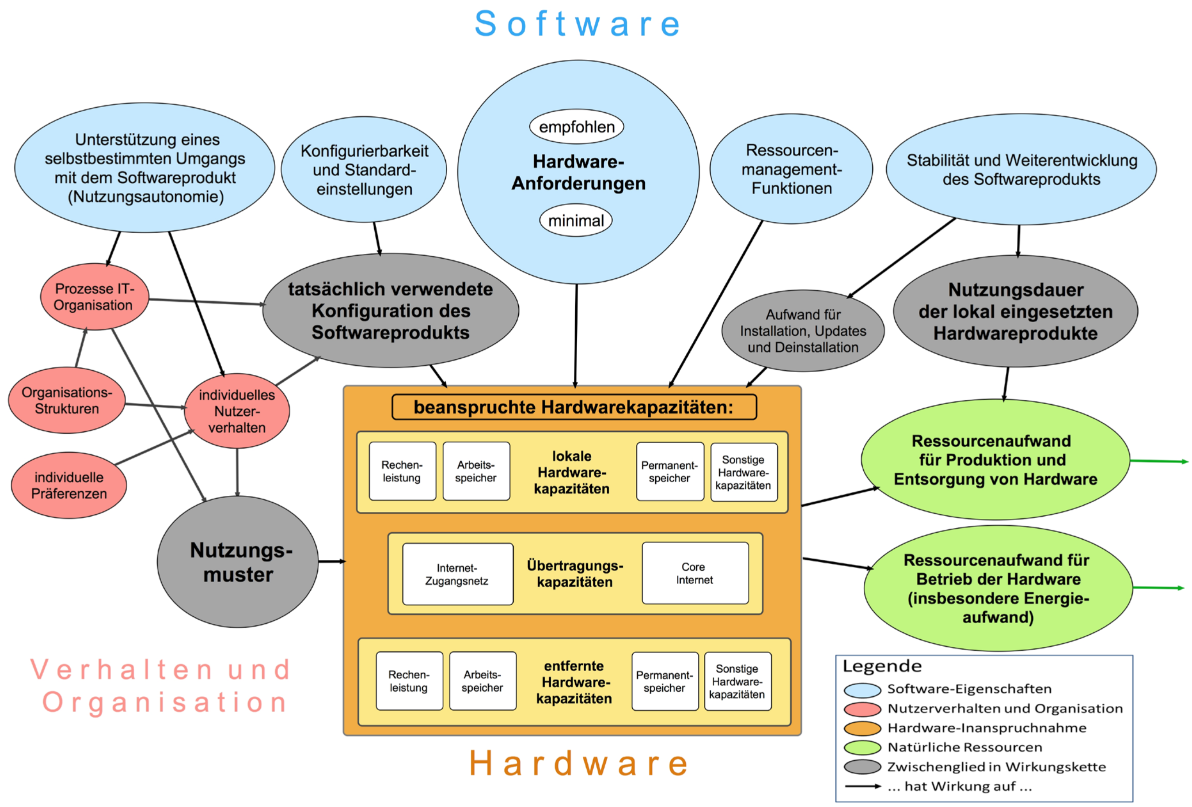
Das Wirkungsmodell trägt zur Strukturierung des Kriterienkatalogs zur Bewertung von Software bei, indem es die unterschiedlichen Wirkungsebenen aufzeigt. Diese finden sich in der Gliederung des Kri-

3 In einigen Fällen kann eine Erweiterung dieser Perspektive notwendig sein, indem analog zum Energiefluss auch Flüsse von Verbrauchsmaterialien wie Papier oder Toner durch die Hardware relevant sind. Ob ein solcher Fall vorliegt, muss abhängig von der untersuchten Software entschieden werden.

4 International Labour Organization (ILO)

terienkatalogs in „Ressourceneffizienz“, „Potenzielle Hardware-Nutzungsdauer“ und „Nutzungsautonomie“ wieder.

Abbildung 6: Wirkungsmodell für Zusammenhänge zwischen Software-Eigenschaften, Nutzerverhalten/Organisation, Hardware-Inanspruchnahme und Ressourcenaufwand



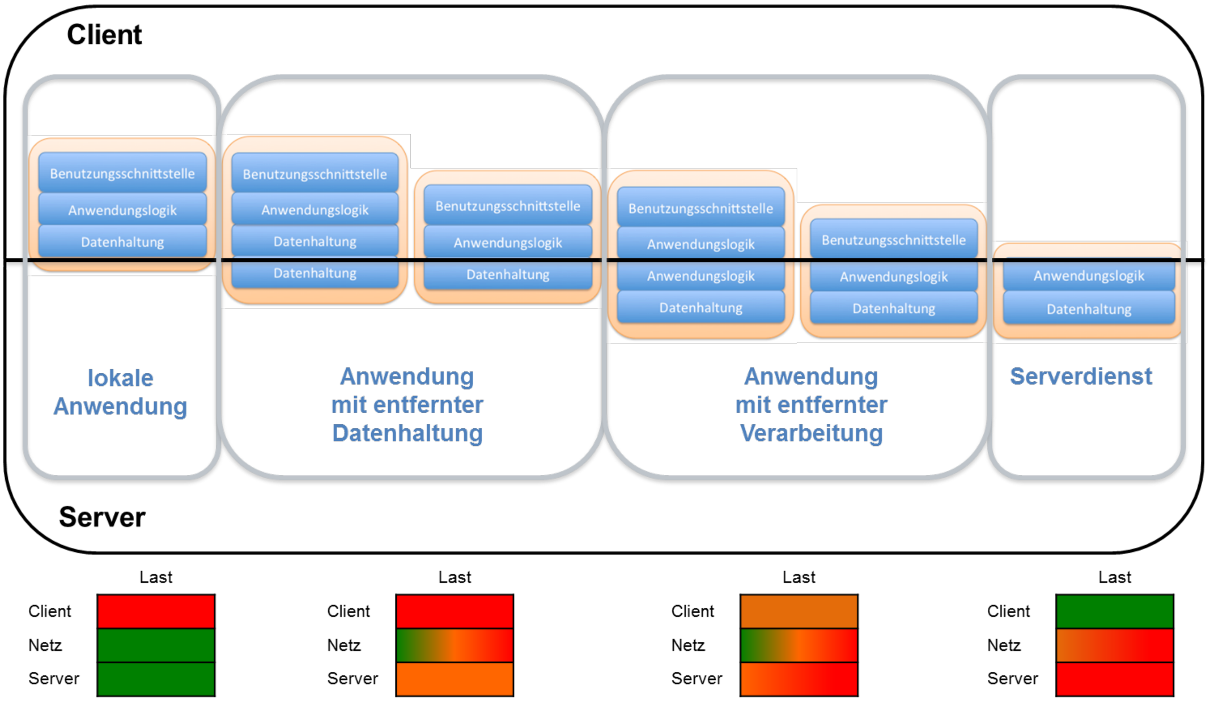
Quelle: Eigene Darstellung, Universität Zürich

3.3 Klassifikation von Softwareprodukten

Die Auswirkung von Software unterscheidet sich je nach Systemumgebung und Anwendungsbereich der Software. Es wurde im Rahmen des Forschungsvorhabens daher eine Klassifikation von Softwareprodukten vorgenommen, die es ermöglicht, die Auswirkungen zuzuordnen und damit auch die Bewertungskriterien je nach Softwareklasse unterschiedlich zu interpretieren.
Es wurde eine auf die Anwendung der Kriterien abgestimmte Klassifikation von Softwareprodukten entwickelt, die in Abbildung 7 schematisch dargestellt ist. Die Klassen unterscheiden sich durch die Arbeitsteilung zwischen Client und Server. Es wurden folgende Softwareklassen festgelegt:

- Lokale Anwendung
- Anwendung mit entfernter Datenhaltung
- Anwendung mit entfernter Verarbeitung
- Serverdienst

Abbildung 7: Klassifikation von Anwendungssoftware bezüglich der Softwarearchitektur



Quelle: Eigene Darstellung nach Lassmann (2006), S. 150

Bei der **lokalen Anwendung** liegt die Benutzungsschnittstelle (Eingabe- und Ausgabegeräte), die Anwendungslogik (Datenverarbeitung) und die Datenhaltung (Speicherung) direkt auf dem lokalen Endgerät des Anwenders, ohne dass Daten über Netzwerke übertragen werden. Typische lokale Anwendungen sind beispielsweise Text- oder Bildverarbeitung. Der Ressourcen- und Energieverbrauch sind entsprechend lokal verortet.

Bei **Anwendungen mit entfernter Datenhaltung** liegen die Daten in einem externen Rechenzentrum oder Serverraum und müssen vor der lokalen Verarbeitung zunächst über ein Netzwerk übertragen werden (z. B. Online-Storage). Hier liegt der überwiegende Teil des Ressourcen- und Energieverbrauchs nach wie vor beim Anwender (Client), ein Teil der Lasten wird jedoch ins Netzwerk und in Richtung Rechenzentrum (Server) verlagert.

Eine noch weitergehende Auslagerung findet bei **Anwendungen mit entfernter Verarbeitung** statt. Ein typisches Beispiel hierfür sind Thin-Client-Anwendungen, die lokal nur eine Benutzerschnittstelle zur Verfügung stellen, bei denen die eigentlichen Rechenoperationen bzw. die Datenverarbeitung aber auf einem Server stattfinden und die Datenhaltung ebenfalls serverseitig erfolgt. Der lokale Client kann dadurch ressourcenschonend arbeiten, während netz- und serverseitig höhere Lasten auftreten. Viele Datenbank- und Internetanwendungen zählen zu dieser Softwareklasse.

Reine **Serverdienste** stellen keine explizite Benutzungsschnittstelle zur Verfügung (allenfalls eine Wartungsschnittstelle für den Administrator) sondern laufen ausschließlich auf dem Server und beanspruchen dessen Ressourcen. Solche Serverdienste können beispielsweise per Zeitplan gestartete Anwendungen (Cron-Jobs) zur Datensicherung, Virenprüfung oder Systemaktualisierung sein. Abhängig von der Art des Serverdienstes findet hier eine Datenübertragung über interne Netzwerke (innerhalb des Rechenzentrums oder RZ-Verbundes) oder über externe Netzwerke statt.

Je nach Softwareklasse hat der Ressourcen- und Energiebedarf einen unterschiedlichen Schwerpunkt, besonders hinsichtlich der Aufteilung zwischen dem lokalen und dem entfernten (serverseitigen) Bedarf. Beim Design von Prüfaufbauten, beispielsweise zum Energieverbrauch der Hardware, muss die

Softwareklasse daher mitberücksichtigt werden. Bei der in Abschnitt 4 beschriebenen exemplarischen Untersuchung von Anwendungssoftware werden gezielt unterschiedliche Softwareklassen ausgewählt, um die Anwendbarkeit des Bewertungsmodells bei unterschiedlichen Wirkungsschwerpunkten zu überprüfen (siehe Abschnitt 4.2).

3.4 Bestandsaufnahme

Zur Bestandsaufnahme von bereits bestehenden Methoden und Kriterien zur Bewertung der Nachhaltigkeitseigenschaften von Software wurden insgesamt 157 Quellen vorselektiert und 61 thematisch relevante Veröffentlichungen (wissenschaftliche Veröffentlichungen und Vergabegrundlagen) weitergehend ausgewertet. Eine Übersicht der ausgewerteten Veröffentlichungen befindet sich im Abschnitt 8 Quellenverzeichnis dieses Berichtes. Aus der Literatur wurden insgesamt 96 Kriterien extrahiert und einem Konsolidierungs- und Strukturierungsprozess unterzogen. Es wurde untersucht, ob für die jeweiligen Kriterien Indikatoren für die Kriterienerfüllung existieren bzw. ob diese messbar oder überprüfbar sind. Sofern Metriken zur Überprüfung der Kriterien vorlagen, wurden diese dokumentiert.

3.5 Bewertung und Ergänzung der bestehenden Methoden und Modelle

Die Bewertung und Ergänzung der bestehenden Methoden und Modelle zur Bewertung von Software erfolgten in einem iterativen Prozess innerhalb des Projektteams. Anhand der Wirkung von Software auf unterschiedliche Nachhaltigkeitsdimensionen wurde eine Systematik entwickelt, innerhalb derer die Kriterien zugeordnet werden konnten (siehe Tabelle 4).

Tabelle 4: Systematik der Bewertungskriterien

1 Ressourceneffizienz	2 Potenzielle Hardware-Nutzungsdauer	3 Nutzungsautonomie
1.1 Hardwareeffizienz 1.2 Energieeffizienz 1.3 Ressourcenmanagement	2.1 Abwärtskompatibilität 2.2 Plattformunabhängigkeit und Portabilität 2.3 Hardwaresuffizienz	3.1 Transparenz und Interoperabilität 3.2 Deinstallierbarkeit 3.3 Wartungsfunktionen 3.4 Unabhängigkeit von Fremdressourcen 3.5 Qualität der Produktinformation

Quelle: Kriterienkatalog für nachhaltige Software, siehe Anhang 1

Die Systematik umfasst nur solche Kriterien, die die direkten Wirkungen der Software (Effekte erster Ordnung) bewerten. Nicht enthalten sind Kriterien, die auf die indirekten Wirkungen abzielen („Green by IT“). Solche indirekten Wirkungen (Effekte zweiter Ordnung) sind beispielsweise Energieeinsparungen, die durch die Anwendung der Software ermöglicht werden. Diese wurden bei der Festlegung des Untersuchungsrahmens (siehe Abschnitt 3.1 Präzisierung der Zielsetzung) bewusst ausgeschlossen.

Anhand der Systematik der Bewertungskriterien wurden Lücken identifiziert, für die in der Literatur noch keine Kriterien definiert wurden. Es wurden im Rahmen des Forschungsvorhabens neue Kriterien entwickelt um diese Lücken zu schließen.

3.6 Integration und Komplexitätsreduktion

Bei der Zuordnung bestehender Bewertungsmethoden für Nachhaltigkeitseigenschaften von Software zur Systematik der Bewertungskriterien zeigen sich naturgemäß Überschneidungen und Mehrfachnennungen ähnlicher Kriterien durch unterschiedliche Literaturquellen. So wird beispielsweise der Aufwand zur Auslieferung der Software mehrfach mit unterschiedlichen Schwerpunkten beschrieben (u.a. Verpackungsmaterialien, Transportemissionen, Möglichkeit zur Online-Auslieferung). In einem anschließenden Bearbeitungsschritt wurden daher die unterschiedlichen Kriterien zusammengefasst und die Komplexität der Ansätze auf ein einheitliches Niveau reduziert.

3.7 Entwicklung eines Kriterienkataloges für nachhaltige Software

Um einen anwendbaren Kriterienkatalog zu entwickeln, wurden die Kriterien mit Indikatoren versehen. Als Indikatoren werden hierbei empirisch bestimmbare Größen bezeichnet, die Aufschluss über einen Sachverhalt geben. Einige Indikatoren sind dabei messbar, wie beispielsweise die prozentuale Auslastung der CPU oder der benötigte Speicherplatz, einige Indikatoren können in Form von Prüfpunkten abgefragt werden, die entweder zutreffen oder nicht („Vorhandensein von ...") und eine weitere Form von Indikatoren arbeitet mit einem Benotungssystem (z. B. „sehr gut", „gut", „genügend", „ungenügend"), um auf qualitative Aspekte eingehen zu können.

Der Kriterienkatalog für nachhaltige Software, der im Rahmen dieses Forschungsvorhabens entwickelt wurde, ist in Anhang 1 dokumentiert. Bei dem Kriterienkatalog handelt es sich um den verabschiedeten und durch den Auftraggeber abgenommenen Sachstand (siehe Anhang 1, Seite 92). Diese sowie zukünftige Versionen werden nach derzeitigem Stand auf einer von der Hochschule Trier, Standort Birkenfeld betriebenen Webseite[5] abgelegt.

Die Kriterien innerhalb des Kriterienkatalogs sind wie folgt strukturiert:

- **Nummer** des Kriteriums, je nach Gliederungsebene ein-bis dreistellig,
- **Bezeichnung** des Kriteriums (Überschrift),
- **Frage**, die durch die Prüfung des Softwareprodukts beantwortet wird,
- evtl. ein auf die Frage folgender **Kommentar**,
- das jeweils unterste Kriterium in der Hierarchie wird durch **Indikatoren** operationalisiert, die pro Kriterium mit Kleinbuchstaben durchgezählt werden.

Als Beispiel für die Struktur der einzelnen Kriterien innerhalb des Kriterienkataloges wird in nachfolgender Abbildung 8 das Kriterium der „Online-Auslieferung" genannt:

[5] http://green-software-engineering.de/kriterienkatalog

Abbildung 8: Beispiel für die Struktur einzelner Kriterien im Kriterienkatalog

1.1.6 Online-Auslieferung

Kann das Softwareprodukt (inkl. aller Programme, Daten und Dokumentation einschließlich Handbüchern) ohne den Transport physischer Datenträger (inkl. Papier) oder anderer materieller Güter (inkl. Verpackung) erworben, installiert und betrieben werden?

Indikatoren:

a) Sind eine Online-Auslieferung und ein Online-Update der Software möglich?

b) Wird unterstützt, dass das Softwareprodukt und seine Updates in der anwendenden Organisation zentral abgelegt werden?

Quelle: Kriterienkatalog für nachhaltige Software; siehe Anhang 1

4 Anwendung der Bewertungsmethodik anhand von Fallbeispielen

Ziel des Arbeitspaketes 2 im Rahmen des Vorhabens ist die systematische Überprüfung der in Arbeitspaket 1 gewonnenen Kriterien und Indikatoren anhand von repräsentativen Software-Architekturen und -Produkten. Das Arbeitspaket gliedert sich in folgende Unteraufgaben:

- Begründete Auswahl von drei Software-Architekturen
- Auswahl von jeweils drei Softwareprodukten pro Software-Architektur
- Abstimmung bzgl. der Auswahl mit dem Auftraggeber und ggf. Anpassung der Auswahl
- Systematische Überprüfung der in Arbeitspaket 1 erarbeiteten und im Workshop (AP 1.2) bewerteten Kriterien und Indikatoren hinsichtlich der Eignung und der praktischen Umsetzbarkeit

4.1 Beschreibung des Vorgehens

Im Arbeitspaket 2 wurden die Auswahl von Software und die Überprüfung der Anwendbarkeit anhand der in Arbeitspaket 1 entwickelten Kriterien und Indikatoren („Kriterienkatalog für nachhaltige Software“, siehe Anhang 1), vorgenommen. Insbesondere für den Bereich der Messung des Energieverbrauchs und der Beanspruchung der Hardwarekapazitäten (Kriterien 1.1.3, 1.1.4 und 1.2) wurden dabei im Labor des Instituts für Softwaresysteme entsprechende Messungen und Bewertungen anhand von geeigneten Nutzungsszenarien und Referenzsystemen vorgenommen (vgl. für die Methodik: Dick et al. 2011, Johann et al. 2013). Abhängig vom jeweiligen konkreten Kriterium wurden auch Augenscheinprüfungen, Dokumentenchecks, Blackbox-Tests etc. durchgeführt. Das für die Fallbeispiele verwendete Vorgehen bei der Aufnahme der einzelnen Indikatoren wird im Sinne einer Operationalisierung in Anhang 3 beschrieben. Die konkreten Anwendungsergebnisse für die untersuchten Softwareprodukte sind in Anhang 4 zu finden.

4.1.1 Standardnutzungsszenarien

Wesentliche Grundlage für viele Indikatoren ist ein Standardnutzungsszenario, welches für jede zu untersuchende Softwareproduktgruppe zunächst festgelegt werden muss. Dieses simuliert die realitätsnahe Nutzung der Software. Das Szenario beinhaltet die Abarbeitung von Aufgaben, für die die Software entwickelt wurde und ggf. die Interaktion mit dem Anwender der Software. Durch die Verwendung von einheitlichen Standardnutzungsszenarien ist es bspw. möglich, den Energieverbrauch und die Hardware-Inanspruchnahme verschiedener Softwareprodukte der gleichen Produktgruppe miteinander zu vergleichen.

Im Rahmen dieses Forschungsvorhaben wurden die Standardnutzungsszenarien für die untersuchten Produktgruppen im Projektteam festgelegt (siehe beispielhaft Anhang 2). Diese Festlegung beinhaltet methodenbedingt eigene Annahmen und Eingrenzungen.

Sofern die Methodik zur Bewertung von Software über den Rahmen dieses Forschungsprojektes hinaus genutzt werden soll, wird daher empfohlen, die Entwicklung von Standardnutzungsszenarien einem transparenten Verfahren zu unterziehen, das interessierte Kreise (beispielsweise Software-Entwickler, -Hersteller und -Anwender) einbezieht. Dieses Vorgehen entspricht der Entwicklung von Produktkategorieregeln (product category rules – PCR), wie sie bei sogenannten „Typ III Umweltdeklarationen“ verwendet werden (DIN EN ISO 14025)[6]. Produktkategorieregeln beschreiben in der Norm eine standardisierte Nutzung von Produkten und bieten die Grundlage zur Durchführung von produktbezogenen Ökobilanzen.

Zur Sicherstellung der Vergleichbarkeit von Softwareprodukten einer Produktgruppe wird mit allen Produkten der Gruppe dasselbe Standardnutzungsszenario ausgeführt. Folgende Überlegungen kommen bei der Erstellung von Standardnutzungsszenarien zum Tragen:

- Standardnutzungsszenarios werden mittels einer Automatisierungssoftware wie z. B. WinAutomation (siehe auch Abschnitt 4.1.2.1) erstellt und enthalten Standardaufgaben, die mit der Software bearbeitet werden. Dabei wird darauf geachtet, dass sich vor jeder Aktion eine Wartezeit befindet, welche den Nutzer simuliert. Des Weiteren wird darauf geachtet, dass möglichst viele verschiedene Standardaktionen ausgeführt werden, die bei einem Nutzer in einem typischen Arbeitsablauf vorkommen.
- Ein wichtiger Faktor bei der Erstellung eines Standardnutzungsszenarios ist die Reproduzierbarkeit der Testabläufe für alle zu vergleichenden Softwareprodukte. Dazu muss im Vorfeld der Erstellung des Szenarios geprüft werden, dass alle Aktionen, die in dem Szenario enthalten sein sollen, auch mit allen zu vergleichenden Softwareprodukten möglichst in Standardkonfiguration, also ohne zusätzliche Pakete oder Plugins durchgeführt werden können.
- Es muss darauf geachtet werden, dass bei der wiederholten Durchführung (in den hier beschriebenen Fallbeispielen werden 30 Testdurchläufe je Messszenario durchgeführt) alle in einem Durchlauf im Referenzsystem persistierten Änderungen vor Ablauf des darauffolgenden Durchlaufs wieder rückgängig gemacht werden müssen, um einen vergleichbaren Durchlauf zu gewährleisten. Darüber hinaus hat es sich als sinnvoll herausgestellt, alle zur Durchführung eines automatisierten Ablaufes notwendigen Voraussetzungen zu notieren (bspw. Virenscanner deaktivieren) und vor dem Start zu prüfen. Auch dies trägt zur reibungslosen automatisierten Durchführung der Testdurchläufe bei.
- Die Synchronisierung der Messungen erfolgt im Rahmen der Fallbeispiele über Zeitstempel. Sämtliche Messpunkte der Messung der Hardwarekapazitäten, wie auch der Messung der elektrischen Leistung, werden mit einem Zeitstempel versehen. Zusätzlich erzeugt der Lasttreiber (s. u.) eine Log-Datei, in der für jeden Messdurchlauf eine Start- und Endzeit sowie Zeitstempel für Beginn und Ende der durchgeführten Aktionen gespeichert wird. So ist es möglich, den einzelnen Aktionen genau die Inanspruchnahme von Hardwarekapazitäten und Energieverbrauch zuzuordnen (z. B. zur Identifikation hardwareintensiver Module für Kriterium 1.1.5, Indikator c)).

[6] DIN EN ISO 14025:2011-10, Umweltkennzeichnungen und -deklarationen - Typ III Umweltdeklarationen - Grundsätze und Verfahren (ISO 14025:2006)

- Zur Verarbeitung und späteren Reproduzierbarkeit werden die Messwerte, die Log-Datei und das Automatisierungsskript an zentraler Stelle gesichert.

Tabelle 5: Überblick der Standardnutzungsszenarien für die ausgewählten Softwareprodukte

Textverarbeitung	Browser
Gesamten Text bearbeiten	E-Mail lesen/schreiben
Inhaltsverzeichnis einfügen & aktualisieren	Web-Videostream anschauen
Ansicht anpassen	Online-Shop besuchen
Inhalte hinzufügen & bearbeiten	Lesezeichen setzen
PDF erzeugen	Add-on installieren
Speichern	Datei downloaden
Content Management Systeme	**Datenbanken**
Kommentare beantworten	Schema bereits vorhanden
Neue Seite erstellen	Daten eintragen
Alle Seiten veröffentlichen	Daten lesen
PDF-Dateien hochladen	Daten verändern
PDF-Dateien verlinken	Daten löschen
Seite betrachten	230 Durchgänge je Funktion, 120.000 Zugriffe je Durchgang
Zusätzlich: Lastgenerierung zur Simulation von Besuchern	

Tabelle 5 gibt einen Überblick über die durchgeführten Aktionen, je nach Softwareklasse. Ein ausführliches Beispiel für den Ablauf eines Standardnutzungsszenarios für Textverarbeitungssoftware findet sich in Anhang 2.

4.1.2 Vorgehensweise zur Messung von Hardware-Inanspruchnahme und Energieverbrauch

Zur Messung des Energieverbrauchs und der in Anspruch genommenen Hardwarekapazitäten gemäß den Kriterien 1.1.3, 1.1.4 sowie 1.2 werden die festgelegten Standardnutzungsszenarien in Aufgaben und zu bearbeitende Teilschritte (Aktionen) unterteilt. Anschließend werden die Aufgaben und Teilschritte mittels einer Automatisierungssoftware (Lasttreiber) zu einem Testablauf zusammengefasst. Um Seiteneffekte durch eine unsaubere Installation oder Altlasten zu vermeiden, wird im ersten Schritt auf dem System Under Test (SUT (Client), SUT (Server)) ein den Anforderungen angepasstes und standardisiertes Betriebssystem-Image jeweils neu aufgespielt. Dabei werden möglichst alle Prozesse deaktiviert, die eine Verfälschung der Messung hervorrufen können. Dies umfasst z. B. Virenscanner, automatische Update-Tools, Indizierungs- und Backupprozesse etc.

Für das System Under Test wird durch Messung zunächst die Grundauslastung (GA) bestimmt, d.h. die mittlere Auslastung der CPU, des Arbeitsspeichers, des Permanentspeichers sowie der über das Netzwerk übertragenen Datenmenge des Systems ohne die zu untersuchende Software. Diese Werte sind für das System Under Test spezifisch und müssen nur einmal ermittelt werden.

Anschließend wird das zu messende Softwareprodukt auf dem System installiert und gestartet. Solange sich die Software noch im Leerlaufzustand befindet, d.h. in einem Zustand nach dem Starten, aber noch ohne die Ausführung eines Nutzungsszenarios oder einer Interaktion mit dem Anwender, wird die Leerlauf-Auslastung (LA) gemessen.

Die dritte Messung dient der Bestimmung der (Brutto-) Auslastung des Systems (BA) während der aktiven Nutzung der Software durch ein Standardnutzungsszenario. Standardisierte Auswertungen sorgen dafür, dass Softwareprodukte, die die gleichen Nutzungsszenarien durchlaufen haben, hinsichtlich ihrer Energieeffizienz und ihrer Nutzung der Hardwarekapazitäten verglichen werden können. Das Szenario wird mit dem Lasttreiber mehrfach wiederholt, um eine möglichst repräsentative Stichprobe zu generieren. Während des Ablaufs des Szenarios werden die Inanspruchnahme der Hardwarekapazitäten sowie der Energieverbrauch gemessen und die aktiven Aufgaben im Aktivitätsprotokoll des Lasttreibers erfasst. Es werden die Systemauslastung der CPU, Haupt- und Festplattenspeicher sowie Netzwerklast und der Gesamtenergieverbrauch des Systems überwacht und aufgezeichnet.

Zur Messung des Energieverbrauchs stehen im Messlabor am Standort Umwelt-Campus Birkenfeld der Hochschule Trier verschiedene Rechner und ein Leistungsmessgerät mit Netzanalysator (Janitza UMG 604[7]) zur Verfügung. Zur Überprüfung der Indikatoren stehen ein Server- und ein Desktopsystem als SUT zur Verfügung (vgl. Tabelle 6). Auf der Softwareseite stehen Referenzinstallationen verschiedener Windows- und Linux-Betriebssysteme sowie Softwarepakete zur Automatisierung von Messungen und zur Messdatenerfassung bereit.

Tabelle 6: Ausstattung des Messlabors

	SUT (Server)	SUT (Client)
System	Asus RS100-E8-PI2	Desktop PC
Mainboard	P9D-M Series	Intel Desktop Board DG33BU
Prozessor	Intel Xenon E3-1220 v3 @ 3,1 GHz	Intel Core 2 Duo E6750 @ 2,66 GHz
Cache (L1/L2/L3)	256KiB/1MiB/8MiB	32KiB/4MiB/n. a.
RAM	8GiB	4GiB
Grafikkarte	On-board ASUS ASPEED Graphics	NVIDIA GeForce 8600 GT
HDD	1TB Seagate ST1000NM0033	320GB WD 3200YS-01PBG0
SSD	256GB Samsung SSD 840 Series	n. a.
Netzwerk	2x Intel I210 Gigabit Network Adapter	Intel 82566DC Gigabit Network Adapter
Netzteil	AcBel FSB009 250W, 80 plus bronze	Antec EarthWatts EA-430D 430 W, 80 plus (weiß)
Betriebssysteme	Ubuntu 14.04.5 LTS	Windows 7 Professional 64-Bit Service Pack 1, Ubuntu 16.04.1 LTS

Quelle: Hochschule Trier

Die Messung der Hardwarekapazitäten erfolgt betriebssystemabhängig direkt auf dem jeweiligen SUT. Es kommt Standardsoftware zum Überwachen der Systemleistung zum Einsatz, die z. T. direkt im Betriebssystem integriert ist oder quelloffen angeboten wird, bspw. Perfmon oder collectl. Folgende Leistungsparameter werden erfasst:

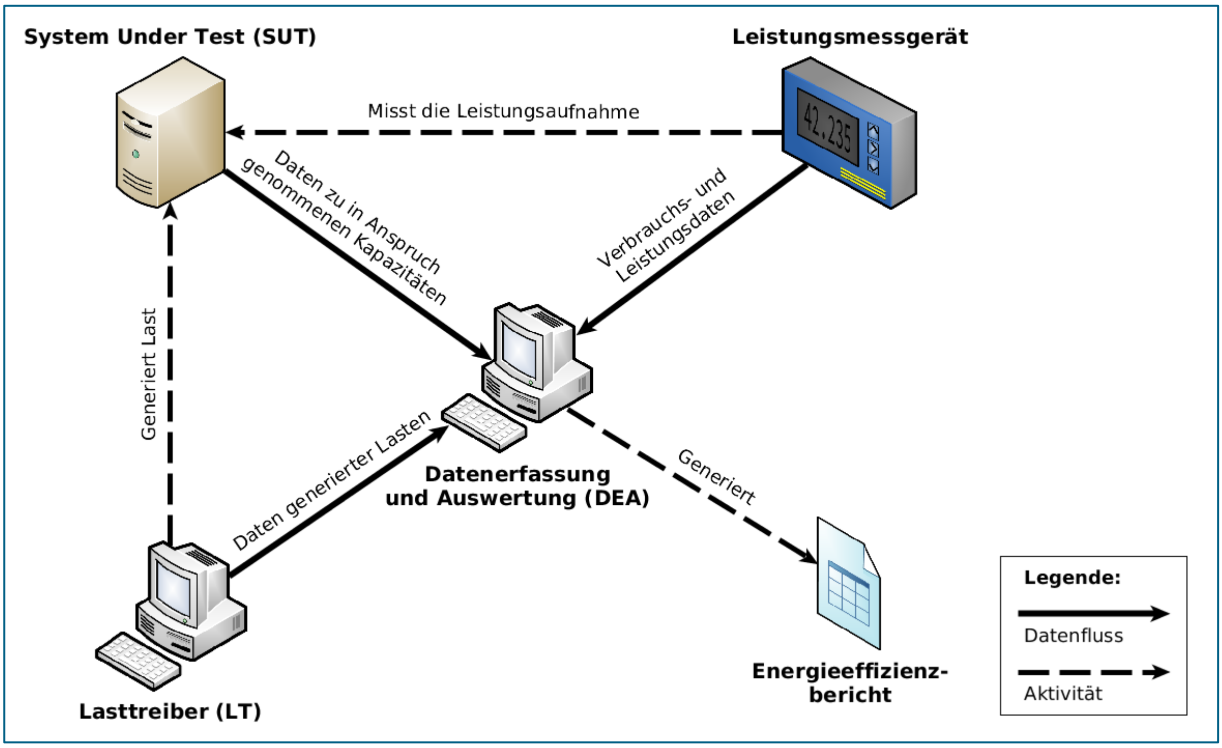
- Prozessorauslastung in Prozent
- Arbeitsspeicherbelegung in Prozent
- Physikalischer Speicher in gelesenen/geschriebenen Bytes
- Netzwerkauslastung in gesendete/empfangene Bytes

[7] Vgl. http://www.janitza.de/umg-604.html

4.1.2.1 Messaufbau

Abbildung 9 zeigt einen Messaufbau, mit dem der Energieverbrauch und die von einem Softwareprodukt in Anspruch genommenen Hardwarekapazitäten erfasst werden. Das zu messende Softwareprodukt wird in diesem Fall auf dem SUT installiert und von einem externen Lasttreiber (LT) belastet. Die Stromversorgung des SUT wird von dem Leistungsmessgerät überwacht. Das SUT sammelt die Messdaten der beanspruchten Hardwarekapazitäten (Leistungsdaten). Die von Leistungsmessgerät, SUT und LT ermittelten Daten werden zentral erfasst und ausgewertet (DEA).

Abbildung 9: Exemplarischer Messaufbau Energie- und Ressourceneffizienzmessung von Software



Quelle: Eigene Darstellung, Hochschule Trier

Es existieren zwei Variationen dieses Aufbaus:

- Wenn kein externer LT verwendet wird (z. B. bei rein lokalen Anwendungen wie Textverarbeitungssoftware), ist dieser in Form einer Automatisierungssoftware im SUT integriert. In diesem Fall trägt die Automatisierungssoftware - wenn auch in vernachlässigbarem Umfang – zur CPU- und RAM-Auslastung sowie zum Energieverbrauch des SUT bei. Dadurch, dass die Automatisierungssoftware bei der Messung der Grundauslastung mitgemessen wird, und diese bei der Analyse wieder vom Energieverbrauch, bzw. der Hardwareauslastung subtrahiert wird (s. Abschnitt 4.1.2.2.), ist die zusätzliche Belastung vernachlässigbar.
- Wenn das zu untersuchende Softwareprodukt auf einem verteilten System beruht (z. B. bei Anwendungen mit entfernter Verarbeitung wie Content Management Systemen) und ein Server mit einer geringen Anzahl an Nutzern simuliert werden soll, wird sowohl das SUT als auch der LT gemessen. In diesem Fall existieren dann zwei SUT (Client und Server). Die in diesem Fall maximale Anzahl von 1-2 Nutzern ist durch die Messkapazitäten des Leistungsmessgeräts eingeschränkt.

Abbildung 10: Labor für Umwelt- und Nachhaltigkeitsinformatik mit SUT (Client), SUT (Server), Datenerfassungsstation und Janitza-Leistungsmessgerät

Quelle: Eigene Fotografie, Hochschule Trier

Der Messaufbau umfasst für die Messungen von lokalen Anwendungen, bei der die komplette Anwendung auf dem Client ausgeführt wird, das System Under Test (SUT Client) sowie das Leistungsmessgerät. Die Aktionen des Standardnutzungsszenarios werden mittels der Automatisierungssoftware WinAutomation 4 Professional[8] generiert. Es handelt sich dabei um Standardsoftware zur Automatisierung von Nutzereingaben, die hier dazu verwendet wird, die Aktionen des Standardnutzungsszenarios automatisch auszuführen.

Bei Messungen von Anwendungen mit entfernter Datenhaltung und -verarbeitung sowie reinen Serverdiensten (wie z. B. Datenbanken) wird das zu untersuchende Softwareprodukt auf dem SUT (Server) installiert. Zum Generieren der (z. B. durch Nutzungseingaben) kommt neben der Automatisierungssoftware bei Softwareprodukten, die Webseiten ausliefern (z. B. Content Management Systeme) ein selbsterstelltes Java-Programm zum Einsatz, welches kontinuierlich in mehreren Tasks die Webseiten vom Server anfordert und so mehrere gleichzeitig mit dem Server interagierende Clients simuliert. Des Weiteren kommt ein Werkzeug zur Generierung von Transaktionen auf einer Datenbank zum Einsatz, welches ebenfalls mehrere gleichzeitig agierende Clients simuliert. Alle diese Tools werden durch den Lasttreiber gesteuert. Die Strommessung erfolgt auch hier über das oben genannte Leistungsmessgerät. Dabei wird bei verteilten Systemen der Energieverbrauch sowohl des SUT (Server und Clients) erfasst und separat ausgewertet, bzw. ggf. abgeschätzt, wenn eine Installation im Messlabor nicht möglich ist.

[8] Vgl. http://www.winautomation.com/

4.1.2.2 Messablauf

Um die Messungen nachvollziehbar und wiederholbar zu machen, wurde zur Überprüfung der Kriterien und Indikatoren (insbesondere 1.1.3, 1.1.4 und 1.2) ein standardisiertes Verfahren entwickelt, welches auf einschlägiger Literatur wie Dirlewanger 2006 und eigenen Veröffentlichungen beruht:

- *Messung der Grundauslastung*: Als Grundauslastung (GA) werden Messungen benannt, bei denen auf der Messhardware ausschließlich das Betriebssystem sowie ggf. genutzte Automatisierungssoftware ausgeführt wird. Die Messwerte der GA werden bei allen folgenden Messungen der Software genutzt, um die durch die Software zusätzlich erzeugte Inanspruchnahme von Hardwarekapazitäten und erzeugten Energieverbrauch berechnen zu können. Dazu werden 10 Messungen der Grundauslastung durchgeführt und der Durchschnitt der in Anspruch genommenen Kapazitäten und des Energieverbrauchs gebildet. Diese Werte werden dann von den Messergebnissen der zu vergleichenden Softwareprodukte subtrahiert:
  Der **Energieverbrauch** der GA wird als das arithmetische Mittel über alle Messungen (n) berechnet. Der Energieverbrauch jeder Einzelmessung wird als Summe der pro Sekunde gemittelten Leistungsaufnahme des SUT (P_i) über die Dauer der Ausführungszeit des Standardnutzungsszenarios (m Sekunden) berechnet (ergibt die Einheit Ws, Wattsekunde). Dividiert man das Ergebnis durch 3600 s/h, erhält man die Grundauslastung (E_{GA}) in Wattstunden (Wh). Die Formel zur Berechnung der Grundauslastung ist demnach
  $$E_{GA} = \frac{1}{3600 \cdot n} \sum_{k=1}^{n} \sum\nolimits_{i=1}^{m} P_i \,,$$
  wobei die Dauer der k-ten Messung m Sekunden beträgt und insgesamt n Messungen durchgeführt werden.
  Die in Anspruch genommenen Hardwarekapazitäten (**Prozessorlast und Arbeitsspeicher**) der GA werden als der Mittelwert aller mittleren aufgenommenen Kapazitätsmessungen in Prozent (%) der maximal verfügbaren Kapazitäten des SUT berechnet. Das heißt, bei insgesamt n Messungen werden die Grundauslastungen der Hardwarekapazitäten berechnet als
  $$GA = \frac{1}{n} \sum_{k=1}^{n} \bar{x}_k \,,$$
  wobei $\bar{x}_k$ das arithmetische Mittel der k-ten Messung in % ist.
  Datentransferraten für Permanentspeicherzugriffe und Netzwerkaktivität werden vom SUT in Megabytes pro Sekunde (MB/s) aufgezeichnet und analog als arithmetisches Mittel der Mittelwerte berechnet.
- Verwendung einer *Standardkonfiguration* (siehe auch Abschnitt 4.4.3): Vor jeder Messung muss sichergestellt werden, dass dieselben Voraussetzungen für die Messung der zu vergleichenden Softwareprodukte hergestellt werden. Zu diesem Zweck wird eine Standardkonfiguration aus Hardware und Betriebssystem (neue Installation mit Standard-Einstellungen) von einem vorher gesicherten Datenträger-Image verwendet. Von der Standardkonfiguration wurden die Referenzwerte der Grundauslastung gemessen. Hierauf wird dann die zu messende Software möglichst ebenfalls in einer Standardversion (z. B. ohne Zusatzpakete, Plugins o. Ä.) installiert. Die Imageverwaltung erfolgt mittels eines selbst entwickelten Tools. Dieses stellt sicher, dass jedes wieder auf die Festplatte des SUT aufgespielte Image byteweise gesichert wurde und somit den identischen Zustand vor der Installation wiederherstellt. Dadurch wird die Vergleichbarkeit der Messungen mehrerer Softwareprodukte ermöglicht, da jedes Softwareprodukt mit derselben Ausgangsbasis (also Standardkonfiguration) arbeitet.
- *Leerlaufmessung*: Bei einer Leerlaufmessung wird zusätzlich zum Betriebssystem auch die Software auf dem SUT ausgeführt, aber es werden keine Aktionen durch einen simulierten Nutzer ausgeführt. Die Aufnahme der Messwerte (Leerlauf-Auslastung (LA)) erfolgt analog zur

Messung der Grundauslastung. Die Berechnung der Effektiven Leerlauf-Auslastung (ELA) für die Indikatoren 1.1.3 a) bis g) wird in Anhang 1 in Tabelle 19 erläutert.

- *Messung des Standardnutzungsszenarios* (siehe auch Abschnitt 4.1.1): Bei der Messung der Standardnutzungsszenarien wird zunächst die Brutto-Auslastung (BA) des Szenarios als die mittlere Auslastung des Referenzsystems über die Ausführungsdauer, analog der Messung der Grundauslastung berechnet. Die Brutto-Auslastung beinhaltet sowohl die Grundauslastung des SUT als auch die zusätzlich durch die Ausführung der Software beanspruchte Auslastung. Außerdem wird die mittlere Ausführungsdauer (t) aller der im Rahmen des Standardnutzungsszenarios durchgeführten Messungen ermittelt. Mit diesen Angaben können anschließend die Indikatoren für die Hardware-Inanspruchnahme (H_i) (Prozessorarbeit, Arbeitsspeicherarbeit, Permanentspeicherarbeit und übertragene Datenmenge nach Kriterium 1.1.4) gemäß der Berechnungsvorschriften in Anhang 1 berechnet werden. Der Energieverbrauch (Bruttowert der elektrischen Arbeit zur Ausführung des Standardnutzungsszenarios gemäß Indikator 1.2.a, E_{BA}) wird ebenfalls analog zur Grundauslastung (E_{GA}) als Mittelwert der Einzelmessungen ermittelt. Der Nettowert des Energieverbrauchs zur Ausführung des Standardnutzungsszenarios ist dann die Differenz zwischen Brutto- und Grundauslastung ($E_{NA} = E_{BA} - E_{GA}$).
- In den Messungen der Fallbeispiele hat sich ein *Messzeitraum* von zehn Minuten für jede Einzelmessung als sinnvoll erwiesen. Dies gilt für das Standardnutzungsszenarien, die Baseline- und Leerlaufmessungen. Der Messzeitraum ist abhängig von den Softwareprodukten und ihrer Nutzung. Er ist so zu wählen, dass das Standardnutzungsszenario durchlaufen werden kann. Daher wird kein konkretes Zeitintervall vorgeschrieben.
- Sämtliche Messwerte werden als *Mittelwert über eine Sekunde* erfasst.
- Alle aufgezeichneten Messwerte, sowohl für Hardwarekapazitäten also auch für Energieverbrauch, werden zentral verwaltet. Jeder der sekündlich erfassten Messpunkte der Leistungsaufnahme und der Hardwareauslastung ist mit einem *Zeitstempel* versehen. Zusätzlich wird durch die Automatisierungssoftware eine Logdatei erstellt, die ebenfalls Zeitstempel für alle von ihr durchgeführten Testdurchläufe (jeweils Start- und Endzeitpunkt des Testdurchlaufs) und die in den Durchläufen enthaltenen Aktionen (jeweils Start- und Endzeitpunkt der Aktion) enthält. Dadurch wird sichergestellt, dass den Messwerten auch die auslösenden Aktionen zugeordnet werden können. Es kann dabei vorkommen, dass sich zwei oder mehr Aktionen in einem Zeitstempel wiederfinden. Dies ist bspw. dann der Fall, wenn eine vorangegangene Aktion noch nicht abgeschlossen ist und die nächste Aktion bereits durchgeführt wird.

4.1.3 Vorgehensweise zur Anwendung der weiteren Kriterien und Indikatoren

Unter Anwendung der weiteren Kriterien und Indikatoren wird die Überprüfung der Kriterien des Kriterienkatalogs verstanden, die nicht aus den oben beschriebenen unmittelbaren Messungen von Hardwarekapazitäten und Energieverbrauch entsprechend den Kriterien 1.1.3, 1.1.4 und 1.2 folgen. Um diese Kriterien auf ein konkretes Softwareprodukt anzuwenden, werden Handbücher, Webinformationen, Experten- und Nutzermeinungen eingeholt und weitere Tests vorgenommen.

Aufgrund der großen Anzahl der identifizierten Kriterien wird nachstehend nur eine Auswahl der Methoden zur Beobachtung vorgestellt. Es werden die Kriterien 1.3.3, 2.1 und 3.2.1 beispielhaft vorgestellt, da so aus jedem Hauptkriterium (1 Ressourceneffizienz, 2 Potenzielle Hardware-Nutzungsdauer und 3 Nutzungsautonomie) ein Unterkriterium betrachtet wird. Des Weiteren enthalten die drei gewählten Kriterien viele dedr auch in anderen Beobachtungen genutzten Vorgehensweisen. Generell gilt, dass alle Beobachtungen zur Evaluierung der Kriterien immer mit den Standardeinstellungen des Softwareprodukts durchgeführt wurden, um eine Vergleichbarkeit zu erzielen. Detaillierte Beschreibungen der Operationalisierung für alle Kriterien befinden sich in Anhang 3. Die Ergebnisse der in den Fallbeispielen aufgenommenen Beobachtungen befinden sich in Anhang 4.

Beispiel 1: Kriterium 1.3.3 Ressourcenschonende Standardeinstellungen

Wie im Kriterienkatalog (Anhang 1) beschrieben, zielt Kriterium 1.3.3 als Unterkriterium von 1.3 „Ressourcenmanagement“ darauf ab, die Frage zu beantworten, ob „die Standardeinstellungen des Softwareprodukts so gewählt sind, dass sie das Ziel der Ressourcenschonung mitberücksichtigen“. Dies wird so überprüft, indem der/die Prüfende die Standardeinstellungen der Software einer Augenscheinprüfung unterzieht und einschätzt, inwieweit die Standardeinstellungen des Softwareprodukts so gewählt sind, dass sie das Ziel der Ressourcenschonung mitberücksichtigen. In diesem Fall ist es wichtig, die Einstellungen während und nach der Installation zu betrachten und gegebenenfalls weitere auf das Produkt zugeschnittene Testszenarien zu erstellen, um die Standardeinstellungen mit anderen Softwareprodukten derselben Kategorie zu vergleichen.

Beispiele für diesbezüglich interessante Einstellungen sind voreingestellte Energieoptionen (vgl. Indikator 1.3.1.a)), die Dauer bis zum Aktivieren eines Energiesparmodus (vgl. Indikator1.3.2 a)), Einstellungen zur Datenkomprimierung und -übertragung etc. Als Ausgangspunkt für die Suche nach weiteren Einstellungen dienen auch die in Kriterium 1.1.5 identifizierten hardwareintensiven Module. In einigen Fällen kann es notwendig sein, weitere Methoden, wie z. B. das Durchsehen von Handbüchern oder Augenscheinprüfungen, heranzuziehen. Ein Beispiel hierfür wäre die voreingestellte Startseite eines Webbrowsers, die bei jedem Start des Softwareprodukts geöffnet wird.

Beispiel 2: Kriterium 2.1 Abwärtskompatibilität

Wie im Kriterienkatalog (Anhang 1) beschrieben, zielt Kriterium 2.1 als Unterkriterium von 2 „Potenzielle Hardware-Nutzungsdauer“ darauf ab, die Frage zu beantworten, ob „der Hersteller des Softwareprodukts garantiert, dass das aktuelle Release auf einem Referenzsystem von vor n Jahren betrieben werden kann“. Zur Beantwortung dieser Frage werden zwei Indikatoren erhoben.

Für Indikator a) wird eine Recherche in der Produktdokumentation, dem Benutzerhandbuch und der Hersteller- bzw. Produktwebseite sowie der Dokumentation und Hersteller- bzw. Produktwebseite von weiterer Software, die der Betrieb des Produkts bedingt, durchgeführt um festzustellen, vor wie vielen Jahren das jüngste der mindestens benötigten Softwareprodukte noch lauffähig war. Falls der Hersteller Angaben zur Hardware macht, werden diese mit den vorhandenen Referenzsystemen verglichen und anschließend das Alter des ältesten vergleichbaren Referenzsystems als Wert des Indikators herangezogen. Dies setzt voraus, dass nicht nur ein aktuelles Referenzsystem definiert wurde, sondern zusätzlich historische Referenzsysteme, die den Stand der Technik in vorangehenden Jahren beschreiben (siehe Abschnitt 6.1 *Referenzsystem „Arbeitsplatzcomputer für Büro-Software“*).

Wenn Angaben zur Hardware nicht zur Verfügung stehen, wird das Veröffentlichungsdatum des jüngsten Betriebssystems verwendet, auf welchem sowohl das Softwareprodukt selbst, als auch die zusätzlich benötigten Softwareprodukte lauffähig sind. Beispiel: Benötigt ein Produkt als Mindestanforderung Windows 2000 (2000 erschienen) und Oracle 11g (2007 erschienen, aber lt. Hersteller lauffähig unter Windows 2000 => lauffähig auf System aus dem Jahr 2000), beträgt die Abwärtskompatibilität im Jahr 2017 17 Jahre.

Wenn seit der ersten Anwendung der Kriterien ausreichend Zeit verstrichen ist (z. B. nach der Veröffentlichung einer neuen Version des Softwareprodukts), kann Indikator b) ausgewertet werden. Hier wird dann geprüft, ob das Standardnutzungsszenario mit dem aktuellen Release des Softwareprodukts auf einer Konfiguration ausführbar ist, die vor einer noch festzulegenden Anzahl von Jahren Standardkonfiguration war.

Beispiel 3: Kriterium 3.2.1 Deinstallierbarkeit der Programme

Wie im Kriterienkatalog (Anhang 1) beschrieben, zielt Kriterium 3.2.1 als Unterkriterium von 3 „Nutzungsautonomie“ darauf ab, ob „Nutzende ausreichend darin unterstützt werden, das Softwareprodukt rückstandsfrei zu deinstallieren“. Zur Beantwortung dieser Frage schlagen wir einen Black-Box Test vor, der zeigt, ob der Zustand nach der Deinstallation des Softwareproduktes derselbe ist wie derjenige vor der Installation. Um dies zu erreichen wird zunächst eine Kopie des Datenträger-Image mit dem Betriebssystem vor der Installation des Softwareproduktes angelegt. Dann wird das Softwareprodukt installiert, das Standardnutzungsszenario ausgeführt und das Softwareprodukt nach der Anleitung im Nutzerhandbuch (falls vorhanden) deinstalliert. Anschließend wird ein weiteres Datenträger-Image erzeugt und mit der Kopie von vor der Installation verglichen. Auf diese Weise können alle Dateien und Dateiänderungen erfasst werden, die nach der Deinstallation auf der Festplatte verbleiben. Zusätzlich wird bspw. für das Betriebssystem Windows von Microsoft noch die Registry nach verbliebenen Einträgen durchsucht.

4.2 Auswahl der untersuchten Software

Basierend auf bisherigen Projekten des Instituts für Softwaresysteme, einer einschlägigen Literaturrecherche, statistischen Absicherungen hinsichtlich privater/gewerblicher Verbreitung, Marktanteilen, Nutzungszahlen etc. und der Befragung von Experten wurden in einem zweistufigen Verfahren drei Softwareproduktgruppen ausgewählt. In einer Grobauswahl wurden dem Umweltbundesamt zunächst sechs Produktgruppen und daraus ausgewählte Softwareprodukte vorgestellt. Wesentliches Ziel war hierbei, eine repräsentative Auswahl zu treffen, um die ermittelten Kriterien und Indikatoren auf einer entsprechenden Bandbreite prüfen zu können und so deren Aussagekraft zu überprüfen. Zur Auswahl wurden insbesondere folgende Aspekte berücksichtigt:

- Nutzerspezifische Aspekte:
  - möglichst hohe Installations- und Nutzerzahlen
  - möglichst hohe Nutzungsdauer
  - verschiedene Nutzergruppen

- Produktspezifische Aspekte:
  - verschiedene Produktgruppen
  - verschiedene Endgeräte
  - verschiedene Betriebssysteme
  - verschiedene Lizenzmodelle

Gleichzeitig wurde gewährleistet, dass aus der im Arbeitspaket 1 entwickelten Software-Klassifikation je ein Beispiel einer Softwareklasse untersucht wird. Hierauf basierend wurden in Abstimmung mit dem Auftraggeber für die drei Produktgruppen konkrete Softwareprodukte ausgewählt. Diese sind in Tabelle 7 aufgeführt.

Tabelle 7: Liste ausgewählter Architekturen und Softwareprodukte

#	Produktgruppe	Architektur	Plattform	Einsatzfeld	Produkte
1	Textverarbeitung	Lokale Anwendung	Desktop/ Mobile	Privat & Geschäftlich	Es wurden zwei Textverarbeitungsprogramme gewählt (TVP1 und TVP2). Bei TVP1 handelt es sich um ein proprietäres Produkt, TVP2 ist quelloffen.
2	Browser	Anwendung mit entfernter Verarbeitung	Desktop/ Mobile	Privat & Geschäftlich	Es wurden drei Internetbrowser (B1, B2 und B3) gewählt. B1 und B2 sind quelloffen, B3 ist proprietär.
3	Content Management System	Anwendung mit entfernter Verarbeitung	Desktop/ Server	Privat & Geschäftlich	Es wurden drei CMS (CMS1, CMS2 und CMS3) gewählt. Alle CMS sind quelloffen.
4	Datenbank	Serverdienst	Server	Geschäftlich	Es wurden drei Datenbanksysteme (DB1, DB2 und DB3) gewählt. DB1 und DB2 sind quelloffen, DB3 ist proprietär.

Quelle: Hochschule Trier

4.3 Exemplarische Anwendungsergebnisse der Kriterien und Indikatoren

Ziel des Arbeitspaketes 2 ist die Überprüfung der in Arbeitspaket 1 entwickelten Kriterien und Indikatoren sowie der dort beschriebenen Softwareklassifikation anhand konkreter Produktgruppen und Softwareprodukte. Dabei wurden nicht alle Indikatoren vollständig für alle Produkte erhoben, insbesondere sofern es sich um bloße Replikationen handelt, die keinen weiteren Erkenntnisgewinn hinsichtlich der Aussagekraft und der Operationalisierbarkeit der Kriterien bringen. Zudem wurden nicht alle potenziellen Kombinationen von Softwareprodukten mit geeigneten Betriebssystemen, möglichen Endgeräten und auch Gerätevarianten vorgenommen, sondern auch hier vor allem auf eine repräsentative Auswahl geachtet. Die ausführlichen Ergebnisse der Fallbeispiele befinden sich in Anhang 4. Die in Arbeitspaket 2 erfolgte Bewertung der Kriterien hinsichtlich ihrer Aussagekraft und Umsetzbarkeit wird in Abschnitt 4.5 weitergehend dargestellt.

4.3.1 Exemplarische Messergebnisse der Kriterien 1.1.3, 1.1.4 und 1.2

Zur Darstellung der Messergebnisse für die Kriterien 1.1.3, 1.1.4 und 1.2 werden Messbeispiele aus verschiedenen Produktgruppen verwendet. Die Darstellung erfolgt entlang der Reihenfolge der Kriterien.

Kriterium 1.1.3 Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration

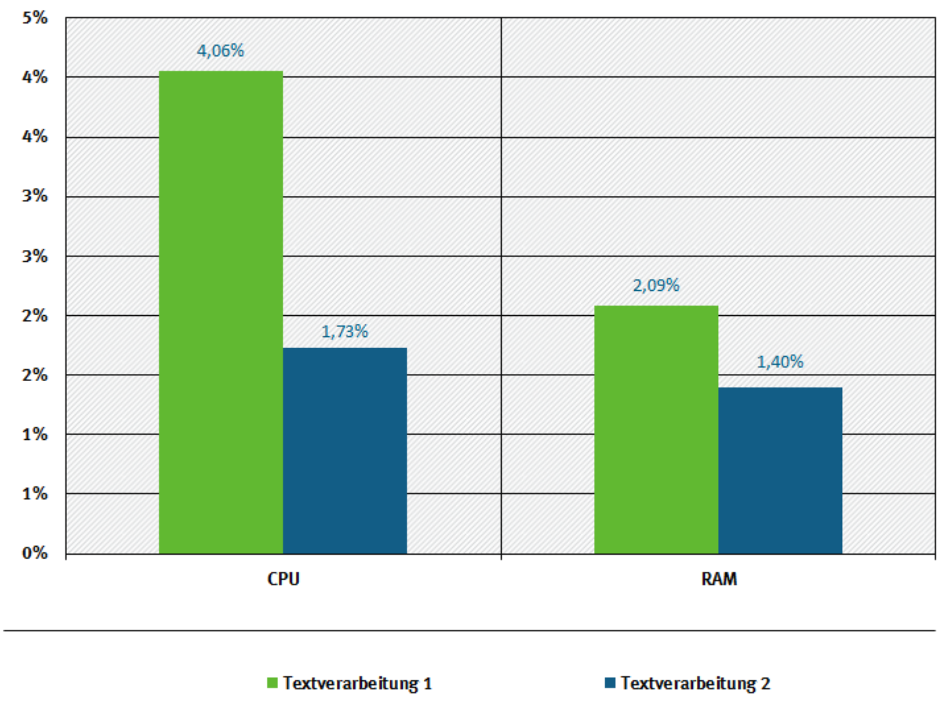
Die Werte für die Indikatoren des Kriteriums 1.1.3 werden wie in Abschnitt 4.1.2.2 und in den Berechnungsvorschriften im Kriterienkatalog (Anhang 1, Kriterium 1.1.3) beschrieben für jedes Softwareprodukt ermittelt und anschließend gegenübergestellt.

Bei der Hardware-Auslastung im Leerlauf handelt es sich um die zusätzliche Auslastung des Systems nach dem Starten der untersuchten Software zusätzliche zur Grundauslastung, die vor dem Starten der Software gemessen wurde. Die Hardware-Auslastung im Leerlauf wird bei lokalen Anwendungen für a) die Prozessorauslastung, b) die Arbeitsspeicherbelegung, c) die Permanentspeicherbelegung und d) die beanspruchte Bandbreite gemessen. Bei serverseitigen Anwendungen werden zusätzlich die Hardware-Auslastungen des Servers (CPU, Arbeitsspeicher und Permanentspeicher) bestimmt. Die

Werte der Hardware-Auslastung sind in Prozent angegeben. Sie geben den Anteil der jeweiligen Messgröße am Referenzsystem an.

Abbildung 11 zeigt die Hardware-Auslastung im Leerlauf zweier Textverarbeitungsprogramme im Vergleich für die Kriterien 1.1.3 a) Prozessorauslastung (CPU) und 1.1.3 b) Mittlere Arbeitsspeicherbelegung (RAM). Bei der CPU-Auslastung verursacht Textverarbeitungsprogramm 1 eine Mehrauslastung von 4 Prozent zusätzlich zur Grundauslastung. Beim Textverarbeitungsprogramm 2 sind es dagegen mit 1,7 Prozent CPU-Auslastung weniger als die Hälfte. Bei der mittleren Arbeitsspeicherauslastung (RAM) liegen die Werte der Mehrauslastung mit rund 2,1 Prozent für Textverarbeitung 1 und rund 1,4 Prozent für Textverarbeitung 2 nicht ganz so weit auseinander.

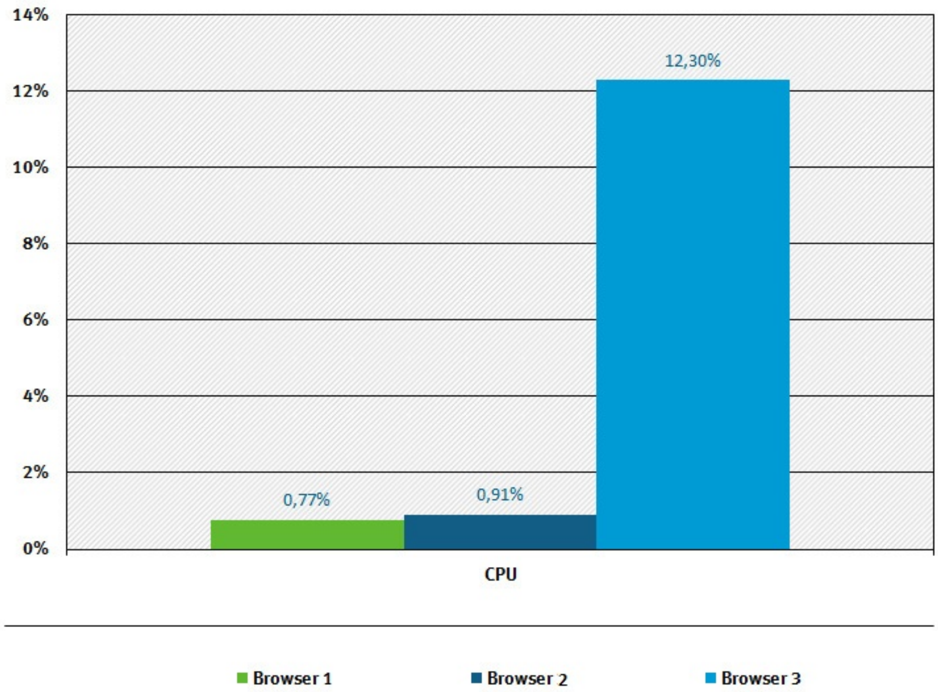
Abbildung 11: Hardware-Auslastung (CPU und RAM) im Leerlauf zweier Textverarbeitungsprogramme



Quelle: Eigene Darstellung, Hochschule Trier

Wendet man das Kriterium 1.1.3 a) Prozessorauslastung auf die drei untersuchten Internetbrowser an, so erhält man die in Abbildung 12 dargestellten Werte. Im Leerlauf lasten die Browser 1 und 2 die CPU um rund 1 Prozent mehr aus. Der Leerlauf von Browser 3 führt dagegen zu einer Mehrauslastung von 12 Prozent zusätzlich zur Grundauslastung.

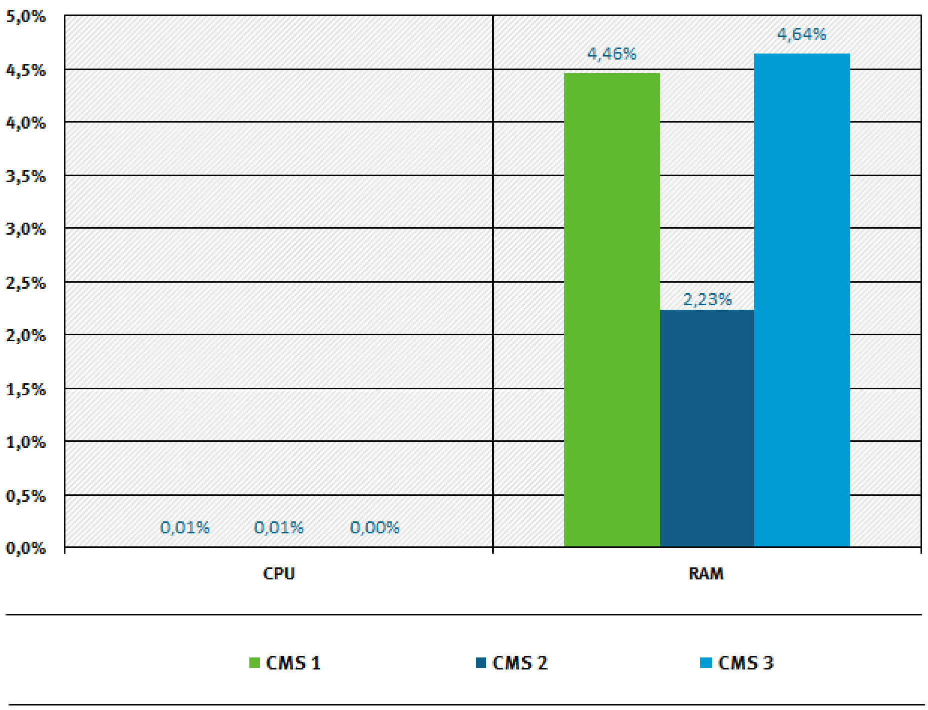
Abbildung 12: Hardware-Auslastung (CPU) im Leerlauf dreier Internetbrowser



Quelle: Eigene Darstellung, Hochschule Trier

Als serverseitige Software wurden drei Content Management Systeme (CMS) untersucht. Abbildung 13 zeigt die Kriterien 1.1.3 e) Serverseitige Prozessorauslastung (CPU) und 1.1.3 f) Serverseitige Arbeitsspeicherbelegung (RAM). Bei der zusätzlichen CPU-Auslastung liegen die Prozentzahlen aller CMS-Programme sehr nahe bei 0 Prozent, weshalb die Unterschiede auch auf Messungenauigkeiten zurückgehen können. Offensichtlich trägt der Leerlauf dieser Programme nicht merklich zu einer zusätzlichen CPU-Auslastung bei. Deutlicher sind die Unterschiede bei der RAM-Auslastung. Die CMS-Programme 1 und 3 bewirken eine zusätzliche RAM-Auslastung von rund 4,5 Prozent. Das CMS 2 im Leerlauf nutzt dagegen mit 2,2 Prozent Arbeitsspeicher etwa nur die Hälfte der übrigen CMS-Programme.

Abbildung 13: Hardware-Auslastung (CPU und RAM) im Leerlauf dreier Content Management Systeme (CMS) auf einem Server



Quelle: Eigene Darstellung, Hochschule Trier

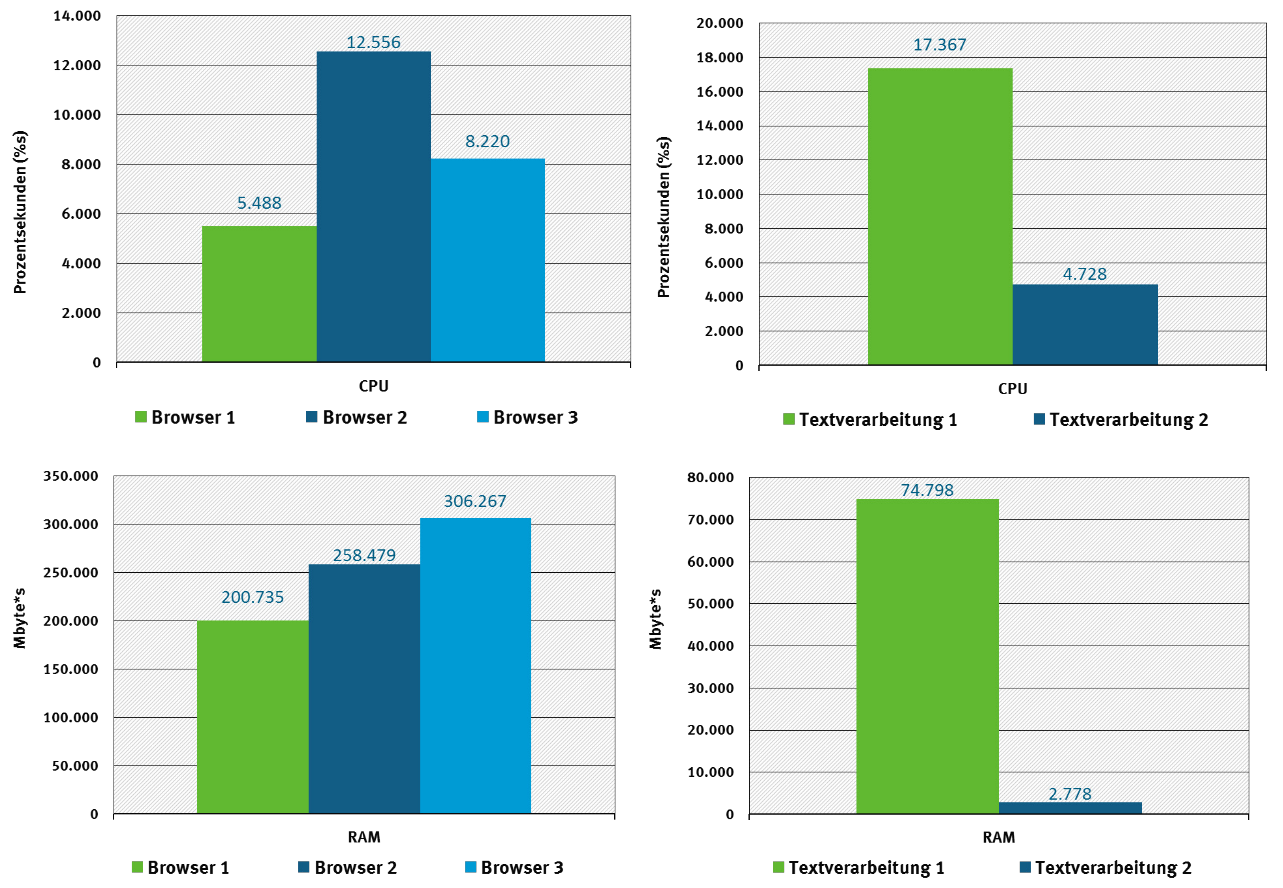
Bei den Textverarbeitungsprodukten konnte eine markante Leerlaufauslastung des Prozessors und des Arbeitsspeichers (Indikatoren a) und b)) beobachtet werden. Bei den untersuchten Browsern zeigte lediglich Browser 3 eine größere Auslastung des Prozessors (Indikator a)). Die Auslastung des Arbeitsspeichers (Indikator b)) ist vernachlässigbar. Die Messung der Content Management Systeme zeigte nur eine sehr geringe Leerlauf-Auslastung des Prozessors (Indikator e)), jedoch wurde der Arbeitsspeicher (Indikator f)) messbar ausgelastet. Bei allen Leerlaufmessungen wurde der Permanentspeicher (Indikatoren c) und g)) nur in vernachlässigbar geringem Ausmaß belegt. Für alle Produkte waren die Abweichung der beanspruchten Bandbreite (Indikator d)) zur Grundauslastung nicht messbar.

Kriterium 1.1.4 Hardware-Inanspruchnahme bei normaler Nutzung unter der Annahme einer Standardkonfiguration und eines Standardnutzungsszenarios

Auch die Werte für die Indikatoren des Kriteriums 1.1.4 werden, wie in Abschnitt 4.1.2.2 und in den Berechnungsvorschriften im Kriterienkatalog (Anhang 1, Abschnitt 1.1.4) beschrieben, für jedes Softwareprodukt ermittelt und anschließend gegenübergestellt. Anders als die Hardware-Auslastung im Kriterium 1.1.3, deren Einheiten für alle Indikatoren *Prozent* ist, unterscheiden sich die Einheiten der Hardware-Inanspruchnahme je nach Indikator. Der Wert stellt das Integral der Hardware-Auslastung über die Ausführungsdauer eines Standardnutzungsszenarios dar (bildlich gesprochen: die Fläche unter der zeitlichen Auslastungskurve vom Beginn der Programmausführung bis zu deren Ende). Dadurch ergeben sich Einheiten für die Prozessorarbeit (1.1.4 a) und e)) von *Prozentsekunden (%s)* und für die Arbeitsspeicherarbeit (1.1.4 b) und f)) von *Prozentsekunden (%s)* oder alternativ *Megabytesekunden (MByte*s)*, für die Permanentspeicherarbeit (Lesen und Schreiben) (1.1.4 c) und g)) von MByte/s*s = MByte und für die übertragene Datenmenge (1.1.4 d)) von *MBit/s*s = MBit*.

Abbildung 14 zeigt eine solche Gegenüberstellung für die Indikatoren a) (CPU) und b) (RAM) der Produktgruppen „Textverarbeitungsprogramme“ und „Browser“.

Abbildung 14: Vergleich der Inanspruchnahme von Prozessor und Arbeitsspeicher

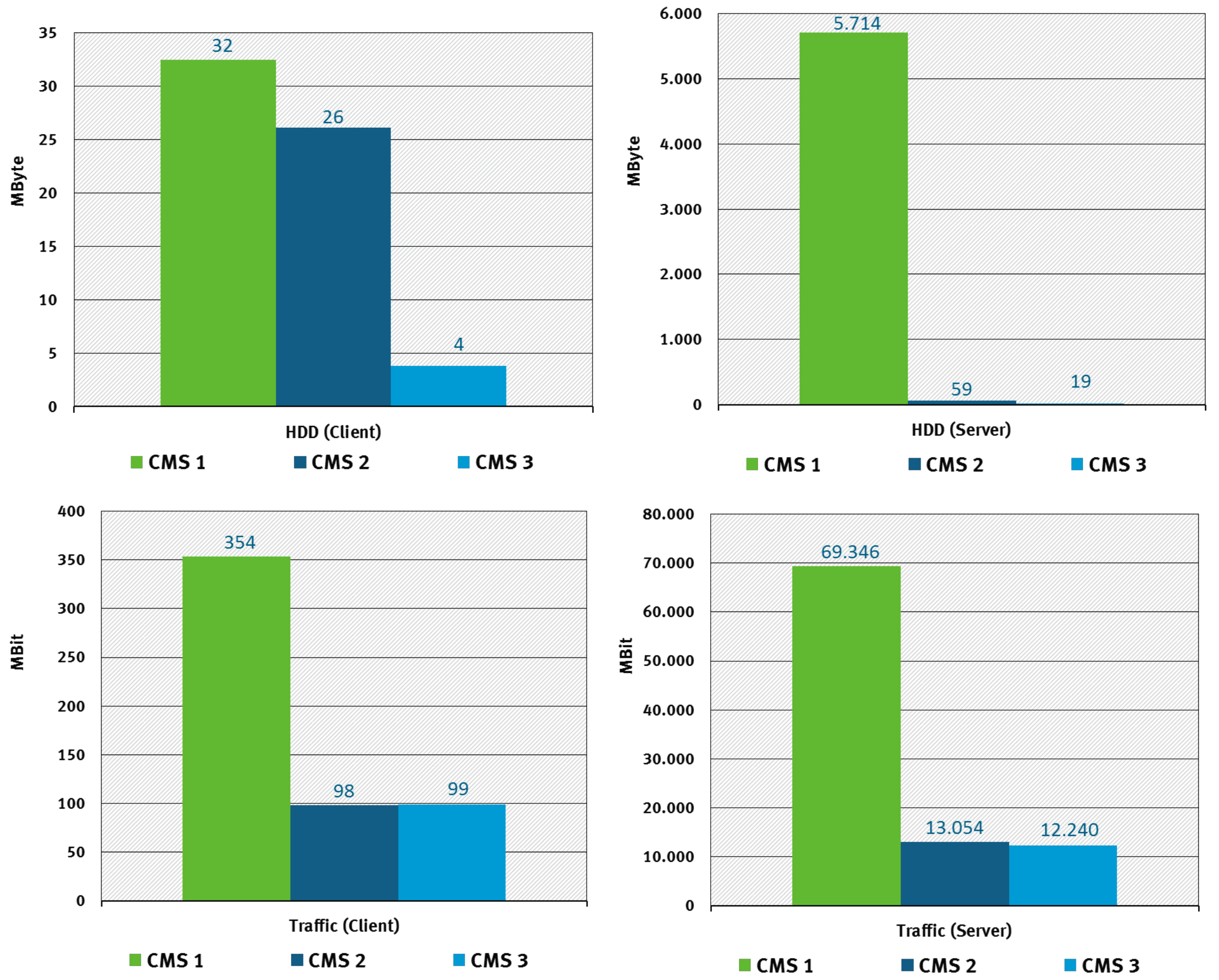


Quelle: Eigene Darstellung, Hochschule Trier

Durch die Einbeziehung von sowohl Ausführungsdauer (t) als auch eines Allokationsfaktors für die Grundauslastung (AF_i) spiegelt die Hardware-Inanspruchnahme (H_i) alle relevanten Faktoren wider, die die Auslastung der jeweiligen Hardwarekapazität beeinflusst. So kann auf einen Blick festgestellt werden, welches der Produkte die Kapazitäten am effizientesten nutzt.

Beim Betrieb der Softwareprodukte konnte auch eine messbare Auslastung der Bandbreite (Indikator d)) und des Permanentspeichers (Indikatoren c) und g)) festgestellt werden. Abbildung 15 zeigt den Vergleich für Produkte der Gruppe „Content Management Systeme“, sowohl auf Server-Seite, als auch auf Client-Seite. Auf der Server-Seite wurde das CMS und dessen zugehörige Datenbank gehostet und ausgeführt, auf der Client-Seite wurde das CMS mithilfe eines festgelegten Internetbrowser aufgerufen und bedient (vgl. Standardnutzungsszenario Tabelle 5).

Abbildung 15: Vergleich der Inanspruchnahme von Permanentspeicher (HDD) und übertragener Datenmenge (Traffic) der Content Management Systeme auf SUT (Client) und SUT (Server)



Quelle: Eigene Darstellung, Hochschule Trier

Die ermittelten Werte erlauben auch hier einen direkten Rückschluss auf die Effizienz der Produkte. Die starke Abweichung des CMS 1 bei Auslastung der Netzwerkbandbreite und Permanentspeicherbelegung kann dadurch erklärt werden, dass die Standardeinstellung von CMS 1 vorsieht, die auf der Webseite dargestellten Bilder nicht auf die Anzeigegröße zu verkleinern, sondern in voller Größe und Auflösung zu übertragen.

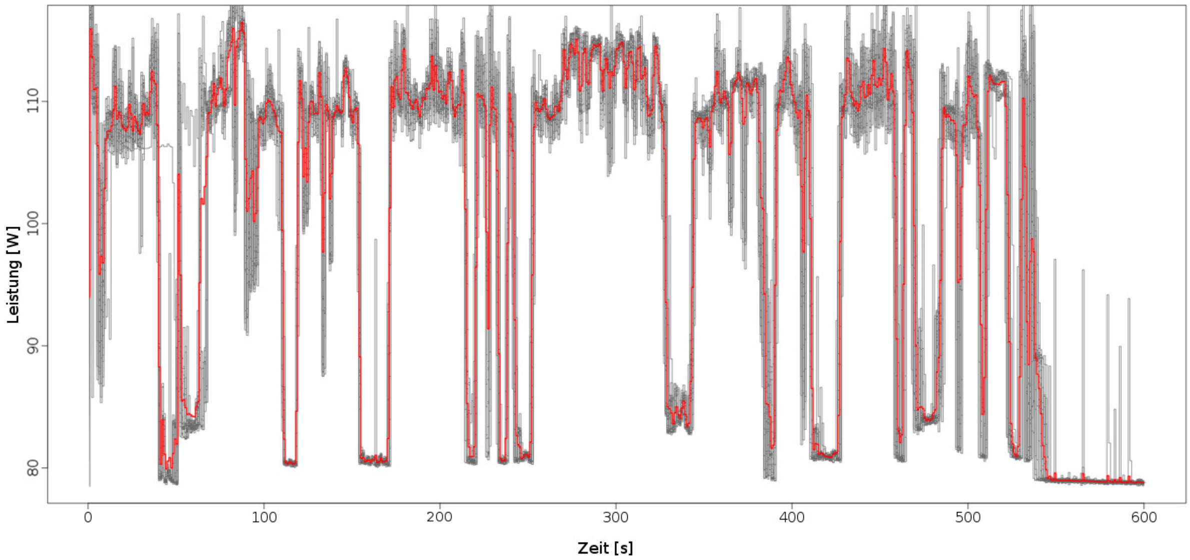
Die Werte der einzelnen Indikatoren für die Softwareprodukte der Fallbeispiele sind in Anhang 4 zu finden. Erwähnenswert ist jedoch der scheinbar proportionale Zusammenhang zwischen Prozessorauslastung und Energieverbrauch (Indikator 1.2.a)).

Kriterium 1.2 Energieeffizienz

Abbildung 16 zeigt eine beispielhafte Darstellung der Leistungsaufnahme des SUT (Client) während der Messung eines Standardnutzungsszenarios für eine Textverarbeitungssoftware. Die Messwerte der 30 Wiederholungen sind in grau dargestellt. Aus den Messungen wurde für jede Sekunde ein Durchschnittswert gebildet (rote Linie). Die in diesem Beispiel errechnete mittlere Standardabweichung der 600 Messungen eines Testdurchlaufs beträgt 2,795 Watt bei einem Mittelwert von 100,315

Watt. Dies zeigt, dass die mittels der automatisierten Lastgenerierung erzeugten Szenarien durchaus für die Operationalisierung der Messungen geeignet sind.

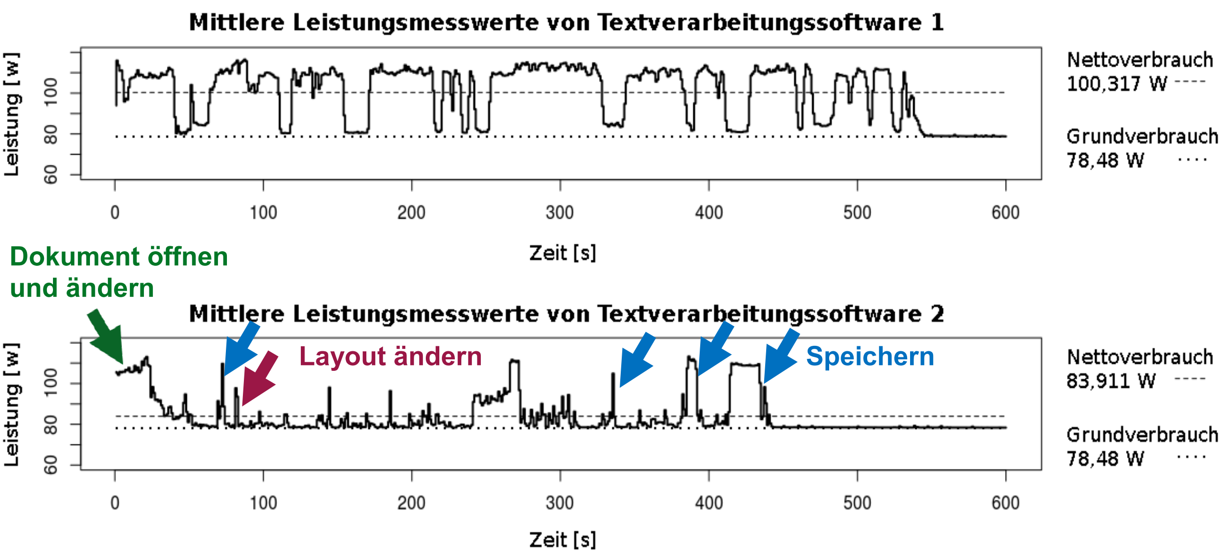
Abbildung 16: Aus 30 Wiederholungen gemittelte Leistungsaufnahme der Messung des Standardnutzungsszenarios für eine Textverarbeitungssoftware



Quelle: Eigene Darstellung, Hochschule Trier

Zum Vergleich des Energieverbrauchs der beiden Textverarbeitungsprogramme zeigt Abbildung 17 die mittlere elektrische Leistungsaufnahme beider Softwareprodukte im Vergleich. Zur Vergleichbarkeit mit denen im Kriterienkatalog (Anhang 1, Tabelle 19) definierten Berechnungsgrundlagen sind zusätzlich der jeweilige Nettoverbrauch und der Grundverbrauch des Referenzsystems eingezeichnet. Des Weiteren enthält die Darstellung beispielhafte Angaben zu den verschiedenen Aktionen (Dokument öffnen und ändern, Layout ändern und Dokument speichern), die während des Standardnutzungsszenarios von Textverarbeitungssoftware 2 durchgeführt wurden.

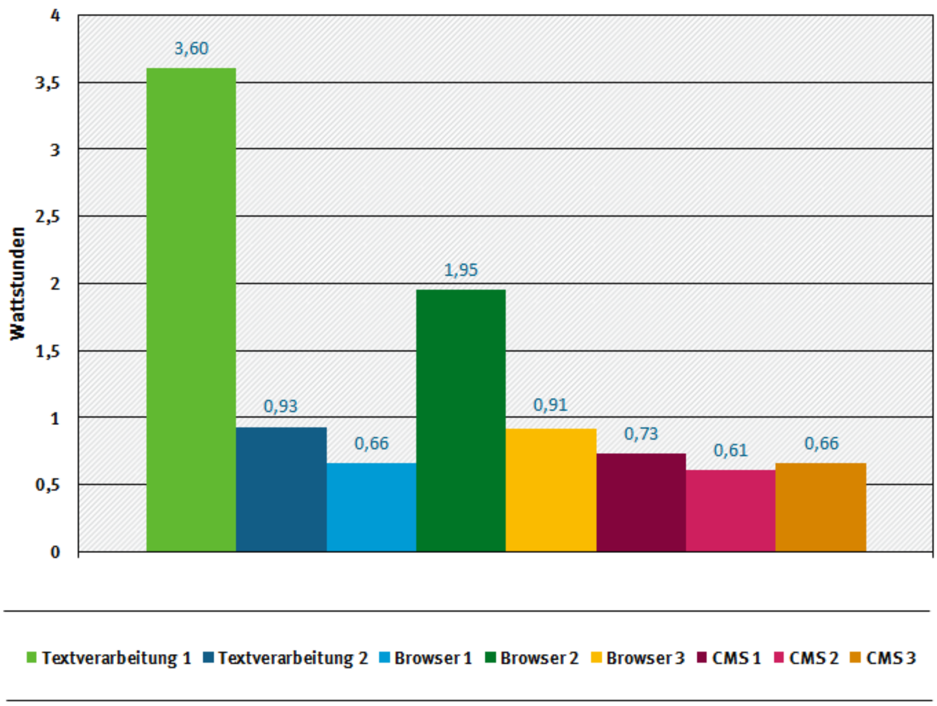
Abbildung 17: Vergleich der Textverarbeitungssoftware: Zeitlicher Verlauf



Quelle: Eigene Darstellung, Hochschule Trier

Als Indikator 1.2.a) wurde die „Messung der zur Ausführung des Standardnutzungsszenarios unter Standardkonfiguration auf dem lokalen Gerät verbrauchten Energie“ festgelegt. Die Ergebnisse der Messungen, die während der Ausführung des jeweiligen Standardnutzungsszenarios der verschiedenen Softwareprodukte durchgeführt wurden, sind in Abbildung 18 dargestellt. Der dargestellte Energieverbrauch beschreibt den Mehrverbrauch, den das System Under Test gegenüber dem Leerlaufzustand aufweist, entsprechend den Berechnungsvorschriften für die Hardware-Inanspruchnahme (vgl. Anhang 1, Tabelle 19). Es lässt sich feststellen, dass die resultierenden Energieverbräuche bei gleichen zu erfüllenden Aufgaben teils sehr stark voneinander abweichen.

Abbildung 18: Vergleich der Energieverbräuche des lokalen Geräts (SUT(Client)) während der Ausführung des Standardnutzungsszenarios

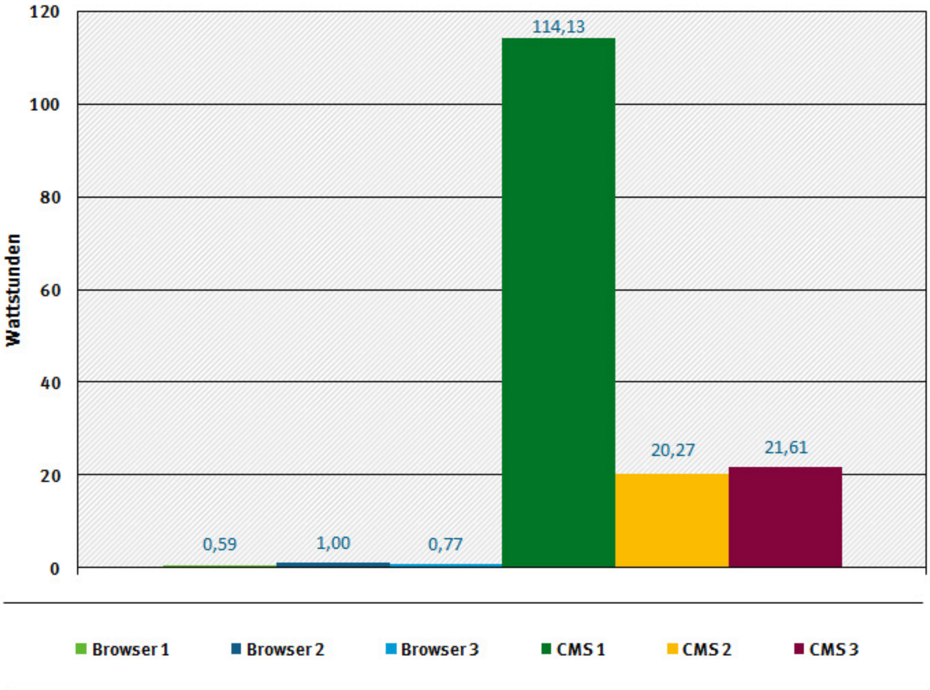


Quelle: Eigene Darstellung, Hochschule Trier

Abbildung 18 stellt den Energieverbrauch der Standardnutzungsszenarien auf dem lokalen System Under Test (Client) der Textverarbeitungsprogramme, Browser und Content Management Systeme gegenüber. Der Energieverbrauch der Content Management Systeme (CMS) liegt innerhalb der Bandbreite von ca. 0,61 bis ca. 0,73 Wattstunden (Wh) relativ dicht beieinander. Bei den Browsern zeigen sich deutlichere Unterschiede mit ca. 0,66 Wattstunden bei Browser 1 und 1,95 Wattstunden bei Browser 2. Am deutlichsten sind die Unterschiede bei den beiden Textverarbeitungsprogrammen mit 0,93 Wattstunden bei Textverarbeitungsprogramm 2 und 3,6 Wattstunden bei Textverarbeitungsprogramm 1. Textverarbeitungsprogramm 1 verbraucht knapp viermal so viel Energie wie das Textverarbeitungsprogramm 2, obwohl beide Programme das gleiche Standardnutzungsszenario durchlaufen.

Ein weiterer Indikator zur Bewertung des Verbrauchs elektrischer Energie ist die verbrauchte Energie, die durch den bei Ausführung des Standardnutzungsszenarios hervorgerufenen Datenverkehr verursacht wird (Indikator 1.2.b)). Dieser wird, im Gegensatz zum vorherigen Indikator, nicht gemessen, sondern auf Basis aktueller Untersuchungen geschätzt. Der Indikator wird nur bei Anwendungen mit entfernter Datenhaltung sowie mit entfernter Verarbeitung ausgewertet (vgl. Abbildung 19). Bei den hier untersuchten Softwaretypen ist dies bei den Browsern und den Content Management Systemen der Fall.

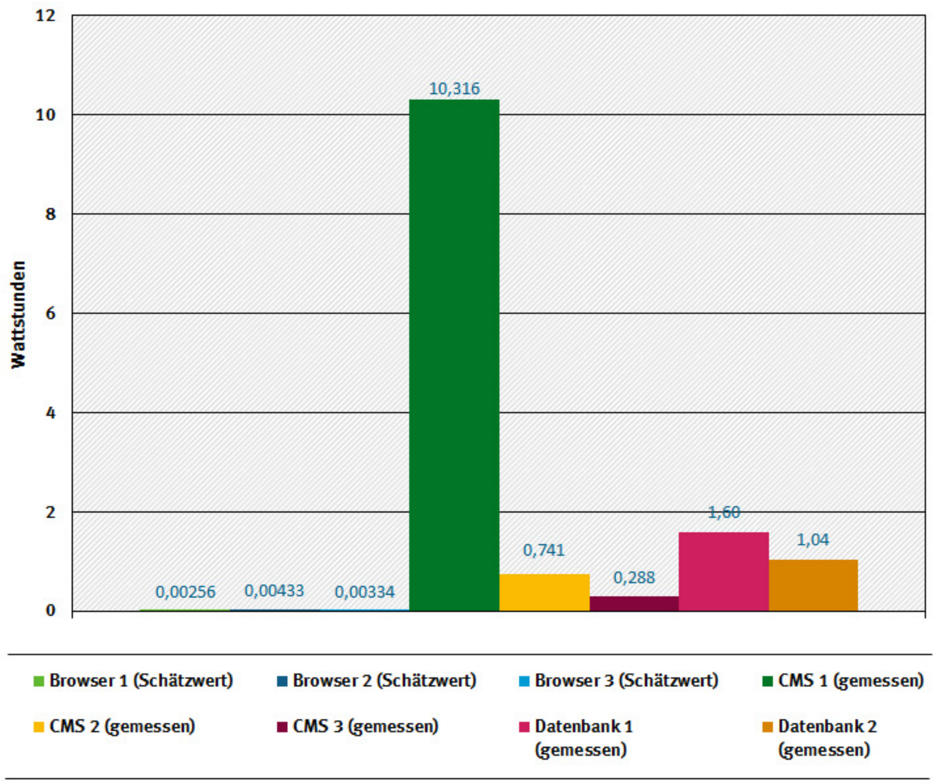
Abbildung 19: Durch den Datenverkehr verursachter Energieverbrauch im Übertragungsnetz zur Ausführung des Standardnutzungsszenarios (Schätzung)



Quelle: Eigene Darstellung, Hochschule Trier

Ergänzend zum lokal verursachten Energieverbrauch (Indikator a)) und dem durch den Datenverkehr verursachten Energieverbrauch (Indikator b)) kann die Energie, die durch die entfernte Speicherung und Verarbeitung von Servern verbraucht wird (Indikator 1.2.c)) ebenfalls einbezogen werden. Im Fall der Browser werden Schätzungen herangezogen, die resultierende Werte für Content Management Systeme wurden gemessen (vgl. Abbildung 20).

Abbildung 20: Energieverbrauch durch die entfernte Speicherung und Verarbeitung in Servern



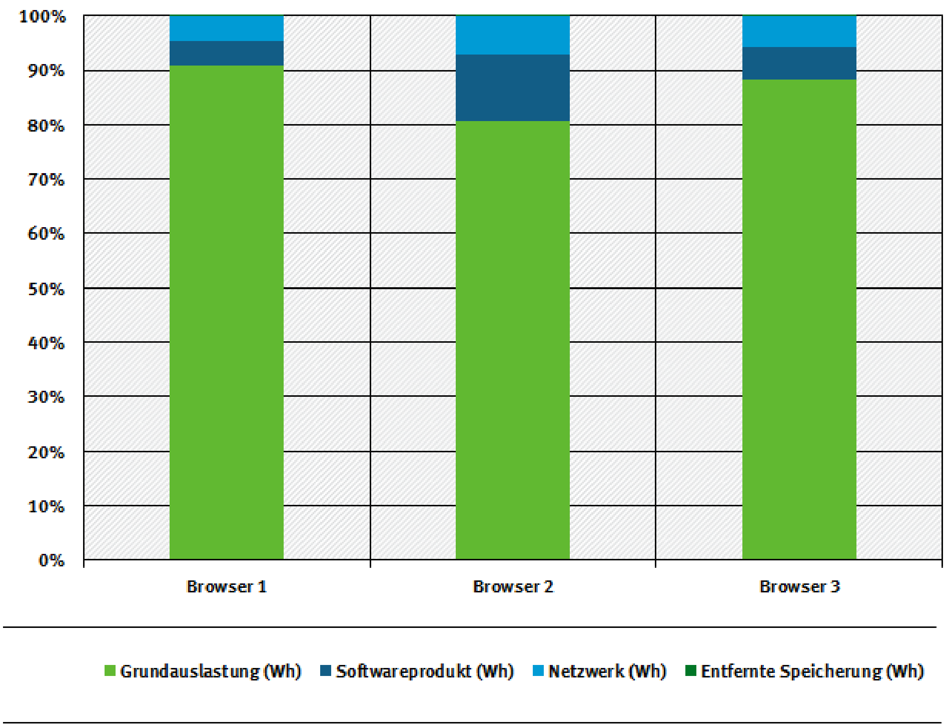
Quelle: Eigene Darstellung, Hochschule Trier

Zusammenfassend stellt Abbildung 21 alle Indikatoren des Kriteriums 1.2, exemplarisch anhand der Produktgruppe „Browser“, summiert dar. Abgebildet sind die prozentualen Werte, um die verschiedenen Verbräuche ins Verhältnis zueinander zu setzen.

Als Grundauslastung wird in Abbildung 21 der Energieverbrauch im Leerlauf auf dem lokalen System Under Test verstanden. Der Energieverbrauch des Softwareprodukts ist der lokale Mehrverbrauch, der durch die Software verursacht wird (vgl. Abbildung 18), der Energieverbrauch des Netzwerks ist der auf Grundlage der übertragenen Datenmenge berechnete Wert (vgl. Abbildung 19) und der Energieverbrauch für entfernte Speicherung und Verarbeitung der Daten ist ein Schätzwert (vgl. Abbildung 20).

Bei der Interpretation der Größenverhältnisse fällt auf, dass die Grundauslastung mit 80 bis 91 Prozent den wesentlichen Anteil am Energieverbrauch ausmacht. Lässt man den Grundverbrauch außer Acht und untersucht die übrigen Verbräuche, so stellt man fest, dass der lokale Mehrverbrauch des Softwareproduktes in einer ähnlichen Größenordnung liegt, wie der Energieverbrauch im Netzwerk. Bei Browser 1 liegen der lokale Energieverbrauch und der Netzwerk-Energieverbrauch mit rund 4 Prozent gleichauf. Bei Browser 3 liegt der Unterschied zwischen 5 und 6 Prozent. Bei Browser 2 ist der Unterschied etwas ausgeprägter er liegt bei 7 Prozent für das Netzwerk und 12 Prozent für den lokalen Mehrverbrauch. Aus diesen Verhältnissen lässt sich schließen, dass der Energieverbrauch im Netzwerk eine relevante Größe beim Gesamtenergieverbrauch eines Softwareprodukts darstellt.

Abbildung 21: Darstellung aller Indikatoren des Kriteriums 1.2 im Verhältnis zueinander, exemplarisch für die Browsermessungen bzw. -schätzungen



Quelle: Eigene Darstellung, Hochschule Trier

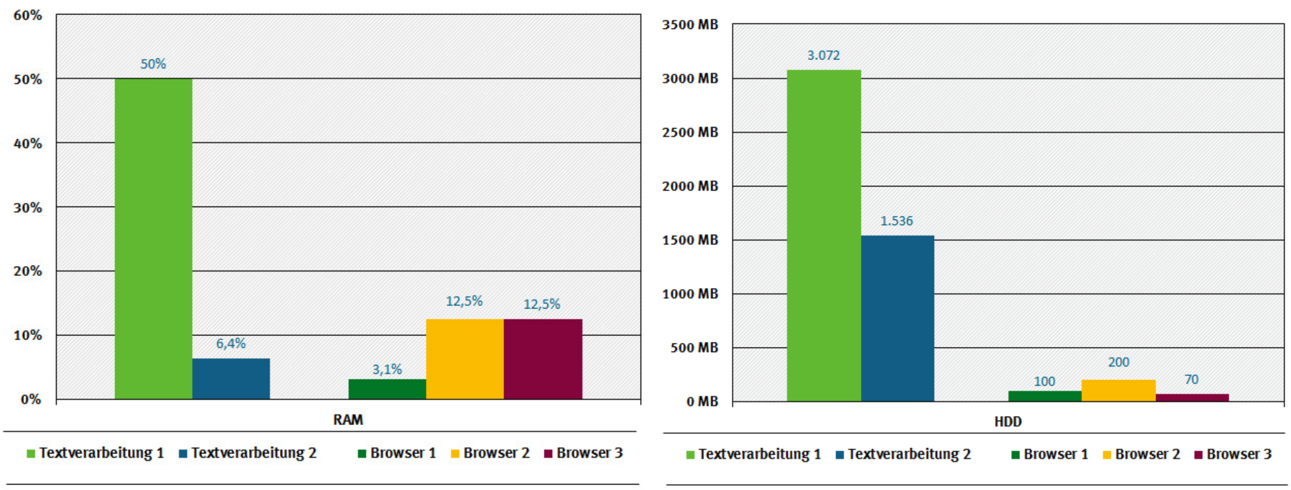
4.3.2 Exemplarische Mess- und Prüfergebnisse weiterer Kriterien

Im Folgenden werden ausgewählte Anwendungsergebnisse der weiteren Kriterien vorgestellt. Auch hier erfolgt die Darstellung entlang der Reihenfolge der Kriterien.

Beispiel 1: Kriterium 1.1.2 Minimale Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)

Für Indikator 1.1.2 a) Minimale lokale Rechenleistung konnte in vielen Fällen, insbesondere für quelloffene Softwareprodukte, keine Angabe der erforderlichen Prozessor-Taktfrequenz in GHz, notwendige Anzahl an Kernen und Busbreite gefunden werden. Viele Hersteller geben stattdessen eher Rechnerarchitektur-Voraussetzung (z. B. „Pentium-kompatibler PC (Pentium III, Athlon oder aktueller)“) an. Für die Softwareprodukte der Fallbeispiele wurden seitens der Entwickler auch keine Angaben zu Indikator 1.1.2 e) Minimale Bandbreite für den Netzzugang gemacht. Aussagen waren jedoch für die erforderliche Mindestausstattung an lokalem Arbeitsspeicher (RAM) und für den lokalen Permanentspeicher (HDD) verfügbar. Abbildung 22 zeigt die Systemvoraussetzungen für RAM und HDD der Softwareprodukte „Textverarbeitung“ und „Browser“. Die Prozentangaben für den lokalen Arbeitsspeicher sind auf die Ausstattung des Messsystems (siehe Abschnitt 4.1.2) mit 4 GB RAM bezogen. Die Systemvoraussetzungen für Permanentspeicher sind in absoluten Werten angegeben.

Abbildung 22: Vergleich minimaler Systemvoraussetzungen (RAM und HDD)



Quelle: Eigene Darstellung, Hochschule Trier

Beispiel 2: Kriterium 1.1.5 Sparsame Hardwarenutzung durch Anpassbarkeit und Unterstützung der Nutzenden bei der Anpassung des Softwareprodukts

Für Kriterium 1.1.5 werden die Ergebnisse der Indikatoren a) bis c) vorgestellt:

Indikator a) „Geschieht die Minimierung der beanspruchten Kapazitäten automatisch und/oder gibt es bei der Installation der Software entsprechende Optionen? (Skala: ja/nein)“: In den Fallbeispielen geschah eine automatische Minimierung der beanspruchten Kapazitäten bei keinem der Produkte vollumfänglich. Datenbanksysteme legen generell schon einen sehr hohen Wert auf Effizienz und bieten auch viele Möglichkeiten die Einstellungen dahingehend anzupassen. Beide Textverarbeitungsprogramme, sowie zwei der CMS bieten zumindest teilweise Optionen zur Konfiguration der Software bei der Installation.

Indikator b) „Falls die Nutzenden eine entsprechende Wahl treffen, ist die Entscheidung für oder gegen Installationsoptionen später jederzeit revidierbar (Skala: ja/nein)“ Bei allen Produkten, die entsprechende Einstellungen anbieten, konnte diese auch später verändert werden.

Indikator c) „Black-Box-Test: Können hardwareintensive Software-Module abgeschaltet werden? (Skala: können abgeschaltet werden/können nicht abgeschaltet werden)“ Bei den **Textverarbeitungsprogrammen** wurden die Funktionen „Automatisches Speichern“ und „Automatische Rechtschreibprüfung“ separat als Black-Box gemessen. Das automatische Speichern erzeugt einen Leistungspeak, die Rechtschreibprüfung hatte keine messbaren Auswirkungen auf den Verbrauch. Beide Funktionen sind deaktivierbar. Bei den **Browsern** konnte durch geschicktes Wählen der bei jedem Start geöffneten Seite eine teilweise deutliche Einsparung erzielt werden. So konnte z. B. durch Ersetzen der medial sehr aufwändigen, in der Standardeinstellung des Browsers 3 voreingestellten Startseite durch eine einfach gestaltete und werbefreie Webseite der Nettoenergieverbrauch im Leerlauf von ca. 1 Wh auf ca. 0,1 Wh reduziert werden. Ebenso reduzierte sich der Netzwerk-Traffic im Leerlauf um über 90 %. Bei **CMS** konnten keine deaktivier- oder anpassbaren, hardwareintensiven Funktionen oder Module identifiziert werden. **Datenbanksysteme** erlaubten es z.B. Indizierungsroutinen anzupassen, die Ordnungsrelationen zum schnelleren Suchen in den Tabellen generieren, abzuschalten.

Beispiel 3: Kriterium 1.1.6 Online-Auslieferung

Zur Auswertung der Indikatoren wurden aus den Herstellerangaben in der offiziellen Dokumentation oder der Produktwebsite ermittelt, ob Auslieferung und Update online möglich waren. Anschließend wurde die Downloadfunktionalität geprüft und die Größe des Downloads notiert. Falls eine ältere Version der Software verfügbar war, wurde diese installiert und es wurde geprüft, ob ein nachträgliches Update auf die aktuell neuste Version möglich war. Des Weiteren wurde geprüft, ob die heruntergeladenen Installationsdateien so abgelegt werden konnten, dass das Softwareprodukt ohne weiteren Download auch auf anderen Geräten installiert werden kann. Als Ergebnis stellte sich heraus, dass für alle untersuchten Produkte eine Online-Auslieferung zur Verfügung steht (Indikator a)) und alle Produkte außer einem Browser das lokale Ablegen der Installationsdaten unterstützen (Indikator b)).

Beispiel 4: Kriterium 1.3.1 Anpassung der beanspruchten Kapazitäten an den Bedarf

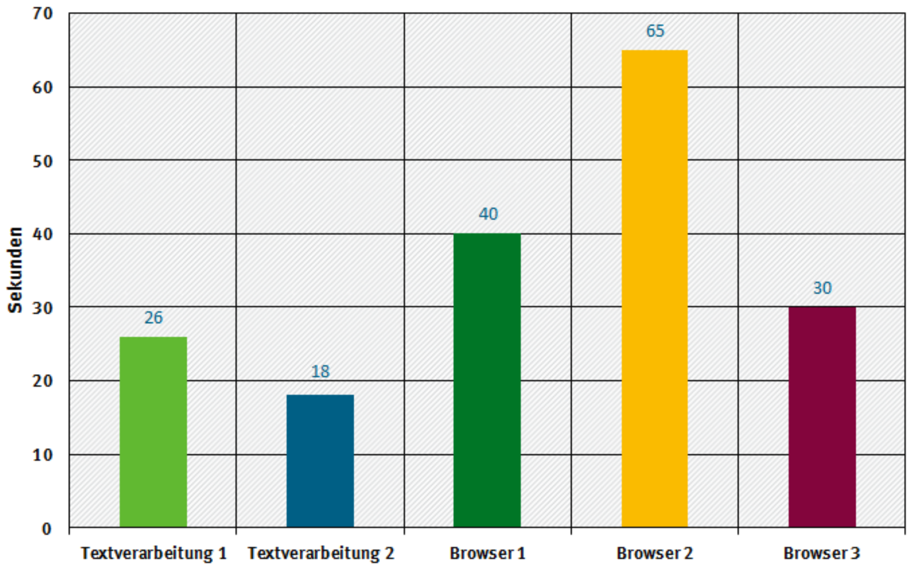
Als Teil des Kriteriums 1.3 Ressourcenmanagement beantwortet Kriterium 1.3.1 die Frage, ob das Softwareprodukt im laufenden Betrieb Hardwarekapazitäten freigibt (und damit auch seinen Energieverbrauch reduziert), wenn diese nicht benötigt werden.

Dazu werden die Ergebnisse der Indikatoren a) bis c) vorgestellt:

Indikator a) „Verfügt das Softwareprodukt über unterschiedliche Modi, deren Wechsel sich messbar auf den Energieverbrauch auswirkt?" Es wurden zunächst verschiedene Modi des Produkts identifiziert und anschließend der Energieverbrauch analog 1.2.1 für die Ausführung eines Szenarios in den identifizierten Modi gemessen. Dabei ließen sich zwar einige Modi identifizieren (z. B. Lesemodus, Bearbeitungsmodus in Textverarbeitungsprogrammen, Privater Modus in Browsern etc.), eine Veränderung des Energieverbrauchs konnte allerdings durch die Messungen nicht nachgewiesen werden.

Indikator b) „Wechselt das Softwareprodukt dynamisch jeweils in einen sparsameren Modus, wenn das möglich ist? (z.B. Schlafmodus)" Es wurde der Energieverbrauch eines Szenarios gemessen, bei dem nach einer Zeit keine weiteren Aktionen durchgeführt wurden. Dabei wurde beobachtet, ob und nach welcher Zeit das Produkt nach der letzten Aktion in einen sparsamen Modus wechselt. Die Ergebnisse sind in Abbildung 23 zusammengefasst.

Abbildung 23: Wartezeit bis zur Aktivierung des Energiesparmodus der Software bei Standardeinstellungen



Quelle: Eigene Darstellung, Hochschule Trier

Dabei ist darauf hinzuweisen, dass sowohl die Content Management Systeme als auch die Datenbanken keinen „Sparmodus“ enthalten. Entsprechend trat auch keine messbare Erhöhung der Energieaufnahme im Leerlauf im Vergleich zum Grundverbrauch auf.

Indikator c) „Falls für die Software spezielle Einstellungen für einen sparsamen Betriebsmodus vorgenommen werden müssen, sind diese Konfigurationsmöglichkeiten zentral und allgemein verständlich zusammengefasst?“ Bei keinem der untersuchten Softwareprodukte sind energierelevante Einstellungen an einer zentralen Stelle zusammengefasst.

Beispiel 5: Kriterium 1.3.4 Feedback zur Beanspruchung von Hardwarekapazitäten und Energie

Zur Beantwortung der Frage, ob die vom Softwareprodukt aktuell beanspruchten lokalen und entfernten Hardwarekapazitäten und resultierenden Energieverbräuche angezeigt werden, wurde zunächst das Softwareprodukt mit Standardeinstellungen installiert. Anschließend wurde in der Produktdokumentation sowie in den Optionen des Produkts nach dem Vorhandensein entsprechender Anzeigen recherchiert.

Indikator a) „Werden die beanspruchten Hardwarekapazitäten, der Datenfluss und die Energieverbräuche angezeigt? (funktionsspezifisch vorhanden mit Vorschlägen zur Ressourceneinsparung, funktionsspezifisch vorhanden, für das Gesamtprodukt vorhanden, nicht vorhanden)“ Es wurde festgestellt, dass lediglich die Produkte der Browser-Gruppe über Anzeigen der beanspruchten Hardwarekapazitäten verfügten. Browser 1 bietet einen Taskmanager, in dem sich Angaben über die derzeitige Speicherbelegung, CPU-Auslastung und Netzwerknutzung der einzelnen Tasks anzeigen lassen. Browser 2 und 3 ermöglichen die Überwachung der Netzwerkauslastung und des Arbeitsspeichers.

Indikator b) „Einschätzung des Prüfers, ob Anzeige korrekt (korrekt / nicht korrekt)“. Es wurde stichprobenartig die angezeigte Kapazitätsauslastung mit der gemessenen verglichen und festgestellt, dass, falls entsprechende Anzeigen vorhanden waren, diese korrekte Werte lieferten.

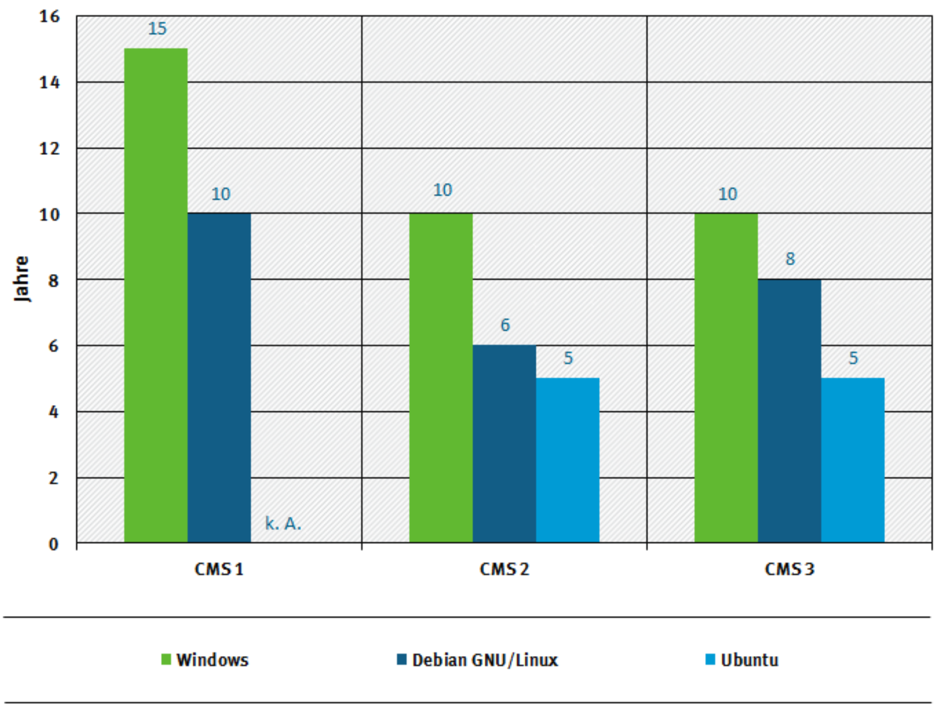
Beispiel 6: Kriterium 2.1 Abwärtskompatibilität

Als Teil von Kriterium 2 „Potenzielle Hardware-Nutzungsdauer“ tragen die folgenden Beispiele zur Beantwortung der Frage bei, bis zu welchem Grad Hardware-Erneuerungszyklen von Software-Erneuerungszyklen entkoppelt sind. Dabei zeigt Kriterium 2.1 auf, wie alt das Referenzsystem sein darf, sodass der Hersteller des Softwareprodukts garantiert, dass das aktuelle Release noch lauffähig ist.

Indikator a) „Zunächst Herstelleraussage (Hardware, ältere Betriebssysteme, ältere Frameworks), da für zurückliegende Jahre keine Standardkonfigurationen definiert sind“. Die Werte wurden wie in 4.1.3, Beispiel 2 beschrieben aufgenommen. Beispielsweise ist Datenbank 2 den Angaben nach auf einem 14 Jahre alten Windows System lauffähig, da sie mindestens Windows 2000 SP4 benötigt, was 2003 veröffentlicht wurde.

Abbildung 24 vergleicht die Ergebnisse des Indikators für die verfügbaren Betriebssysteme am Beispiel der Content Management Systeme (CMS). Für das CMS 1 konnten nur die beiden Betriebssysteme Windows und Debian GNU überprüft werden, während für Ubuntu keine Angaben (k.A.) gemacht werden können.

Abbildung 24: Abwärtskompatibilität der Content Management Systeme nach Betriebssystem



Quelle: Eigene Darstellung, Hochschule Trier

Indikator b) entfällt, da noch keine früheren Standardkonfigurationen vorhanden sind.

Beispiel 7: Kriterium 2.2 Plattformunabhängigkeit und Portabilität

Kriterium 2.2 beantwortet die Frage, ob das Softwareprodukt auf verschiedenen aktuell verbreiteten produktiven Systemumgebungen (Hardware und Software) betrieben werden kann und ob die Nutzenden zwischen diesen ohne Nachteil wechseln können.

Indikator a) „Herstellerangaben (kompatibel mit verschiedenen Betriebssystemen, Laufzeitumgebungen)" Tabelle 8 zeigt das Ergebnis der Recherche.

Tabelle 8: Betriebssystemkompatibilität der Softwareprodukte

	Textverarbeitungsprogramm		Browser			CMS			Datenbank		
Betriebssystem	1	2	1	2	3	1	2	3	1	2	3
Windows	X	X	X	X	X	X	X	X	X	X	X
Linux	--	X	X	X	--	X	X	X	X	X	X
OS X	X	X	X	X	--	X	X	X	X	X	--

Legende: (X = kompatibel, -- = nicht kompatibel)

Indikator b) „Standardnutzungsszenario auf verschiedenen aktuell verbreiteten produktiven Systemumgebungen ausführen, dabei die Daten- und Softwareeinstellungsportabilität prüfen".

Mit Ausnahme von Textverarbeitungsprogramm 1, welches eine unterschiedliche Benutzungsoberfläche aufweist, die sich aber nicht nachteilig auf die Nutzung auswirkt, bieten alle Produkte identische Einstellungsoptionen und Funktionalitäten in den unterstützten Betriebssystemen.

Beispiel 8: Kriterium 3.1.2 Transparenz und Interoperabilität der Programme

Die restlichen Beispiele sind Teil von Kriterium 3 „Nutzungsautonomie“. Sie tragen zur Beantwortung der Frage bei, ob der Hersteller des Softwareprodukts die Autonomie des Nutzenden im Umgang mit dem erworbenen Produkt respektiert.

Kriterium 3.1.2 „Transparenz und Interoperabilität“ zeigt auf ob Anwendungs-Programmier-Schnittstellen (APIs) klar dokumentiert sind und ob die Verbreitung und Weiterentwicklung des Programms unterstützt wird.

Es werden die Ergebnisse der Indikatoren b) und c) vorgestellt:

Indikator b) „Ist der Quellcode vollständig offengelegt?“ Dies ist ein Beispiel für einen einfachen ja/nein Indikator. Die Resultate für die in den Fallbeispielen untersuchten Produkte sind in Tabelle 9 zu finden.

Indikator c) „Ist die Software unter einer Lizenz veröffentlicht, die es erlaubt, die Software weiterzuentwickeln?“

Tabelle 9: Offenlegung des Quellcodes und Lizenzmodell

	Textverarbeitungsprogramm		Browser			CMS			Datenbank		
	1	2	1	2	3	1	2	3	1	2	3
b)	nein	ja	nein	ja	nein	ja	ja	ja	ja	ja	nein
c)[9]	nein	ja (MPL, LGPL v. 3, GPL v. 3+)	nein, große Teile werden aber in einem open-source Projekt verfügbar gemacht.	ja (MPL, GPL, LGPL)	nein	ja (GPL v2+)	ja (GPL v2+)	ja (GPL)	ja (GPL v2)	ja (eigene Lizenz, kompatibel mit GPL)	nein

Beispiel 9: Kriterium 3.1.3 Kontinuität des Softwareproduktes

Kriterium 3.1.3 beantwortet die Frage, ob das Softwareprodukt über längere Zeiträume genutzt werden kann, ohne dass schwerwiegende Nachteile (insbesondere Probleme der IT-Sicherheit) auftreten und ob der Nutzende die Wahl hat, unnötige Updates zu vermeiden.

Es werden die Ergebnisse der Indikatoren a), b) und d) vorgestellt:

Indikator a) „Wie lang ist der Zeitraum, für den der Anbieter die zukünftige Unterstützung des Produkts mit Sicherheitsupdates garantiert?“ Bei fast allen quelloffenen Produkten, für die in den Fallbeispielen dieser Indikator erhoben wurde, wurden vom Hersteller keine Angaben zu diesem Zeitraum gegeben. Allerdings wurden in diesen Fällen auch stets kontinuierlich und kostenlos Updates für die Produkte angeboten, sodass der Indikator hier nicht zutrifft. Für die übrigen Fälle zeigt Tabelle 10 einen Überblick.

9 Abkürzungen der Lizenzen:
- Mozilla Public License (MPL): https://www.mozilla.org/en-US/MPL/
- GNU Lesser General Public License (LGPL): https://www.gnu.org/licenses/#LGPL
- GNU General Public License (GPL): https://www.gnu.org/licenses/#GPL

Tabelle 10: Zeitraum, für den der Hersteller Sicherheitsupdates garantiert

Textverarbeitungs-programm		Browser			CMS			Datenbank		
1	2	1	2	3	1	2	3	1	2	3
ca. 11 Jahre	k. A.	k. A.	k. A.	Keine genauen Angaben vorhanden	k. A.	k. A.	3 Jahre für LTS und kontinuierliche, kostenlose Veröffentlichung von Updates	k. A.	k. A.	Unbegrenzt

Legende: k. A. = keine Angabe

Indikator b) „Reagiert der Hersteller zeitnah auf das Bekanntwerden von Sicherheitslücken?" Anhand von Recherchen zu bekannten, offenen und geschlossenen Sicherheitslücken (z. B. bei CVE Details[10], in öffentlichen Issue-Trackern etc.) wurde vom Prüfenden eine Einschätzung vorgenommen, ob der Hersteller zeitnah auf Sicherheitslücken reagiert. Tabelle 11 zeigt die Übersicht der von CVE Details aufgenommenen Sicherheitslücken in 2015 und 2016, die als Grundlage für die Einschätzung dienten. Maßstab ist dabei der CVSS Score (Common Vulnerability Scoring System), der auf einer Skala von 0 bis 10 den Schweregrad von Sicherheitslücken und deren Behebungsdauer berücksichtigt, wobei 10 dem höchsten Schwergrad entspricht.

Tabelle 11: Sicherheitsrisiken in den Softwareprodukten der Fallbeispiele

Daten aus CVE Details	Textverarbeitungsprogramm		Browser			CMS			Datenbank		
	1	2	1	2	3	1	2	3	1	2	3
Anzahl Fälle 2015	40	5	187	179	231	13	11	4	1	3	605
Gewichtete, mittlere CVSS Score 2015	9,6	6,6	7,2	7,2	9	6,76	5,5	5	5	6,3	5,9
Anzahl Fälle 2016	29	3	159	133	122	6	20	6	0	4	782
Gewichtete, mittlere CVSS Score 2016	9,1	9	6,8	8,8	7,3	6,55	5,5	4,3	k.A.	7,8	6,1

Indikator d) „Besteht die Möglichkeit, nur differenzielle Updates zu erhalten?" Dies ist ebenfalls ein Beispiel für einen einfachen ja/nein Indikator, jedoch wurde im Verlauf der Recherchen festgestellt, dass in einigen Fällen ein Unterschied zwischen der Auslieferung von neuen Versionen (nicht differenziell) und von Updates innerhalb einer Version (i.A. differenziell) besteht. Die Resultate für die in den Fallbeispielen untersuchten Produkte sind in Tabelle 12 zu finden.

10 http://www.cvedetails.com/ bietet das Common Vulnerability Scoring System (CVSS), das als eine Möglichkeit herangezogen wurde um IT-Sicherheitslücken zu bewerten. Dabei wurde insbesondere die Schwere der Lücken und deren Behebungsdauer berücksichtigt. Weitere Informationen unter http://www.first.org/cvss.

Tabelle 12: Bereitstellung differentieller Updates

Textverarbeitungsprogramm		Browser			CMS			Datenbank		
1	2	1	2	3	1	2	3	1	2	3
nein*	nein	ja	ja	nein*	ja	nein	nein	ja	nein*	konnte nicht festgestellt werden

* neue Versionen nicht differenziell, Updates innerhalb einer Version i.A. differenziell.

Beispiel 10: Kriterium 3.2.1 Deinstallierbarkeit der Programme

Kriterium 3.2.1 beantwortet die Frage, ob Nutzende ausreichend darin unterstützt werden, das Softwareprodukt rückstandsfrei zu deinstallieren.

Indikator a) „Deinstallation der Software und Vergleich mit dem Zustand vor der Installation, der gleich sein muss." Anhand eines Black-Box Tests wurde festgestellt ob nach der Deinstallation Dateien, Ordner und, im Falle der Windows-Betriebssysteme, Einträge in der Registrierungsdatenbank (Registry) auf dem SUT verblieben. Die Ergebnisse werden in Tabelle 13 dargestellt. Dabei ist festzuhalten, dass der Indikator bei serverseitig ausgeführten Systemen, wie z. B. Content Management Systemen (CMS) nicht sinnvoll angewendet werden konnte, da die Deinstallation nicht über einen eigenen Deinstallationsprozess, sondern durch das manuelle Entfernen der Dateien und Datenbanken geschieht.

Tabelle 13: Verbleibende Dateien und Registry-Einträge nach Deinstallation

Textverarbeitungsprogramm		Browser			CMS			Datenbank		
1	2	1	2	3	1	2	3	1	2	3
14 Ordner, 1 Datei, 100+ Einträge in Windows Registry	*	29 Dateien	19 Dateien	Deinstallation unmöglich	nicht sinnvoll anwendbar	nicht sinnvoll anwendbar	nicht sinnvoll anwendbar	*	*	*

* Zustand vor Installation und nach Deinstallation identisch

Beispiel 11: Kriterium 3.3.1 Datenwiederherstellbarkeit

Kriterium 3.3.1 beantwortet die Frage, ob sich der letzte Zustand der Daten nach einem unerwünschten Programmabbruch wiederherstellen lässt.

Indikator a) „Macht der Hersteller dazu Angaben, und lassen sich diese im Test bestätigen" Datenwiederherstellbarkeit war bei beiden **Textverarbeitungsprogrammen** verfügbar (Auto-Speichern). Ebenso erlaubten alle **Browser** das automatische Speichern der geöffneten Seiten/Tabs. Alle **CMS** boten eine Entwurfsfunktion, jedoch nur CMS 2 speicherte Änderungen am momentan geöffneten Artikel alle 60s automatisch und warnte beim Verlassen der Seite, wenn noch nicht gespeicherte Änderungen vorlagen. Beide anderen CMS boten kein automatisches Speichern der Artikel. Bei den **Datenbanksystemen** ist das Kriterium nicht sinnvoll anwendbar.

Beispiel 12: Kriterium 3.5.2 Ressourcenrelevanz der Produktinformation

Kriterium 3.5.2 beantwortet die Frage, ob die Produktinformation alle Angaben, die die Nutzenden benötigen, um die Ressourcenbeanspruchung durch das Softwareprodukt gering zu halten, in strukturierter Form enthalten, und ob die Angaben korrekt sind.

Indikator a) „Qualitative Beurteilung der Vollständigkeit und Verständlichkeit“. In der Gruppe der Browser sind Hinweise im Hinblick auf die Ressourcenrelevanz in den Produktinformationen zu finden. Browser 1 und 2 machen Angaben, wie eine Reduktion des benötigten Arbeitsspeichers erreicht werden kann. Bei serverbasierten Systemen und insbesondere bei den Datenbanken finden sich im Regelfall auch Einstellungsmöglichkeiten für Speicher und Prozessorressourcen, sowie Hinweise zur Effizienzsteigerung.

4.4 Weitere Hinweise zur Anwendbarkeit des Kriterienkatalogs

4.4.1 Hinweise zur vorgeschlagenen Mess- und Prüfmethodik

Ziel der Messungen ist es nach Möglichkeit den gesamten, durch das Softwareprodukt induzierten, Ressourcenverbrauch zu berücksichtigen. Es ergeben sich jedoch durch die in Abschnitt 3.1.2 beschriebenen Klassen von Softwareprodukten einige Einschränkungen, die die Systemgrenzen je nach Klasse verschieben können:

- Da bei lokalen Anwendungen keine Daten über Netzwerke übertragen werden (sollten), müssen keine Ressourcen von etwaigen Serverdiensten einbezogen werden. Alle Softwarekomponenten sind auf einem einzigen Computersystem installiert und werden als Black-Box gemessen.
- Reine Serverdienste beanspruchen neben den Ressourcen zum Betrieb der Hardware des Servers auch solche zur Übertragung der Daten von und zu Clients. Da die gemessene Software auf dem Server ausgeführt wird, wird auch dessen Ressourcenverbrauch, genau wie bei Messungen lokaler Anwendungen, als Black-Box gemessen. Der durch die Datenübertragung hervorgerufene Energieverbrauch wird mittels eines Schätzwertes anhand der übertragenen Datenmenge ermittelt und fließt so ebenfalls in die Gesamtbewertung ein (vgl. Kriterium 1.2, Indikator b)).
- Bei Anwendungen mit entfernter Datenhaltung oder Datenverarbeitung kommen neben den Ressourcen zur Datenübertragung noch Verbräuche externer Systeme hinzu, die nicht direkt gemessen werden können. Hier wird, neben der Schätzung des durch die Datenübertragung hervorgerufenen Energieverbrauchs, auch der durch Datenverarbeitung und Datenhaltung auf externen Systemen hervorgerufene Ressourcenverbrauch durch einen Schätzwert approximiert (vgl. Kriterium 1.2, Indikator c)).

Des Weiteren werden in den Laborversuchen Peripheriegeräte (wie z. B. Monitore, Drucker, Scanner etc.) nicht mit gemessen. Dies ändert sich jedoch auch gegebenenfalls mit dem Versuchsaufbau, bspw., wenn mobile Geräte oder Laptops gemessen werden und der Monitor nicht deaktiviert werden kann.

4.4.2 Festlegung von Referenzsystemen

Für den Zeitpunkt der Messung muss aus Gründen der Vergleichbarkeit ein Referenzsystem festgelegt werden. Diese Festlegung eines solchen Standardsystems erfolgt aus dem Grund der repräsentativen Darstellung der Leistung eines Standardrechners. Dabei soll keine rückwirkende Betrachtung durchgeführt werden, und das Referenzsystem soll für kommende Jahre auf Basis eines standardisierten Verfahrens in vergleichbarer Form ermittelt werden können. Zur Ermittlung der Kriterien, die von einem Referenzsystem abhängen (wie beispielsweise die Systemanforderungen aus Kriterium 1.1.1 und 1.1.2, aber auch die Messungen der Energieaufnahme und der Auslastung der Hardwarekapazitäten) wurden für die Fallbeispiele im Rahmen dieses Vorhabens die im Labor vorhandenen System Under Test (SUT, vgl. Tabelle 6) als Referenzsysteme angesehen. Für weitere, zukünftige Messungen, die außerhalb dieses Forschungsprojektes durchgeführt werden können, wird in Abschnitt 6.1 ein Referenzsystem „Arbeitsplatzcomputer für Büro-Software“ definiert und Regeln festgelegt, wie dieses System fortgeführt werden kann.

4.4.3 Festlegung der Standardkonfiguration

Die Standardkonfiguration umfasst das Betriebssystem und ggf. weitere zum Betrieb benötigte Software sowie das Hardware-Referenzsystem. Die Software-Standardkonfiguration wird idealerweise als Festplattenimage an zentraler Stelle gespeichert. So kann nach jeder Messung eines Softwareprodukts derselbe Zustand auf dem SUT durch das erneute Aufspielen des Images wiederhergestellt werden. Anschließend, nach Aufsetzen des frischen Systems, wird die nächste zu vergleichende Software installiert. Durch dieses Verfahren ist die Integrität der Standardkonfiguration sichergestellt.

4.5 Bewertung der Kriterien und Indikatoren hinsichtlich Aussagekraft und Umsetzbarkeit

Ein zentraler Punkt der Aufnahme der Kriterien für die Softwareprodukte der Fallbeispiele ist die Bewertung der Operationalisierung hinsichtlich Anwendbarkeit und Aussagekraft der Kriterien. Tabelle 14 zeigt die Ergebnisse des Bewertungsprozesses seitens der Auftragnehmer. Die Bewertung reicht von „sehr gut anwendbar, bzw. sehr hohe Aussagekraft“ (+++) über „weder positive noch negative Anwendbarkeit, bzw. Aussagekraft“ (0) bis „sehr schwierig anwendbar, bzw. sehr geringe Aussagekraft“ (---)

Tabelle 14: Einschätzung der Anwendbarkeit und Aussagekraft der entwickelten Kriterien

	Anwendbarkeit	Aussagekraft
1.1.1 Empfohlene Systemvoraussetzungen	0	+
1.1.2 Minimale Systemvoraussetzungen	++	++
1.1.3 Hardware-Auslastung im Leerlauf	+	++
1.1.4 Hardware-Auslastung bei normaler Nutzung (Standardnutzungsszenario)	-	+++
1.1.5 Sparsame Hardwarenutzung durch Anpassbarkeit	--[1]	+
1.1.6 Online-Auslieferung	+++	--
1.2 Energieeffizienz	--[2]	++
1.3.1 Anpassung der beanspruchten Kapazitäten an den Bedarf	+	+
1.3.2 Anpassung des Bedarfs an die verfügbaren Kapazitäten	--	0[3]
1.3.3 Ressourcenschonende Standardeinstellungen	++	+
1.3.4 Feedback zur Beanspruchung von Hardwarekapazitäten u. Energie	++	+
2.1 Abwärtskompatibilität	+[4]	++
2.2 Plattformunabhängigkeit und Portabilität	++[5]	+
2.3 Hardwaresuffizienz	0	++
3.1.1 Transparenz der Datenformate und Datenportabilität	+++	++

	Anwendbarkeit	Aussagekraft
3.1.2 Transparenz und Interoperabilität der Programme	+++	++
3.1.3 Kontinuität des Softwareproduktes	++	++
3.1.4 Transparenz des Prozessmanagements	++	+
3.2.1 Deinstallierbarkeit der Programme	+	++
3.2.2 Löschbarkeit der Daten	-[6]	0
3.3.1 Datenwiederherstellbarkeit	++	++
3.3.2 Selbstreparaturfähigkeit	0	+
3.4.1 Offlinefähigkeit	+++	++
3.5.1 Verständlichkeit und Überschaubarkeit der Produktinformationen	+[7]	++
3.5.2 Ressourcenrelevanz der Produktinformation	++	+++[8]

[1] Indikatoren c) und e) sind schwierig zu erheben
[2] Messgerät wird benötigt. Schätzung.
[3] schwer zu erheben, nur bei mobilen Geräten relevant
[4] Indikator b) bisher nicht bewertet. Erfassung erfordert definierte ältere Referenzsysteme.
[5] Software muss ggf. auf vielen Plattformen getestet werden
[6] Indikatoren a) und c) schwierig zu prüfen
[7] Test mit Nutzern kann beliebig aufwändig sein
[8] Ggf. großes Optimierungspotential und Möglichkeit, Ressourcenrelevanz zu kommunizieren

Folgende Kriterien sind unseres Erachtens besonders geeignet, um potenzielle Optimierungspotentiale der Softwareprodukte aufzudecken:

- Kriterium 1.1.2) Angaben zu minimalen Systemvoraussetzungen schaffen einen direkten Zusammenhang zwischen Softwareprodukt und Hardwarebedarf.
- Kriterium 1.1.3) Die Leerlaufauslastung der Hardware sollte im Idealfall verschwindend gering sein. Ist dies nicht der Fall deutet dies auf Optimierungspotentiale hin.
- Kriterium 1.1.4) Die gemessenen Hardwarekapazitäten (Prozessor, RAM, Festplatte (HDD) sowie Netzwerklast) bei normaler Nutzung bedingen einen Großteil des erzeugten Energiebedarfs und sind z.T. für weitere Berechnungen notwendig.
- Kriterium 1.2) Die Energieeffizienz als summarischer Indikator zeigt auf einen Blick die Aggregation der Verbräuche der Hardwareressourcen.
- Kriterien 2.1) „Abwärtskompatibilität“, 2.2) „Plattformunabhängigkeit und Portabilität“ und 2.3) „Hardwaresuffizienz“ zeigten in den Fallbeispielen eine hohe Relevanz bezüglich potenzieller Möglichkeiten der Reduktion des Hardwareverbrauchs.
- Kriterium 3.4.1) Die Tatsache ob Softwareprodukte offline nutzbar sind oder nicht ist ebenfalls ein guter Indikator für Optimierungspotentiale durch Einsparung entfernter Energieverbräuche.
- Kriterium 3.5.1) Wir sehen ggf. große Optimierungspotentiale und insbesondere auch die Möglichkeit Ressourcenrelevanz zu kommunizieren, wenn die Produktinformationen der Softwareprodukte hinsichtlich einer Ressourcenschonenden Nutzung verbessert werden können.

5 Handlungsempfehlungen für die Entwicklung eines Umweltzeichens

Die Ergebnisse der Anwendung der Bewertungsmethodik (siehe Abschnitt 4) machen deutlich, dass sich verschiedene Software-Anwendungen bei gleichem Standardnutzungsszenario in ihrer Ressourceneffizienz, der beanspruchten Hardware, dem mit ihrer Ausführung verbundenen Energieverbrauch und der Nutzungsautonomie deutlich unterscheiden. Die entwickelte Methodik ist daher dazu geeignet, solche Produkteigenschaften festzustellen und Softwareprodukte anhand ihrer Umwelteigenschaften zu differenzieren.

Eine der Fragestellungen dieses Vorhabens (siehe Abschnitt 3.1 Präzisierung der Zielsetzung) ist es, zu klären, ob die Methodik dazu angewendet werden kann, die Vergabekriterien für ein Umweltzeichen (Label) festzulegen. Nach Auswertung der Anwendungsergebnisse kann diese Frage grundsätzlich bejaht werden. Es kann die Empfehlung ausgesprochen werden, dass ein Umweltzeichen für ressourceneffiziente Software entwickelt werden sollte. Software, die in besonderem Maße sparsam mit den Hardwareressourcen umgeht und in ihrer Nutzung einen niedrigen Energieverbrauch aufweist, sollte mit einem Umweltzeichen gekennzeichnet werden können. Weitere Aspekte, die eine langfristige Nutzung der Software sowie Verbraucherschutz berücksichtigen, sollten in solch ein Umweltzeichen integriert werden.

Die Entwicklung und Auswahl von Vergabekriterien für ein Umweltzeichen müssen gemäß der Norm ISO 14024[11] auf Grundlage fundierter wissenschaftlich-technischer Untersuchungen erfolgen. Ein wesentliches Merkmal der Vergabekriterien muss sein, dass die Anforderungen messbar sind und damit eine objektive Unterscheidung der Umweltauswirkungen der Produkte zulassen.

Vergabekriterien eines Umweltzeichens zeichnen sich gemäß der Norm insbesondere durch folgende Eigenschaften aus:

1. Vergabekriterien adressieren die wesentlichen Umweltauswirkungen eines Produktes entlang dessen Produktlebensweges.
2. Kriterien müssen richtungssicher sein, d.h. die Erfüllung der Kriterien muss Vorteile (für Mensch und Umwelt) bieten.
3. Nur solche Produkteigenschaften werden abgefragt, die zu einer Unterscheidung von Produkten beitragen, nicht solche, die von allen Produkten gleichermaßen erfüllt werden.
4. Die Anforderungen müssen mit überprüfbaren Indikatoren hinterlegt sein, die das Kriterium bestätigen (z.B. Überprüfung des Kriteriums Energieeffizienz durch Messung des Energieverbrauchs als Indikator).
5. Zur Quantifizierung der Indikatoren muss auf Prüfvorschriften verwiesen werden, die eine unabhängige und reproduzierbare Überprüfung ermöglichen (z.B. Verweis auf eine Norm oder Vorgabe einer Prüfvorschrift).
6. Formulierung einer Nachweisregelung, in welcher Form die Einhaltung gegenüber einer Vergabestelle nachgewiesen werden muss (z. B. externer Labortest oder Eigenerklärung).

Der bestehende Kriterienkatalog zur Bewertung von Softwareprodukten (siehe Anhang 1) kann diese Anforderungen teilweise erfüllen. Grundsätzlich adressieren die Kriterien die Umweltwirkungen von Software und eine Nachweisführung über die Messung oder Überprüfung der Indikatoren ist möglich. Bei der Anwendung der Bewertungsmethodik in den Fallbeispielen (siehe Abschnitt 4) ist jedoch auch deutlich geworden, dass nicht alle Kriterien gleichermaßen für eine unabhängige Überprüfung von Software geeignet sind bzw. mit einem vertretbaren Aufwand bewertet werden können. Die Überlegungen zur Bewertung der jeweiligen Kriterien ist in Abschnitt 4.5 dokumentiert. In Tabelle 15 wird vor diesem Hintergrund ein eingeschränkter

[11] DIN EN ISO 14024 :2001-02, Umweltkennzeichnungen und -deklarationen (Umweltkennzeichnung Typ I) - Grundsätze und Verfahren

Kriterienkatalog dargestellt, der mögliche Kriterien aus der Bewertungsmethodik benennt, die potenziell für ein Umweltzeichen geeignet erscheinen.

Tabelle 15: Software-Kriterien zur potenziellen Anwendung in einem Umweltzeichen

Kriterium	Hersteller-angabe	Messwert	F&E-Bedarf
1 Ressourceneffizienz			
1.1.2 Minimale Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)	x		
1.1.3 Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration		x	x
1.1.4 Hardware-Inanspruchnahme bei normaler Nutzung unter der Annahme einer Standardkonfiguration und eines Standardnutzungsszenarios		x	x
1.2 Energieeffizienz		x	x
2 Potenzielle Hardware-Nutzungsdauer			
2.1 Abwärtskompatibilität	x		
2.2 Plattformunabhängigkeit und Portabilität	x		
2.3 Hardwaresuffizienz	x		x
3 Nutzungsautonomie			
3.1.1 Transparenz der Datenformate und Datenportabilität	x		x
3.1.2 Transparenz und Interoperabilität der Programme	x		
3.1.3 Kontinuität des Softwareproduktes	x		
3.2.1 Deinstallierbarkeit der Programme	x	x	
3.4.1 Offlinefähigkeit	x	x	
3.5.1 Verständlichkeit und Überschaubarkeit der Produktdokumentation, Lizenz- und Nutzungsbedingungen	x		x

Die Kriterien in Tabelle 15 decken dabei die wesentlichen durch die Software-Bewertungsmethodik abgedeckten Fragestellungen ab. Mit Blick auf die Eignung für ein Vergabekriterium unterscheiden sich die ausgewählten Kriterien. Teilweise sind die Kriterien sofort und ohne großen Aufwand anwendbar. Dies sind insbesondere jene Kriterien, die durch die reine Überprüfung von Herstellerangaben nachgewiesen werden können (z.B. 1.1.2, 2.2, 3.1.2 usw.). Solche Kriterien sind prinzipiell auch unmittelbar für die Beschaffung von Softwareprodukten anwendbar.

Anspruchsvoller sind solche Kriterien, deren Überprüfung mit einer Messung verbunden ist.
Für diese Kriterien müssen für ein Umweltzeichen noch reproduzierbare Messvorschriften entwickelt werden. Teilweise sind diese sehr einfach festlegbar (z.B. 3.4.1 Offlinefähigkeit). Im Fall der Kriterien, die ein Referenzsystem (das von dem Referenzsystem „Arbeitsplatzcomputer für Büro-Software“ in Abschnitt 6.1 abweicht) und ein Standardnutzungsszenario erfordern (1.1.3, 1.1.4, 1.2) ist hierfür jedoch noch ein größerer Forschungs- und Entwicklungsaufwand nötig. Weiterer Entwicklungsaufwand ist außerdem nötig für solche Kriterien, die eine qualitative Aussage machen, und diese Aussage noch in eine quantifizierbare Größe übersetzen müssen (insbesondere 3.1.1 und 3.5.1).

Für alle Kriterien gilt, dass für ein Umweltzeichen ein geeignetes Ambitionsniveau festgelegt werden muss. D.h. ab welchem Erfüllungsgrad oder unterhalb welchen Schwellenwertes bezogen auf die einzelnen Kriterien kann ein Softwareprodukt als ressourceneffizient bzw. nachhaltig bezeichnet werden?

Zur Entwicklung eines Umweltzeichens sollten daher noch folgende Vorarbeiten geleistet werden:

- Entwicklung von Standardnutzungsszenarien.
  Abhängig von der zu untersuchenden bzw. mit einem Umweltzeichen zu kennzeichnenden Software müssen geeignete Standardnutzungsszenarien entwickelt werden, die eine typische Nutzung der Software beschreiben. Diese Szenarien sollten idealerweise zusammen mit Software-Entwicklern und Software-Anwendern in Stakeholder-Dialogen entwickelt werden.
- Weitere Festlegungen zur Vereinheitlichung der Messmethoden zur Bestimmung der Hardware-Inanspruchnahme und des Energieverbrauchs von Hardware und Datenübertragung.
- Überprüfung einer größeren Anzahl an Software-Anwendungen anhand des in Tabelle 15 genannten Kriterienkataloges und Ableitung von Mindestanforderungen bzw. Benchmarks für ressourceneffiziente Software.

Auf Grundlage der Ergebnisse dieser weiteren Vorarbeiten sollte dann eine Vergabegrundlage für ein Umweltzeichen abgeleitet werden.

Das Konzept zur Bewertung von Software wurde der für das Umweltzeichen Blauer Engel zuständigen Jury Umweltzeichen auf ihrer Sitzung am 13. Dezember 2017 vorgestellt. Die Jury hat sich für die Entwicklung eines Umweltzeichens für Software ausgesprochen. Das Umweltbundesamt wurde gebeten, weitere Forschungsaktivitäten zur Entwicklung von Vergabekriterien durchzuführen.

6 Bausteine zur weiteren Verwendung der Bewertungsmethodik

Die Anwendung der Bewertungsmethodik anhand von Fallbeispielen, die in Abschnitt 4 beschrieben wird, konnte im Rahmen des Forschungsvorhabens mit projektspezifischen Festlegungen für das Referenzsystem, die Standardnutzungsszenarien und Messaufbauten durchgeführt werden. Da die Bewertungsmethodik nach Abschluss des Forschungsvorhabens auch durch Dritte genutzt werden soll, ist es notwendig, weitere Hilfsmittel zu entwickeln und zur Verfügung zu stellen, die bei der reproduzierbaren Anwendung der Methodik unterstützen. Hierzu wurden im Rahmen des Forschungsvorhabens noch drei weitere Bausteine entwickelt:

- Referenzsystem zur Durchführung von Messungen an Büro-Software
- Software zur Auswertung von Hardwareauslastung und Energieverbrauch und
- Erfassungstool zur Erhebung der Bewertungskriterien

Das Referenzsystem, das in Abschnitt 6.1 dargestellt wird, beschreibt einen Desktop-Computer, der zur Anwendung des Kriterienkatalogs an Büro-Software eingesetzt wird. Die technischen Parameter des Referenzsystems bilden die Bezugsgrößen zur Bestimmung der Indikatoren gemäß Anhang 1 (beispielsweise für Kriterium „1.1.2 b) Minimaler lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems“). Das Referenzsystem wird vom Prüflabor benötigt, das den Kriterienkatalog anwendet oder vom Software-Entwickler der das Verhalten der Software auf dem Referenzsystem beobachten und optimieren möchte. Für den Anwender der Software bieten die technischen Parameter des Referenzsystems Hinweise darauf, ob das Softwareprodukt auf anderen, ähnlich ausgerüsteten Systemen lauffähig ist.

In Abschnitt 6.2 wird eine im Rahmen des Forschungsvorhabens entwickelte Software vorgestellt, die dazu dient, die Hardwareauslastung und den Energieverbrauch einheitlich auszuwerten. Voraussetzung zur Nutzung der Auswertungssoftware ist, dass bereits alle erforderlichen Messungen (Performance und Energiebedarf) am Referenzsystem durchgeführt und in Logfiles (Textdateien, die die Messwerte enthalten) abgelegt wurden. Als Ergebnisse liefert die Auswertungssoftware die mittlere Grundauslastung (GA_i), Leerlaufauslastung (LA_i) und Brutto-Auslastung (BA_i) gemäß Anhang 1 Tabelle 19, die zur Bestimmung der Indikatoren Hardware-Auslastung im Leerlauf (Kriterium 1.1.3), Hardware-Inanspruchnahme bei normaler Nutzung (Kriterium 1.1.4) und Energieeffizienz (Kriterium 1.2) erforderlich sind. Weiterhin kann anhand der Auswertungssoftware überprüft werden, ob die Messwerte plausibel und ausreichend genau sind. Die Auswertungssoftware richtet sich an Prüflabore, die den Kriterienkatalog anwenden und an interessierte Software-Entwickler, die das Verhalten ihrer Software überprüfen möchten.

Als weiterer Baustein wird in Abschnitt 6.3 ein Erfassungstool vorgestellt, mit dem der reduzierte Kriterienkatalog gemäß den Empfehlungen für ein Umweltzeichen aus Abschnitt 5 erfasst und einheitlich abgespeichert werden kann. Das Erfassungstool ist als Tabellenkalkulations-Anwendung programmiert und kann mit Microsoft EXCEL oder LibreOffice Calc bearbeitet werden. Das Erfassungstool bietet eine Exportmöglichkeit der Kriterien als XML-Datei. Diese Datei (ssd-info.xml) kann an die untersuchte Software angehängt, weitergegeben oder in Software-Datenbanken übernommen werden. Das Erfassungstool richtet sich an alle Anwender der Bewertungsmethodik, insbesondere auch an Beschaffer von Software, die mithilfe des Werkzeugs eine einheitliche Übersicht über die Software-Produkteigenschaften erhalten.

6.1 Referenzsystem „Arbeitsplatzcomputer für Büro-Software“

6.1.1 Anforderungen an das Referenzsystem

Die Messungen innerhalb des vorliegenden Projektes erfolgten auf einem exemplarischen Messsystem („System Under Test“, siehe Abschnitt 4.1.2), das im Labor der Hochschule Trier am Standort Umwelt-Campus Birkenfeld vorhanden war. Dadurch ließen sich die Hardware-Auslastung (z.B. CPU, RAM, Netzwerk) und der Energieverbrauch an diesem System untersuchen und zur Beschreibung von prozentualen Auslastungen auf dessen Maximalwerte beziehen.

Damit die Kriterien auch durch jedes beliebige andere Labor bzw. jeden interessierten Anwender oder Antragsteller für ein Umweltzeichen genutzt werden können, ist es erforderlich, einheitliche Referenzsysteme zu definieren. Diese Systeme sind abhängig vom Stand der Technik (d.h. der Zeit) und der untersuchten Softwareklasse (z.B. Büro-Software, Server-Anwendung, Mobiltelefon-Anwendung).

Das Referenzsystem beschreibt einen zum Zeitpunkt der Anwendung der Softwarekriterien aktuellen Computer, wie er in typischen Arbeitsumgebungen der untersuchten Software eingesetzt wird. Dafür werden die technische Parameter des Computers wie CPU-Taktfrequenz und Anzahl der Kerne (Cores), Speicherkapazität (RAM), Festplattengröße und -typ, Leistungsaufnahme sowie Betriebssystem benannt. Das Referenzsystem wird vom Prüflabor benötigt, das die Bewertungskriterien an konkreten Softwareprodukten anwendet. Durch die Nutzung des definierten Referenzsystems ist sichergestellt, dass die Messergebnisse reproduzierbar sind.

Das Referenzsystem der Hardware muss außerdem folgende Anforderungen erfüllen:

- Die technische Ausstattung muss repräsentativ für den jeweiligen Stand der Technik des Referenzjahres bei einem typischen Anwender der Software sein.
- Es muss die Anforderungen an die Arbeitsumgebungen zur Anwendung der Software erfüllen (z.B. Betriebstemperaturen, Geräuschemissionen, elektrische Sicherheit).

- Ein Prüflabor oder Anwender des Kriterienkataloges muss in der Lage sein, das System tatsächlich zu beschaffen (keine Prototypen, Spezialanfertigungen oder unverhältnismäßig teure Hardware).
- Das Referenzsystem muss konkret mit Hersteller und Modellnamen bezeichnet werden (und nicht rein generisch anhand von technischen Parametern).
- Die zu untersuchende Software muss auf dem Referenzsystem (inklusive dem zu benennenden Betriebssystem) lauffähig sein.

Die *Büro-Software* wird voraussichtlich die erste Softwareklasse sein, die durch ein Umweltzeichen oder die öffentliche Beschaffung adressiert wird. Aus diesem Grund wird im Rahmen dieser Untersuchung ein Referenzsystem „Arbeitsplatzcomputer für Büro-Software“ entwickelt. Büro-Software könnten beispielsweise Produkte zur Textverarbeitung, Bildbearbeitung, Internetbrowser, Lernprogramme oder Datenbank-Anwendungen (z.B. Fachanwendungen zur Lohnabrechnung) sein.

6.1.2 Auswahl eines Referenzsystems

Um das Referenzsystem möglichst praxisnah zu wählen, wurden Arbeitsplatzcomputer ausgewählt, die von einer zentralen Beschaffungsstelle der öffentlichen Hand in der Vergangenheit tatsächlich beschafft wurden. Die Computer wurden im Rahmen von öffentlichen Ausschreibungen als wirtschaftlichste Angebote zur Erfüllung der jeweiligen Leistungsparameter identifiziert. Die Daten des Referenzsystems basieren auf dem sogenannten „BW-PC“ (Baden-Württemberg-Personal-Computer)[12], der seit dem Jahr 2007 kontinuierlich von der Albert-Ludwigs-Universität Freiburg technisch spezifiziert, öffentlich ausgeschrieben und per Rahmenvertrag an Hochschulen des Landes Baden-Württemberg vertrieben wird. Die Computer der Reihe BW-PC zeichnen sich durch eine besonders hohe Energieeffizienz aus, bei für die typische Büroanwendung angemessen ausgewählter technischer Leistungsfähigkeit. Die Reihe hat in Hochschulen des Landes Baden-Württemberg eine weite Verbreitung und wird kontinuierlich fortgeführt. Es erscheinen in regelmäßigen Abständen neue Modelle, die technische Neuerungen und Möglichkeiten berücksichtigen. Hersteller der BW-PC sind (je nach Ergebnis der durchgeführten Ausschreibungen) typischerweise Markenhersteller, wie Hewlett Packard (HP), Dell oder Fujitsu.

Ausgehend von den in der Vergangenheit beschafften Geräten der Reihe BW-PC wurde als dynamisches Referenzsystem „Arbeitsplatzcomputer für Büro-Software“ eine Zeitreihe beginnend ab dem Jahr 2010 gebildet. Fehlende Angaben zu den Hardware-Komponenten und -parametern wurden durch Informationen aus Hersteller-Datenblättern geschlossen. Angaben zum Energieverbrauch der Computer konnten Energiedatenblättern der entsprechenden Hersteller entnommen werden.

Tabelle 16 und Tabelle 17 nennen für das jeweilige Betrachtungsjahr die technischen Parameter und das Computer-Modell. Bei der Anwendung des Kriterienkatalogs für nachhaltige Software (siehe Anhang 1) muss das genutzte Referenzsystem („System Under Test“) genannt werden. In der Regel sollte das aktuellste, verfügbare Referenzsystem genutzt und das Betrachtungsjahr genannt werden. Für die Angabe der Abwärtskompatibilität (Kriterium 2.1) ist es entscheidend, dass das Softwareprodukt auf einem älteren Referenzsystem lauffähig ist. Zukünftig soll die Zeitreihe weiter fortgeführt und die Tabelle stets um die neuen Daten der BW-PC ergänzt bzw. aktualisiert werden. Ziel ist es pro Jahr die Daten eines aktuellen Referenzsystems zur Verfügung zu stellen.

[12] https://www.bw-pc.uni-freiburg.de/

Tabelle 16: Beschreibung des Referenzsystems als Zeitreihe 2010 – 2013

Technische Parameter	2010	2011	2012	2013
Hersteller	HP	HP	HP	Fujitsu
Modell	8000 Elite CMT[13]	8200 Elite CMT[14]	8200 Elite SFF[15]	Esprimo P920[16]
Prozessor	Intel Pentium E5400	Intel i3-2100	Intel Pentium E5700	Intel i5-3470
Cores	2	2	2	4
Taktfrequenz	2,7 GHz	3,1 GHz	3,0 GHz	3,2 GHz
RAM	2 GB (DDR3, 1.333MHz)	2 GB (DDR3, 1.333MHz)	2 GB (DDR3, 1.333MHz)	4 GB (DDR3, 1.333MHz)
Festplatte (HDD / SSD)	HDD SATA II 250 GB (3 Gbit/s)	HDD SATA II 250 GB (3 Gbit/s)	HDD SATA III 250GB (6 Gbit/s)	HDD SATA III 500GB (6 Gbit/s)
Grafikkarte	Intel GMA 4500	Intel HD Graphics 2000	Intel HD Graphics 2000	Intel HD Graphics 2500
Netzwerk		LAN GigaBit	LAN GigaBit	LAN GigaBit
Betriebssystem	Win 7	Win 7	Win 7	Win 7
Netzteil		90%	90%	91% - 94%
Idle state [W]	42,08	31,27	31,43	24,8
Sleep mode [W]	2,89	2,4	2,37	1,23
Off mode [W]	0,8	1,11	1,07	0,35
TEC [kWh/a]	152,57	115,96	116,32	89,1

Tabelle 17: Beschreibung des Referenzsystems als Zeitreihe 2014 – 2017

Technische Parameter	2014	2015	2016	2017
Hersteller	Fujitsu	Fujitsu	Fujitsu	Fujitsu
Modell	Esprimo P920[17]	Esprimo P920[18]	Esprimo P956[19]	Esprimo P957[20]
Prozessor	Intel i5-4590	Intel i5-4590	Intel i5-6500	Intel i5-6500
Cores	4	4	4	4
Taktfrequenz	3,3 GHz	3,3 GHz	3,2 GHz	3,2 GHz

13 Datenblatt: https://support.hp.com/de-de/document/c03150433
14 Datenblatt: https://support.hp.com/de-de/document/c03366133
15 Datenblatt: https://support.hp.com/ca-en/document/c02781693
16 Datenblatt: https://www.bw-pc.uni-freiburg.de/dateien/bwpc4/datenblatt-profpc-esprimo-p910-e85plus-de.pdf
17 Datenblatt: https://sp.ts.fujitsu.com/dmsp/Publications/public/ds-ESPRIMO-P920-0Watt-de.pdf
18 Das Referenzsystem für das Jahr 2015 ist identisch mit dem des Vorjahres 2014
19 Datenblatt: https://sp.ts.fujitsu.com/dmsp/Publications/public/ds-ESPRIMO-P956-E90-de.pdf
20 Datenblatt: https://sp.ts.fujitsu.com/dmsp/Publications/public/ds-ESPRIMO-P957-E90-de.pdf

Technische Parameter	2014	2015	2016	2017
RAM	4 GB (DDR4, 2133MHz)	4 GB (DDR4, 2133MHz)	4 GB (DDR4, 2133MHz)	8 GB (DDR4, 2133MHz)
Festplatte (HDD / SSD)	HDD SATA III 500GB (6 Gbit/s)	HDD SATA III 500GB (6 Gbit/s)	HDD SATA III 500GB (6 Gbit/s)	SSD SATA III 128GB (6 Gbit/s)
Grafikkarte	Intel HD Graphics 4600	Intel HD Graphics 4600	Intel HD Graphics 530	Intel HD Graphics 530
Netzwerk	LAN GigaBit	LAN GigaBit	LAN GigaBit	LAN GigaBit
Betriebssystem	Win 8.1	Win 8.1	Win 8.1	Win 10
Wirkungsgrad Netzteil	91% - 94%	91% - 94%	84% - 92%	84% - 92%
Short idle / long idle [W]	12,9 / 12,1	12,9 / 12,1	12,9 / 12,3	11,6 / 11,2
Sleep mode [W]	0,86	0,86	0,9	0,81
Off mode [W]	0	0	0,1	0,13
TEC [kWh/a]	55,8	55,8	56,5	51,1

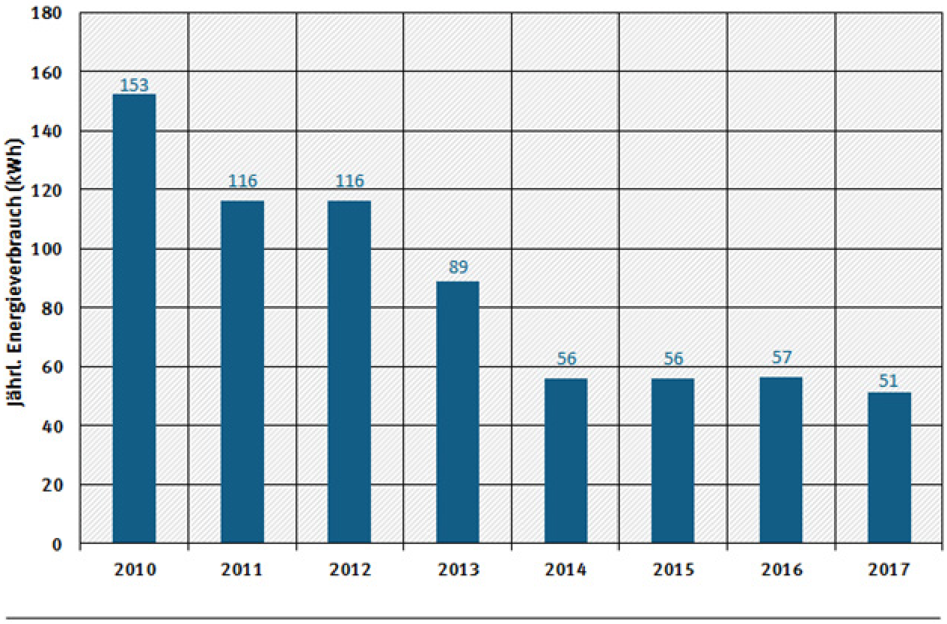
6.1.3 Beschreibung des Referenzsystems

Bei den Computern des Referenzsystems „Arbeitsplatzcomputer für Büro-Software" handelt es sich um professionelle Desktop-Computer überwiegend in Mini-Tower-Gehäusen. Da die Geräte unter dem Gesichtspunkt der Energieeffizienz ausgewählt wurden, sind die Wirkungsgrade der eingesetzten Netzteile sehr hoch und die Energieverbräuche der Rechner vergleichsweise niedrig. Alle Rechner erfüllen die Anforderungen der Energieeffizienzkennzeichnung Energy Star des jeweiligen Jahres, die von der amerikanischen Umweltbehörde EPA vergeben wird.

In den Tabellen zum Referenzsystem (Tabelle 16 und Tabelle 17) werden die Leistungsaufnahmen der Computer in unterschiedlichen Betriebszuständen angegeben. Diese Leistungsangaben wurden von den Herstellern gemäß den Messanforderungen des Energy Star (Program Requirements for Computers) bestimmt. Der „idle state" wird im Energy Star 5.0 als ein Leerlaufzustand definiert, der nach dem Starten des Betriebssystems und anderer Software sowie dem Öffnen eines Nutzer-Accounts einkehrt. Während des Leerlaufzustandes wird keine Anwendersoftware ausgeführt. Der überarbeitete Energy Star 6.1, der seit Juni 2014 Anwendung findet, unterteilt diesen Zustand in „short" und „long idle". Der „short idle" beschreibt dabei gemäß Energy Star 6.0 den Leerlaufzustand 5 Minuten nach Hochfahren des Systems oder Beenden einer aktiven Arbeitslast. Der „long idle" ist der Leerlaufzustand, der sich nach 15 Minuten Inaktivität einstellt. Weitere Modi, die in beiden Ausgaben des Energy Star Anwendung finden, sind der „sleep mode", ein Energiesparmodus, der spätestens nach 30 Minuten der Inaktivität eintreten muss und der „off mode", der bei ausgeschaltetem Computer gemessen wird.

Der Energy Star legt Rechenregeln fest, nach denen der typische jährliche Energieverbrauch der Computer anhand seiner Leistungswerte berechnet und als TEC-Wert (Typical Energy Consumption) ausgedrückt wird. Die Berechnung beinhaltet im Wesentlichen, dass sich der Computer täglich 9,6 Stunden im Leerlaufzustand befindet und die übrige Zeit entweder im Energiesparmodus oder im Auszustand. Der eigentliche Lastzustand, also der Zustand während Anwendungsprogramme ausgeführt werden, wird bei der TEC-Berechnung vernachlässigt. Untersucht man die TEC-Werte der unterschiedlichen Arbeitsplatzcomputer in der oben festgelegten Zeitreihe für das Referenzsystem, so stellt man fest, dass sich der jährliche Energieverbrauch der Computer ausgehend von einem Wert von 153 Kilowattstunden im Jahr 2010 bis zum Jahr 2017 auf ein Drittel (51 Kilowattstunden) verringert hat (siehe Abbildung 25). Die Computer des Referenzsystems sind demnach immer energieeffizienter geworden.

Abbildung 25: Typischer jährlicher Energieverbrauch (TEC) der Referenzsystem-Computer

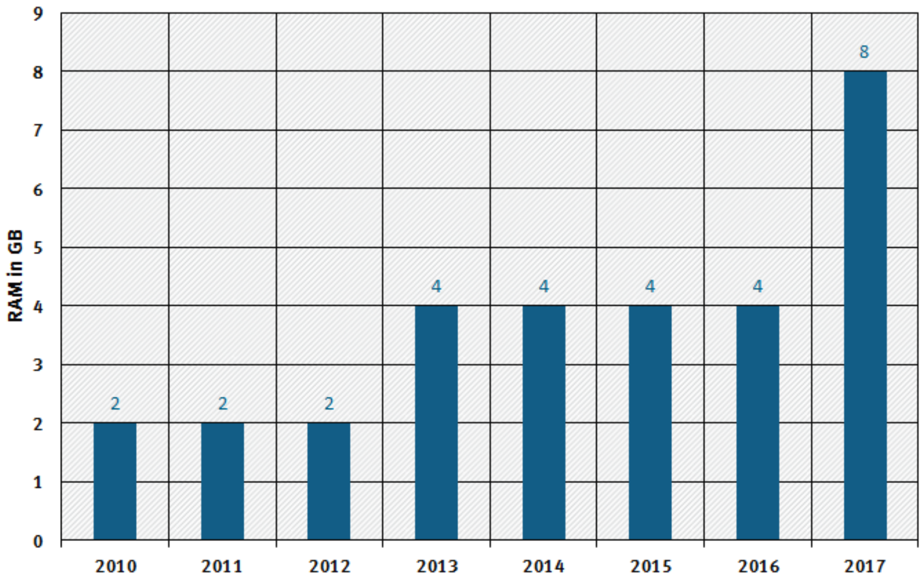


Quelle: Eigene Darstellung, Hochschule Trier

Zusätzlich zum sinkenden Energieverbrauch, geht der Trend der Zeitreihe hin zu steigender Leistungsfähigkeit der PC-Systeme. So ist ein sprunghafter Anstieg der Prozessorleistung von 2012 auf 2013 zu bemerken. Statt CPUs mit zwei Kernen sind ab 2013 CPUs mit vier Kernen verbaut. Damit steigt die Leistungsfähigkeit der CPUs von 2x 3,0 GHz auf 4x 3,2 GHz an, wodurch bei entsprechender Softwareunterstützung und Anwendungsszenario pro Zeiteinheit rund doppelt so viele Rechenoperationen durchgeführt werden können.

Außerdem lässt sich eine Zunahme der Ausstattung mit Arbeitsspeicher (RAM) beobachten (siehe Abbildung 26). Während im Jahr 2010 die Ausstattung mit RAM noch bei 2 Gigabyte liegt, ist sie beim Referenz-PC im Jahr 2017 auf 8 Gigabyte angestiegen.

Abbildung 26: Ausstattung der Referenzsystem-Computer mit Arbeitsspeicher (RAM)



Quelle: Eigene Darstellung, Öko-Institut

Im Jahr 2017 kommt außerdem gemäß Tabelle 17 erstmals statt einer rotierenden Festplatte (Hard Disk Drive – HDD) ein halbleiterbasierter Flash-Speicher (Solid State Drive – SSD) zum Einsatz. SSD-Festplatten zeichnen sich in der Nutzung durch einen geringeren Stromverbrauch, schnelleren Datenzugriff und geringere Störanfälligkeit gegenüber konventionellen HDD-Festplatten aus.

Beim vorinstallierten Betriebssystem gibt es in der Zeitreihe ebenfalls Veränderungen. In den Jahren 2010 bis 2013 wurde das Betriebssystem Microsoft Windows 7 vorinstalliert, im Zeitraum 2014 bis 2016 die Folgeversion Windows 8.1 und im Jahr 2017 die Version Windows 10.

6.2 Auswertungssoftware zu Hardwareauslastung und zum Energieverbrauch

Ziel dieses Arbeitspaketes ist eine Auswertungssoftware zu entwickeln, mit der die Hardware-Auslastung (Indikatoren der Kriterien 1.1.3 und 1.1.4) und der Energieverbrauch (Kriterium 1.2) der zu untersuchenden Software bestimmt werden können. Die Auswertungssoftware soll für die Betriebssysteme Windows und Linux als Open Source-Software zur Verfügung gestellt werden. Die Auswertungssoftware wertet dazu Rohdaten von bereits durchgeführten Messungen der zu untersuchenden Software in Form von Logfiles des Ressourcen- und Energieverbrauchs aus.

Folgende Anforderungen wurden an die Auswertungssoftware gestellt:

- Sie soll anschaulich die Anwendung des Kriterienkatalogs unterstützen, um so die Bewertung der Software zu erleichtern.
- Es soll eine Schritt-für-Schritt Anleitung für die Messungen des Energie- und Ressourcenverbrauchs der Software entwickelt werden.
- Sie soll die Messergebnissen zum Energieverbrauch und zur Hardwareinanspruchnahme in einem Bericht zusammenzuführen, die durch Standard-Benchmarking-Werkzeugen aufgenommen wurden und im CSV-Format vorliegen.
- Aus dem Testablauf sollen die Einzelmessungen aus dem gesamten Messzeitraum extrahiert, die gemessenen Werte aggregiert und schließlich für jede Hardware-Komponente (CPU, RAM, Netzwerk und Festplatte) sowie für den Energieverbrauch ausgewertet werden.

- Sie soll einfache Statistiken und Diagramme bieten, die es ermöglichen, mehrere untersuchte Softwareprodukte miteinander zu vergleichen.

Basierend auf diesen Anforderungen wurden die *Auswertungssoftware* ***OSCAR*** – ***O****pen source-****S****oftware* ***C****onsumption* ***A****nalysis in* ***R*** zur Auswertung von bereits vorliegenden Rohdaten einer Softwaremessung entwickelt. Es können sowohl Datensätze des durch die Ausführung der Software verursachten Energieverbrauchs als auch entsprechende Hardware-Auslastungen ausgewertet werden.

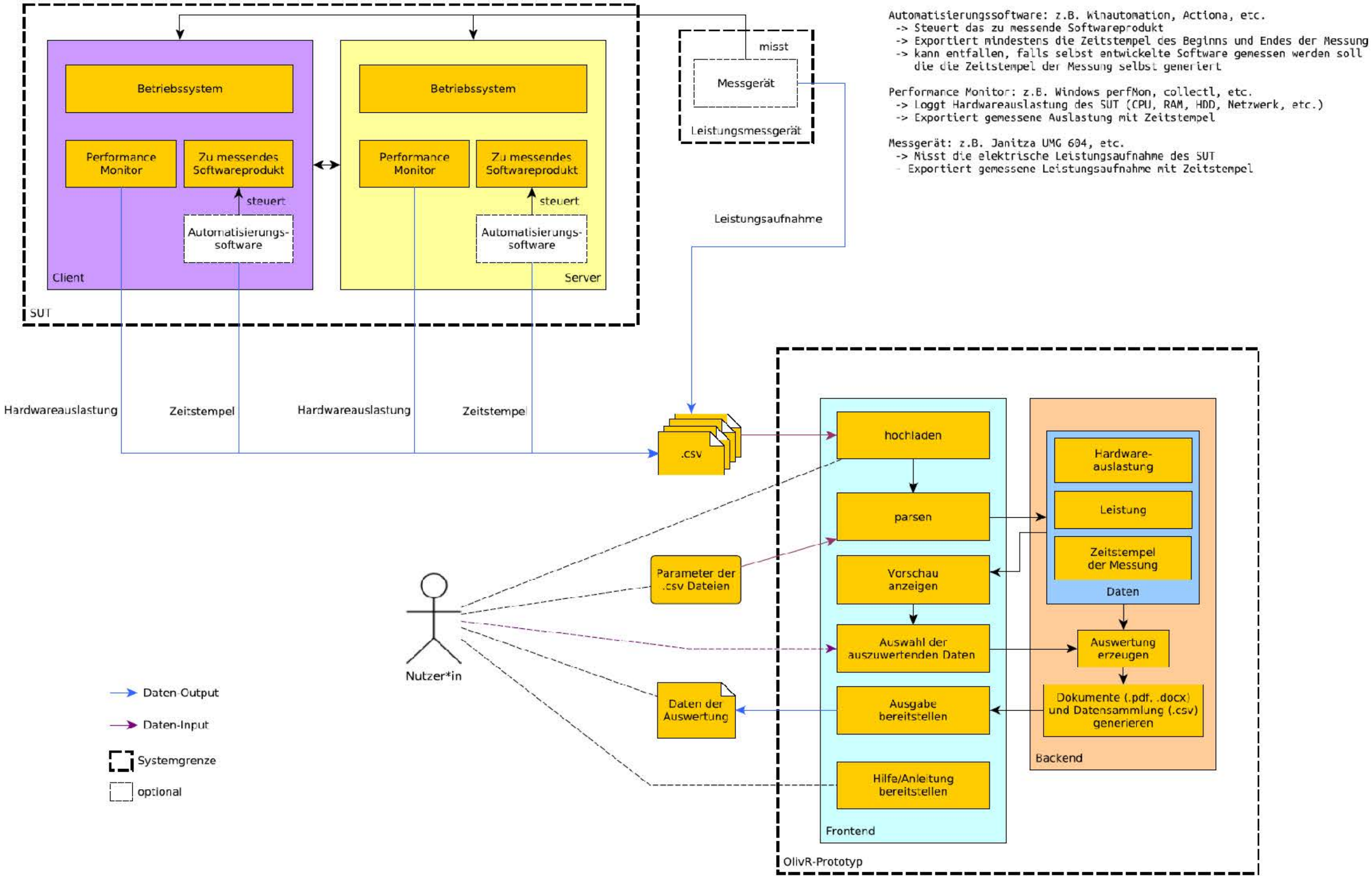
Eine ausführliche Darstellung der Auswertungssoftware OSCAR findet sich in der zusammen mit der Auswertungssoftware zur Verfügung gestellten Schritt-für-Schritt-Anleitung (siehe Anlage zum Forschungsbericht).

Die Schritt-für-Schritt-Anleitung gliedert sich in drei Teile:

- Einordnung und Bezug zum Kriterienkatalog für nachhaltige Software (siehe Abschnitt 3.7 und Anhang 1).
- Anleitung zur Durchführung von Messungen von Energieverbrauch und Hardware-Auslastung von Software mittels Standard-Benchmarking-Werkzeugen.
- Bedienungsanleitung der Auswertungssoftware OSCAR.

Die folgende Abbildung 27 dient als Überblick und visualisiert den generellen Ablauf der Messungen und das Zusammenspiel mit der Auswertungssoftware OSCAR.

Abbildung 27: Übersicht über die Zusammenarbeit und Funktionalitäten der einzelnen Bestandteile für eine Messung (System Under Test (SUT), Messgerät, OSCAR-Auswertungssoftware)



Quelle: Eigene Darstellung, Hochschule Trier

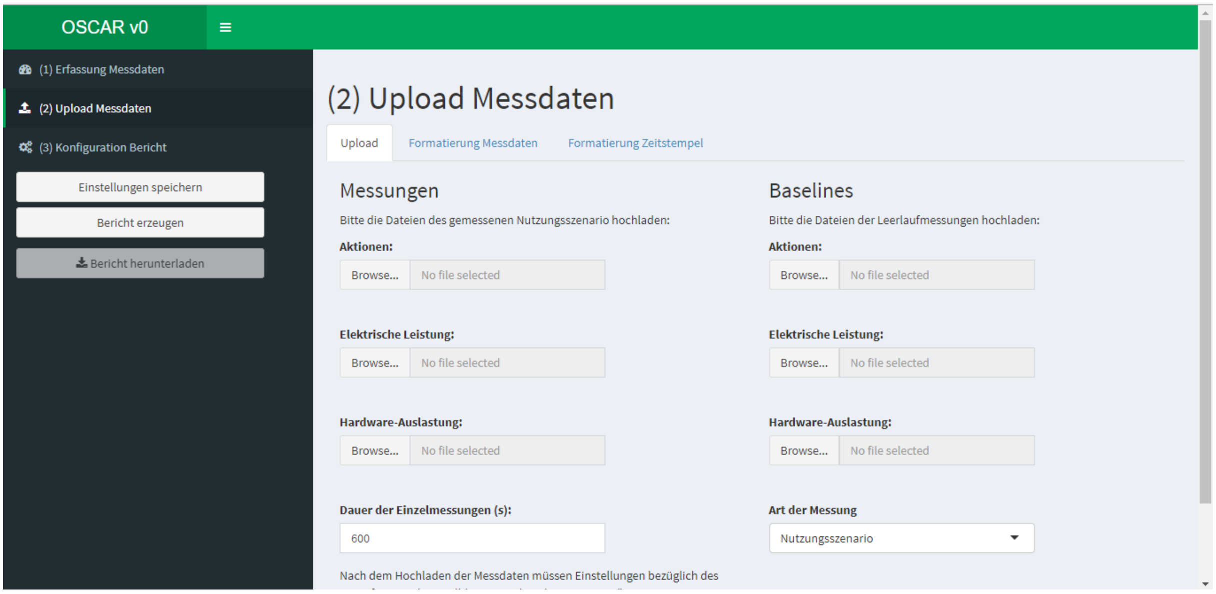
Das Konzept zur Bewertung der Software nach deren Energie- und Ressourcenaufwand gliedert sich in folgende Schritte:

1. Zu Beginn wird das zu untersuchende Softwareprodukt auf dem System Under Test (SUT) installiert und das Standardnutzungsszenario mittels Aufzeichnung und/oder manueller Eingabe von Interaktionen entwickelt.
   Die Messphase beginnt, nachdem relevante Voreinstellungen an Hardwarekomponenten wie dem Messgerät sowie dem Computer konfiguriert wurden und das Szenario anschließend gestartet wurde.
2. Nachdem die Messungen beendet sind, werden die generierten Dateien im Format CSV an das Auswertungstool OSCAR übergeben.
3. Im Auswertungstool müssen noch einige Angaben zu den vorliegenden Daten gemacht werden, wie beispielsweise das Datumsformat und die Zuordnung der Messwerte zu den auszuwertenden Indikatoren. Des Weiteren bietet OSCAR unterschiedliche Möglichkeiten, die Analysen zu individualisieren. Der Messbericht kann als PDF -Dokument heruntergeladen werden.

Die Einzelheiten zur Nutzung werden in der Bedienungsanleitung der Auswertungssoftware OSCAR erläutert.

Abbildung 28 zeigt einen Screenshot der Auswertesoftware OSCAR, in dem die Funktionalitäten zum Hochladen der Dateien mit den Messdaten von den Messungen der Standardnutzungsszenarien sowie ggf. der Baseline-Messung (jeweils Logfile, Verbrauchs- sowie Performancemessungen) zu erkennen sind.

Abbildung 28: Beispiel-Screenshot der Auswertungssoftware OSCAR



Quelle: Eigene Darstellung, Hochschule Trier

6.3 EXCEL-Tool zur Erfassung der Bewertungskriterien

Als weiterer Baustein zur Unterstützung der Anwendung der Bewertungsmethodik wurde ein Erfassungstool entwickelt, mit dem die Indikatoren eines reduzierten Kriterienkataloges abgespeichert und weitergegeben werden können. Das Tool sollte folgende Funktionalitäten vorweisen:

- Benennung des Referenzsystems und der untersuchten Software,
- manuell ausfüllbare Tabelle der Kriterien des reduzierten Kriterienkatalogs,
- Erläuterungen und unterstützende Informationen zu den Kriterien,
- soweit möglich Ausfüllhilfe durch Drop-Down-Felder,
- Exportfunktion der eingegeben Werte in ein XML-Format (ssd-info.xml),
- Importfunktion der Tabellenwerte aus einem XML-Format (ssd-info.xml).

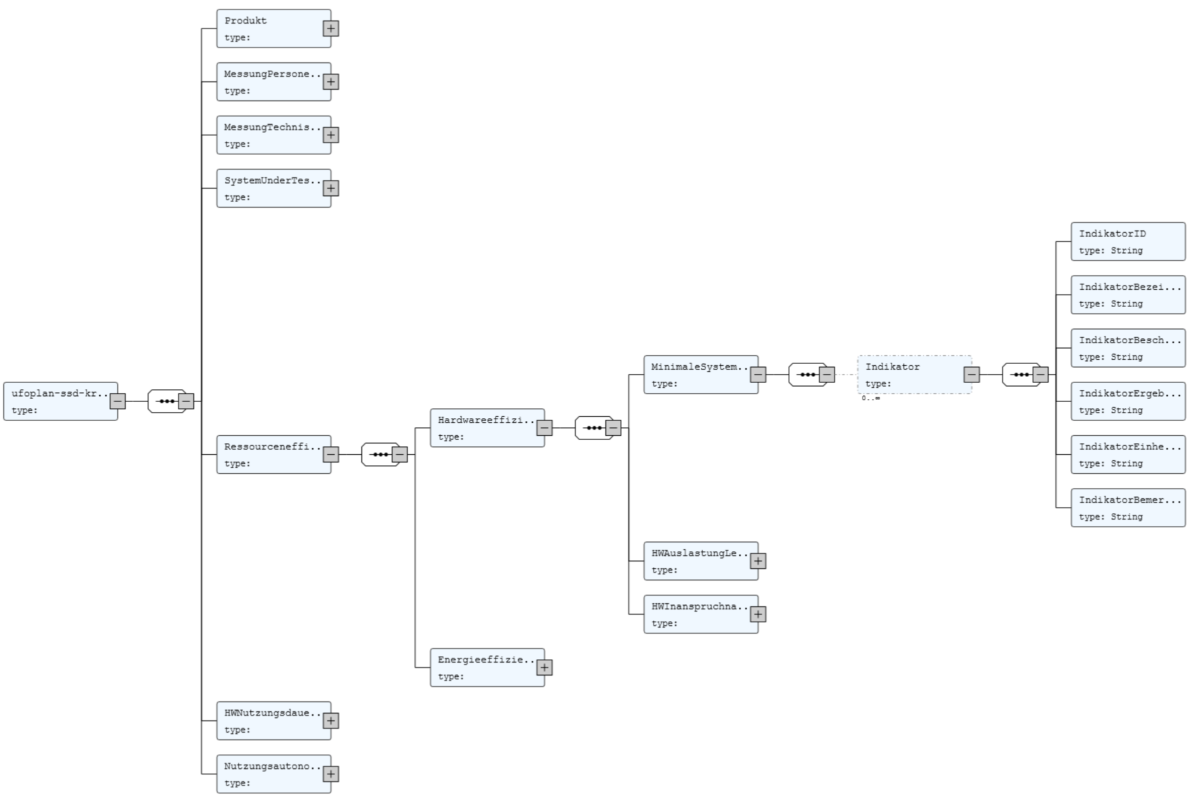
Die folgenden Anforderungen wurden an die zu entwickelnde XML-Datei gestellt: Die XML-Datei

- beinhaltet Indikatorenwerte (Messwerte und qualitative Bewertungen) der jeweils untersuchten Software und
- kann sowohl als Datenaustauschformat als auch als Produktinformation genutzt werden.

Basierend auf diesen Anforderungen und dem reduzierten Kriterienkatalog (siehe Tabelle 15) wurde ein XML-Schema (siehe Abbildung 29 und Anhang 5) entwickelt, das:

- die jeweiligen Indikatorenwerte,
- Informationen über die zu bewertende Software,
- Informationen über die Bewertungen durchführende Person inklusive ihrer Institution,
- Information über die der Energie- und Ressourcenmessung zu Grunde liegenden Infrastruktur (bspw. Messgerät) und dem genutzten SUT umfasst.

Abbildung 29: Visualisierung der Struktur des XML-Schema „ssd-info“



Quelle: Eigene Darstellung, Hochschule Trier

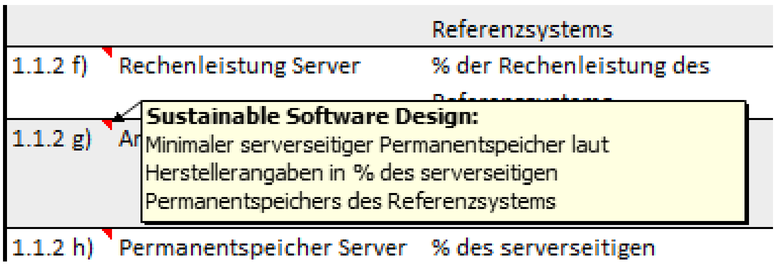
Dieses XML-Schema (mit dem Namen „ssd-info.xsd“) wurde dann in Excel eingebunden, um ein Erfassungstool anzubieten, das über weit verbreitete Office-Programme genutzt werden kann. Zur besseren Nutzerführung wurde das entstandene Excel-Formular aufbereitet und mit Hinweisen zur Nutzung, den Kriterien und Indikatoren, sowie zum zugrunde liegenden Projekt ergänzt.

Abbildung 30: Screenshot des Erfassungstools: Hinweise zum zugrunde liegenden Projekt und zur Bearbeitung, zu Erfassungsmöglichkeit von Informationen zum Produkt sowie zur Messung.

Quelle: Eigene Darstellung, Hochschule Trier

Über Kommentarfelder erhält der Nutzer des Erfassungstools Erläuterungen zu den zu erfassenden Indikatoren (siehe Operationalisierung Anhang 3 und Abbildung 31).

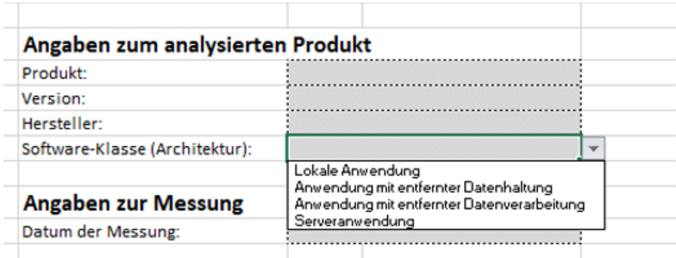
Abbildung 31: Screenshot des Erfassungstools: Informationen zu den zu erfassenden Indikatoren werden über einen Kommentar angezeigt.



Quelle: Eigene Darstellung, Hochschule Trier

Um sicher zu stellen, dass die Bewertung der Indikatoren einheitlich erfasst werden und die Ergebnisse vergleichbar sind, werden für einige Indikatoren Werte vorgegeben, die über eine Dropdown-Liste ausgewählt werden können (siehe Abbildung 32). Dies ist beispielsweise der Fall, wenn Indikatoren nicht erhoben werden können, da sie sich auf Server-Produkte beziehen und eine lokale Anwendung analysiert wird (Ergebnis: „entfällt, da lokale Anwendung“). In Fällen, in denen zur Erfassung des Indikators eine Frage zu beantworten ist (Beispiel: Indikator 3.1.3.a) „Reagiert der Hersteller zeitnah auf das Bekanntwerden von Sicherheitslücken?“ mögliche Ergebnisse: „ja / nein / teils“).

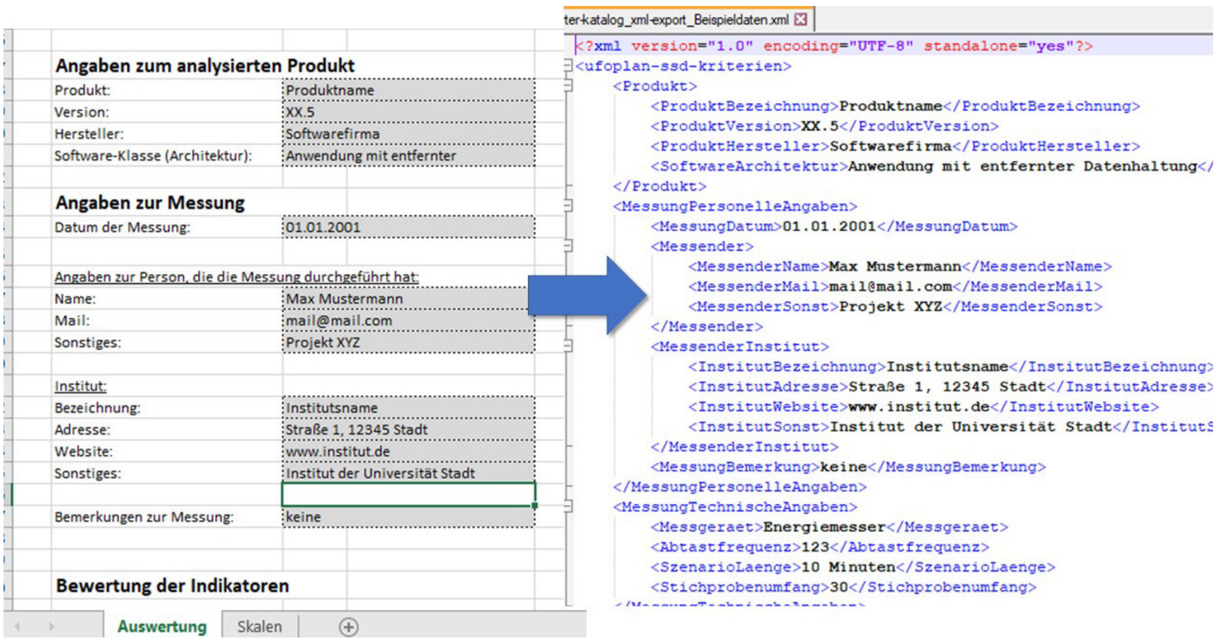
Abbildung 32: Screenshot des Erfassungstools: Auswahlmöglichkeiten bei vorgegebenen Ergebniswerten



Quelle: Eigene Darstellung, Hochschule Trier

Nachdem das Erfassungstool als Excel-Formular entsprechend erstellt und aufbereitet war, wurde es mit Hilfe vorhandener Mess- und Erfassungsdaten erfolgreich getestet.

Abbildung 33: Export der Excel-Datei in eine XML-Datei basierend auf dem XML-Schema "ssd-info"



Quelle: Eigene Darstellung, Hochschule Trier

Das entwickelte Erfassungstool zur Erhebung der Bewertungskriterien von Software ermöglicht folgende Nutzungen:

- Erfassung von Bewertungsergebnissen in EXCEL und Export in XML,
- Import von vorhandenen XML-Daten in EXCEL zur Änderung von Bewertungsergebnissen.

Alternativ zu Microsoft Office Excel ist auch eine Nutzung mit LibreOffice Calc möglich.

7 Schlussfolgerungen und weiterer Forschungsbedarf

Schlussfolgerungen

Das Forschungsprojekt zur „Entwicklung und Anwendung von Bewertungsgrundlagen für ressourceneffiziente Software“ hat die methodischen Grundlagen entwickelt, Softwareprodukte miteinander zu vergleichen und Effizienzanforderungen an diese zu stellen. Software hat einen messbaren Einfluss auf den Energieverbrauch und kann durch steigende Hardware-Inanspruchnahme zu einer vorzeitigen Alterung (Obsoleszenz) von Hardware beitragen. Die Wirkungszusammenhänge zwischen der Nutzung von Software und der Hardware-Inanspruchnahme sind jedoch sehr komplex. Mit dem Forschungsprojekt ist es gelungen, diese Komplexität auf vergleichsweise leicht zu erfassende Indikatoren zu reduzieren. Durch die Systematisierung der Software-Eigenschaften innerhalb des Kriterienkatalogs in *Ressourceneffizienz*, *potenzielle Hardware-Nutzungsdauer* und *Nutzungsautonomie* werden die wesentlichen Wirkungen von Software praktikabel strukturiert.

Bei der Überprüfung des Bewertungsinstruments an realen Softwareprodukten zeigt sich, dass sich diese bei gleicher Funktionalität stark in ihrem Ressourcenverbrauch unterscheiden. Für eine Aussage, welcher Ressourcenverbrauch für welche Funktionalität im Sinne eines Grenzwertes vertretbar ist, ist es jedoch zum derzeitigen Forschungsstand zu früh. Ebenso kann derzeit noch keine Aussage darüber getroffen werden, wie die Bewertungskriterien zu gewichten sind. D. h. welcher Indikator einen besonders hohen Einfluss auf den Ressourcenverbrauch aufweist.

Dennoch ist es bereits mit den Ergebnissen dieses Forschungsprojektes möglich, Anforderungen an ressourceneffiziente Softwareprodukte zu stellen. Die Indikatoren, die in den Handlungsempfehlungen für die Entwicklung eines Umweltzeichens (siehe Abschnitt 5) vorgeschlagen werden, sind dazu geeignet, die wichtigsten umweltbezogenen Software-Eigenschaften abzufragen. Diese Anforderungen können bereits jetzt bei der Entwicklung neuer Software berücksichtigt werden. Öffentliche Auftraggeber sollten im Rahmen der umweltverträglichen öffentlichen Beschaffung Mindestanforderungen an neu zu programmierende Software stellen und damit einen wachsenden Markt für ressourceneffiziente Software schaffen. Vor dem Hintergrund, dass ressourceneffiziente Software weniger Hardwarekapazitäten in Anspruch nimmt, den Energieverbrauch verringert und die Nutzungsdauer von Hardware verlängert, ist dies auch aus finanziellen Gesichtspunkten vorteilhaft.

Bislang wurde Software in den produktpolitischen Steuerungsinstrumenten, wie Ökodesign-Anforderungen, Energieeffizienzkennzeichnung, Produkthaftung und Umweltkennzeichnung ausgeblendet. Durch die im Rahmen dieses Forschungsvorhabens entwickelte Bewertungsmethodik lassen sich solche Instrumente perspektivisch auch für Software anwenden. Hierzu sind jedoch noch weiterer Forschungsbedarf und eine Weiterentwicklung der Bewertungsmethodik erforderlich.

Weiterer Forschungsbedarf

Kurzfristig besteht ein weiterer Forschungsbedarf um die Bewertungsmethodik zu Anforderungen für ein **Umweltzeichen „Blauer Engel“ für Softwareprodukte** weiter zu entwickeln. Diese Forschung beinhaltet insbesondere die Festlegung von Standardnutzungsszenarien für zu kennzeichnende Software und die weitere Überprüfung der Methodik in der Praxis. Die Entwicklung der Umweltzeichenanforderungen sollte im engen Austausch mit Software-Entwicklern erfolgen. Für die Entwicklung von Standardnutzungsszenarien, d.h. der standardisierten Abfolge von Aufgaben und Nutzerinteraktionen, die durch die Software während der Messungen ausgeführt werden, sollte ein definiertes Verfahren entwickelt werden, das für neue Softwaretypen angewendet werden kann.

Es besteht ein Zusammenhang zwischen dem Funktionsumfang eines Softwareprodukts und dessen Hardware-Inanspruchnahme. Bei wachsenden Möglichkeiten, die ein Softwareprodukt bietet, steigen in der Regel auch die Anforderungen an die Hardware und der Energieverbrauch. Die Bewertungsmethodik bietet derzeit noch keinen Ansatzpunkt, wie steigender Funktionsumfang (Nutzen) bei steigenden Hardware-Anforderungen (Aufwand) im Sinne einer Effizienzkennzahl (Verhältnis von Nutzen zu Aufwand) berücksichtigt werden kann. Als weiterer Forschungsbedarf wird daher die **Bewertung des Nutzens und der Funktionalität** von Software gesehen.

Die bestehende Methodik bewertet Software als fertig programmiertes Produkt. Die Phase der Softwareherstellung liegt derzeit noch außerhalb der Bewertung, obwohl auch hier deutliche Effizienzpotenziale erwartet werden. Bei einer Weiterentwicklung der Methodik sollte daher auch eine mögliche Ausweitung des Betrachtungsrahmens hin zur **Softwareentwicklung** überprüft werden. Dabei sollten zusätzlich auch Kriterien ergänzt werden, die auf die **sozialverträgliche Softwareherstellung** (z. B. Einhaltung von ILO-Kernarbeitsnormen[4]) abzielen.

Bei den Kriterien zum Energieverbrauch von Software lag der Schwerpunkt der vorliegenden Untersuchung auf lokaler Hardware-Inanspruchnahme (Desktop-Computer) und entfernten Hardware-Inanspruchnahme (Server). Der Energieverbrauch in den Übertragungsnetzen wurde dagegen auf Grundlage der übertragenen Datenmengen grob abgeschätzt. Die Abschätzung zeigt, dass die Datenübertragung mit einem erheblichen Energieverbrauch verbunden ist, der in der gleichen Größenordnung liegt, wie der im lokalen Computersystem verursachte Mehrverbrauch (vgl. Abbildung 21). Über die Hardware-Inanspruchnahme in Netzwerken macht die Berechnungsmethodik darüber hinaus keine Aussage. Vor dem Hintergrund der Relevanz von Netzwerken und den bestehenden Unsicherheiten sollte der **Energie- und Ressourcenverbrauch innerhalb von Netzwerken** weitergehend erforscht und quantifiziert werden.

Software wird nicht nur in Desktop-Computern verwendet, sondern selbstverständlich auch in vielen **anderen Geräten des täglichen Lebens** wie mobilen Endgeräten (z. B. Smartphones, Tablet-Computer), Haushaltsgeräten (Waschmaschinen, Fernseher, Stromzähler), Internet-of-Things, Fahrzeugen, Verkehrsleitsystemen usw. Perspektivisch sollten auch Software-Anwendungen für solche Geräte in den Untersuchungsrahmen aufgenommen werden. Diese Ausweitung beinhaltet auch die Festlegung neuer Referenzsysteme (z. B. Referenz-Smartphone) und Standardnutzungsszenarien.

Aktuell stellt die Nutzung der Blockchain-Technologie eine wichtige Softwareentwicklung dar, mit der Geschäftsprozesse dezentralisiert abgewickelt werden können. Auf dieser Technologie bauen vielfältige Geschäftsmodelle auf, wie die die Abrechnung von Dienstleistungen oder (als prominentes Beispiel) der Handel mit der elektronischen Währung Bitcoins. Die Blockchain-Technologie trägt zur verstärkten Nutzung von Datennetzen und weltweit verteilten Servern bei. Ihr Energie- und Ressourcenverbrauch kann derzeit nur unzureichend abgeschätzt werden. Weitere Forschungen sollten sich daher auch auf den **Energie- und Ressourcenverbrauch durch die Nutzung der Blockchain-Technologie** konzentrieren.

8 Quellenverzeichnis

Abdullah, Rusli; Abdullah, Salfarina; Din, Jamilah; Tee, Mcxin; others (2015): A Systematic Literature Review of Green Software Development in Collaborative Knowledge Management Environment. In: International Journal of Advanced Computer Technology (IJACT) 9, S. 136.

Abdullah, Rusli; Abdullah, Salfarina; Tee, Mcxin (2014): Web-based knowledge management model for managing and sharing green knowledge of software development in community of practice. In: Software Engineering Conference (MySEC), 2014 8th Malaysian. IEEE, S. 210–215.

Afgan, Naim Hamdia (2010): Sustainability paradigm: intelligent energy system. In: Sustainability 2 (12), S. 3812–3830.

Afzal, Shehla; Saleem, M. Faisal; Jan, Fahad; Ahmad, Mudassar (2013): A Review on Green Software Development in a Cloud Environment Regarding Software Development Life Cycle:(SDLC) Perspective. In: International Journal of Computer Trends and Technology (IJCTT) 4 (9), S. 3054–3058.

Agarwal, Shalabh; Nath, Asoke; Chowdhury, Dipayan (2012): Sustainable Approaches and Good Practices in Green Software Engineering. In: IJRRCS 3 (1), S. 1425–1428.

Ahmad, Ruzita; Baharom, Fauziah; Hussain, Azham (2014): A Systematic Literature Review on Sustainability Studies in Software Engineering. In: Proceedings of KMICe. Knowledge Management International Conference (KMICe) 2014. Malaysia, 12 – 15 August 2014.

Albertao, Felipe (2004): Sustainable Software Engineering. Carnegie Mellon University Silicon Valley. Online verfügbar unter http://www.scribd.com/doc/5507536/Sustainable-Software-Engineering#about, zuletzt aktualisiert am 04.09.2008, zuletzt geprüft am 01.06.2018.

Albertao, Felipe; Xiao, Jing; Tian, Chunhua; Lu, Yu; Zhang, Kun Qiu; Liu, Cheng (2010): Measuring the Sustainability Performance of Software Projects. In: IEEE Computer Society (Hg.): 2010 IEEE 7th International Conference on e-Business Engineering (ICEBE 2010), Shanghai, China. Technical Committee on Electronic Commerce (TCEC), S. 369–373. Online verfügbar unter http://doi.ieeecomputersociety.org/10.1109/ICEBE.2010.26, zuletzt geprüft am 01.06.2018.

Amsel, Nadine; Ibrahim, Zaid; Malik, Amir; Tomlinson, Bill (2011): Toward sustainable software engineering (NIER track). In: Proceedings of the 33rd International Conference on Software Engineering. ACM, S. 976–979.

Ardito, Luca; Morisio, Maurizio (2014): Green IT-Available data and guidelines for reducing energy consumption in IT systems. In: Sustainable Computing: Informatics and Systems 4 (1), S. 24‐32.

Berkhout, Frans; Hertin, Julia (2001): Impacts of Information and Communication Technologies on Environmental Sustainability: speculations and evidence. Report to the OECD. Hg. v. Organisation for Economic Co-operation and Development OECD. Brighton. Online verfügbar unter http://www.oecd.org/dataoecd/4/6/1897156.pdf, zuletzt geprüft am 01.06.2018.

Bouwers, Eric; van Deursen, Arie; Visser, Joost (2013): Evaluating usefulness of software metrics: an industrial experience report. In: Proceedings of the 2013 International Conference on Software Engineering. IEEE Press, S. 921–930.

Bozzelli, Paolo; Gu, Qing; Lago, Patricia (2013): A systematic literature review on green software metrics. Technical Report: VU University Amsterdam. http://www.sis.uta.fi/~pt/TIEA5_Thesis_Course/Session_10_2013_02_18/SLR_GreenMetrics.pdf, zuletzt geprüft am 01.06.2018.

Calero, C.; Bertoa, M.F; Angeles Moraga, M. (2013a): A systematic literature review for software sustainability measures. In: Green and Sustainable Software (GREENS), 2013 2nd International Workshop on, S. 46–53.

Calero, Coral; Bertoa, Manuel F.; Moraga, Maria Ángeles (2013b): Sustainability and Quality: Icing on the Cake. In: RE4SuSy@RE. Citeseer. Online verfügbar unter http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.415.6075&rep=rep1&type=pdf, zuletzt geprüft am 01.06.2018.

Calero, Coral; Moraga, M.; Bertoa, Manuel F. (2013c): Towards a software product sustainability model. In: arXiv preprint arXiv:1309.1640.

Calero, Coral; Moraga, Maria Ángeles; Bertoa, Manuel F.; Duboc, Leticia (2015): Green Software and Software Quality. In: Coral Calero und Mario Piattini (Hg.): Green in Software Engineering: Springer, S. 231–260.

Capra, E.; Francalanci, C.; Slaughter, S. A. (2012): Measuring Application Software Energy Efficiency. In: IT Professional, S. 54–61.

Capra, Eugenio; Francalanci, Chiara; Slaughter, Sandra A. (2011): Is software green? Application development environments and energy efficiency in open source applications. In: Information and Software Technology 54, S. 60–71. Online verfügbar unter http://ac.els-cdn.com/S0950584911001777/1-s2.0-S0950584911001777-main.pdf?_tid=a288bc86-f725-11e1-960e-00000aacb362&acdnat=1346827861_377674c8edd54a640c2d57df2bd4951b zuletzt geprüft am 01.06.2018.

Dick, Markus; Kern, Eva; Drangmeister, Jakob; Naumann, Stefan; Johann, Timo (2011): *Measurement and Rating of Software-induced Energy Consumption of Desktop PCs and Servers*. In: Pillmann, Werner; Schade, Sven; Smits, Paul (eds.): Innovations in Sharing Environmental Observation and Information. Proceedings of the 25th EnviroInfo Conference "Environmental Informatics" October 5-7, 2011, Ispra, (VA) Italy. Part 1 and 2, 2011, pp. 290-299.

Dick, Markus; Naumann, Stefan (2010): Enhancing Software Engineering Processes towards Sustainable Software Product Design. In: Klaus Greve und Armin B. Cremers (Hg.): EnviroInfo 2010: Integration of Environmental Information in Europe. Proceedings of the 24th International Conference on Informatics for Environmental Protection, October 6 - 8, 2010, Cologne/Bonn, Germany. Aachen: Shaker, S. 706–715.

Dirlewanger, Werner (2006): *Measurement and Rating of Computer Systems Performance and of Software Efficiency: An Introduction to the ISO /IEC 14756 Method and a Guide to its Application*. Kassel University Press (7. Dezember 2006)

EPA ENERGY STAR (2014): ENERGY STAR Program Requirements Product Specification for Computers: Eligibility Criteria, Version 6.1. Environmental Protection Agency. Online verfügbar unter http://www.energystar.gov/sites/default/files/specs//Version%206%201%20Computers%20Final%20Program%20Requirements.pdf, zuletzt geprüft am 01.06.2018.

EPA Office of Air and Radiation, Climate Protection Partnerships Division (2015): National Awareness of ENERGY STAR for 2014. Analysis of CEE Household Survey. Hg. v. U.S. Environmental Protection Agency. Online verfügbar unter https://www.energystar.gov/sites/default/files/asset/document/National_Awareness_of_ENERGY_STAR_2014_v6_508_1.pdf, zuletzt geprüft am 01.06.2018.

Erdmann, Lorenz; Hilty, Lorenz M.; Goodman, James; Arnfalk, Peter (2004): The Future Impact of ICTs on Environmental Sustainability. Technical Report EUR 21384 EN. Hg. v. Carlos Rodríguez Casal, Christine Van Wunnik, Luis Delgado Sancho, Jean Claude Burgelman und Paul Desruelle. European Commission; Joint Research Centre; IPTS - Institute for Prospective Technological Studies. Seville (Technical Report Series, EUR 21384 EN). Online verfügbar unter http://ftp.jrc.es/EURdoc/eur21384en.pdf, zuletzt geprüft am 01.06.2018.

Europäische Union (2011b): Beschluss der Kommission vom 9. Juni 2011 zur Festlegung der Umweltkriterien für die Vergabe des EU-Umweltzeichens für Tischcomputer. (Bekannt gegeben unter Aktenzeichen K(2011) 3737)Text von Bedeutung für den EWR. Online verfügbar unter https://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:02011D0337-20161010.

Europäische Union (Hg.) (2011a): Beschluss der Kommission vom 6. Juni 2011 zur Festlegung der Umweltkriterien für die Vergabe des EU-Umweltzeichens für Notebooks. (Bekannt gegeben unter Aktenzeichen K(2011) 3736)Text von Bedeutung für den EWR. Online verfügbar unter https://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:32011D0330.

Finkbeiner, Matthias; Schau, Erwin M.; Lehmann, Annekatrin; Traverso, Marzia (2010): Towards life cycle sustainability assessment. In: Sustainability 2 (10), S. 3309–3322. Online verfügbar unter http://www.mdpi.com/2071-1050/2/10/3309/pdf, zuletzt geprüft am 01.06.2018.

Fujitsu Technology Solutions (Hg.) (2010): Green Label-Kategorien bei Fujitsu Technology Solutions. White Paper.

Fujitsu Technology Solutions (Hg.) (2012): Green Label Levels at Fujitsu Technology Solutions. White Paper.

GeSI, Global e-Sustainability Initiative; The Climate Group (2008): SMART 2020: Enabling the low carbon economy in the information age. Online verfügbar unter https://www.theclimategroup.org/sites/default/files/archive/files/Smart2020Report.pdf zuletzt geprüft am 01.06.2018.

Gröger, Jens; Köhn, Marina; Albers, Erik; Löhr, Patrik; Lohmann, Wolfgang; Naumann, Stefan (2015): Nachhaltige Software. Dokumentation des Fachgesprächs „Nachhaltige Software" am 28.11.2014. Hg. v. Umweltbundesamt. Öko-Institut e.V. Dessau-Roßlau. Online verfügbar unter http://www.umweltbundesamt.de/publikationen/nachhaltige-software, zuletzt geprüft am 01.06.2018.

Gröger, Jens; Quack, Dietlinde; Grießhammer, Rainer; Gattermann, Marah (2013): TOP 100-Umweltzeichen für klimarelevante Produkte: Freiburg. Online verfügbar unter http://www.ecodialog.de/oekodoc/1739/2013-433-de.pdf, zuletzt geprüft am 01.06.2018.

Held, Alexandra (2010): Entwicklung und Operationalisierung von Kriterien zur Bewertung der Nachhaltigkeit von Softwareprodukten. Abschlussarbeit zur Erlangung des akademischen Grades Master of Science eingereicht am Umwelt-Campus Birkenfeld. Masterarbeit. Fachhochschule Trier, Standort Umwelt-Campus Birkenfeld, Hoppstädten-Weiersbach. ISS Institut für Softwaresysteme in Wirtschaft, Umwelt und Verwaltung.

Hilty, Lorenz; Lohmann, Wolfgang; Behrendt, Siegfried; Evers-Wölk, Michaela; Fichter, Klaus; Hintemann, Ralph (2015): Grüne Software. Schlussbericht zum Vorhaben: Ermittlung und Erschließung von Umweltschutzpotenzialen der Informations- und Kommunikationstechnik (Green IT). Studie im Auftrag des Umweltbundesamtes, Berlin, Förderkennzeichen 3710 95 302/3 (im Druck).

Horne, Ralph E. (2009): Limits to labels: The role of eco-labels in the assessment of product sustainability and routes to sustainable consumption. In: International Journal of Consumer Studies 33 (2), S. 175–182. Online verfügbar https://researchbank.rmit.edu.au/view/rmit:3981, zuletzt geprüft am 01.06.2018.

International Standard ISO/IEC 25010:2011, 01.03.2011: Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models.

Kern, Eva; Dick, Markus; Naumann, Stefan; Guldner, Achim; Johann, Timo (2013): *Green Software and Green Software Engineering – Definitions, Measurements, and Quality Aspects*. In: Hilty, Lorenz M.; Aebischer, Bernard; Andersson, Göran; Lohmann, Wolfgang (Eds.): ICT4S ICT for Sustainability. Proceedings of the First International Conference on Information and Communication Technologies for Sustainability, ETH Zurich, February 14-16, 2013. ETH Zurich, University of Zurich and Empa, Swiss Federal Laboratories for Materials Science and Technology (2013). http://www.zora.uzh.ch/id/eprint/84888/

Kern, Eva; Dick, Markus; Naumann, Stefan; Guldner, Achim; Johann, Timo (2013): Green Software and Green Software Engineering – Definitions, Measurements, and Quality Aspects. In: Lorenz M. Hilty, Bernard Aebischer, Göran Andersson und Wolfgang Lohmann (Hg.): ICT4S ICT for Sustainability. Proceedings of the First International Conference on Information and Communication Technologies for Sustainability, ETH Zurich, February 14-16, 2013. Zürich: ETH Zurich, University of Zurich and Empa, Swiss Federal Laboratories for Materials Science and Technology, S. 87–94. Online verfügbar unter http://e-collection.library.ethz.ch/eserv/eth:6558/eth-6558-01.pdf, zuletzt geprüft am 01.06.2018.

Koçak, Sedef Akınlı; Alptekin, Gülfem Işıklar; Bener, Ayşe Başar (2014): Evaluation of Software Product Quality Attributes and Environmental Attributes using ANP Decision Framework. In: Proceedings of the Third International Workshop on Requirement Engineering for Sustainable Systems (pp. pp. 37-44). Karlskrona: Central Europe Workshop Proceedings.

Koçak, Sedef Akinli; Calienes, Giovanna Gonzales; Alptekin, Gülfem Isiklar; Bener, Ayse Basar (2013): Requirements Prioritization Framework for Developing Green and Sustainable Software using ANP-based Decision Making. In: EnviroInfo, S. 327–335.

Lago, Patricia; Jansen, Toon; Jansen, Marten (2010): The service greenery-integrating sustainability in service oriented software. In: International Workshop on Software Research and Climate Change (WSRCC), co-located with ICSE, Bd. 2.

Lago, Patricia; Koçak, Sedef Akinli; Crnkovic, Ivica; Penzenstadler, Birgit (2015): Framing sustainability as a property of software quality. In: Communications of the ACM 58 (10), S. 70–78.

Lami, Giuseppe; Fabbrini, Fabrizio; Fusani Mario (2012): Software Sustainability from a Process-Centric Perspective. In: D. Winkler, R.V O'Connor und R. Messnarz (Hg.): EuroSPI 2012, CCIS 301: Springer, S. 97–108.

Lassmann, W. (2006). Wirtschaftsinformatik - Nachschlagewerk für Studium und Praxis. Wiesbaden. 613. doi:10.1007/978-3-8349-9152-2

Mazijn, B.; Doom, R.; Peeters, H.; Vanhoutte, G.; Spillemaeckers, S.; Taverniers, L. et al. (2004): Ecological, Social and Economic Aspects of Integrated Product Policy. Integrated Product Assessment and the Development of the Label `Sustainable Development´ for Products. CP/20. SPSD II - Part I - Sustainable production and consumption patterns. Online verfügbar unter http://www.bernardmazijn.be/fileadmin/pdf/sd-label_products_bernardmazijn.pdf, zuletzt geprüft am 01.06.2018.

Naumann, Stefan; Dick, Markus; Kern, Eva; Johann, Timo (2011): The GREENSOFT Model: A Reference Model for Green and Sustainable Software and its Engineering. In: SUSCOM 1 (4), S. 294–304. DOI: 10.1016/j.suscom.2011.06.004.

Penzenstadler, Birgit; Mahaux, Martin; Salinesi, Camille (2013): RE4SuSy: Requirements Engineering for Sustainable Systems. In: Journal of Systems and Software.

Prakash, Siddharth; Manhart, Andreas; Stratmann, Britta; Reintjes, Norbert (2008): Environmental product indicators and benchmarks in the context of environmental labels and declarations. Öko-Institut e.V.; Ökopol GmbH.

RAL gGmbH (Hg.) (2014) Vergabegrundlage für Umweltzeichen RAL-UZ 78a, 2014-11: Vergabegrundlage für Umweltzeichen - Computer.

RAL gGmbH (Hg.) (2014): Vergabegrundlage für Umweltzeichen RAL-UZ 100, 2014-06: Vergabegrundlage für Car-Sharing, online verfügbar unter http://www.nachhaltigkeitskriterien.de/download/139.pdf, zuletzt geprüft am 01.06.2018.

RAL-UZ 161, 2012-07: Basic Criteria for Award of the Environmental Label Energy-Conscious Data Centers, zuletzt geprüft am 28.03.2014.

Schipper, Irene (2015): TCO Certified Smartphones versus Fairphone. A comparison of sustainability criteria. Hg. v. GoodElectronics Network Südwind. Stichting Onderzoek Multinationale Ondernemingen (SOMO), Centre for Research on Multinational Cooperations, Netherlands. Amsterdam. Online verfügbar unter https://www.somo.nl/wp-content/uploads/2015/07/TCO-Certified-Smartphones-versus-Fairphone.pdf, zuletzt geprüft am 01.06.2018

Schmidt, Benno (2014): Strategien für eine integrativ-nachhaltige Software-Entwicklung. Hochschule Bochum, Fachbereich Geodäsie. Bochum (14-02). Online verfügbar unter. http://www.hochschule-bochum.de/fileadmin/media/fb_v/prof_schmidt/Berichte/SchmidtWytzisk_UINW2014.pdf, zuletzt geprüft am 01.06.2018

Schmidt, Benno; Wytzisk, Andreas; Plödereder, In E.; Grunske, L.; Schneider, E.; Ull, D.; others (2014): Software Engineering und Integrative Nachhaltigkeit. In: Erhard Plödereder, Lars Grunske, Eric Schneider und Dominic Ull (Hg.): INFORMATIK 2014. Big Data – Komplexität meistern. – Proceedings. 2. Workshop "Umweltinformatik zwischen Nachhaltigkeit und Wandel (UINW) / Environmental Informatics between Sustainability and Change"), 26.09.2014, im Rahmen der INFORMATIK 2014, 22.-26.09.2014 in Stuttgart. P-232: Lecture Notes in Informatics (LNI), S. 1935–1945.

Scholl, Gerd; Simshäuser, Ulla (2002): Machbarkeitsuntersuchung für Umweltzeichen-Analyse der Möglichkeiten zur Akzeptanzerhöhung des Umweltzeichens" Blauer Engel" für Haushaltsgroßgeräte (" Weiße Ware") bei potenziellen Zeichennehmern: Umweltbundesamt. Online verfügbar unter http://www.umweltbundesamt.de/sites/default/files/medien/publikation/short/k2169.pdf.

Stieß, Immanuel; Birzle-Harder, Barbara (2013): Der Blaue Engel-ein Klassiker mit Potenzial: eine empirische Studie zu Verbraucherakzeptanz und Marktdurchdringung des Umweltzeichens.

Sundblad, Yngve; Lind, Torbjörn; Rudling, Jan (2002): IT product requirements and certification from the users' perspective. In: Proceedings of WWDU 2002 Conference, S. 280–282. Online verfügbar unter http://cid.nada.kth.se/pdf/CID-176.pdf.

Taina, Juha (2011): Good, Bad, and Beautiful Software - In Search of Green Software Quality Factors. In: CEPIS UPGRADE XII (4), S. 22–27. Online verfügbar unter http://www.cepis.org/upgrade/media/taina_2011_41.pdf, zuletzt geprüft am 01.06.2018.

TCO Development (Hg.) (2012): TCO Certified Notebooks 4.0. Online verfügbar unter http://tcodevelopment.com/files/2012/12/TCO-Certified-Notebooks-4.0_eco-templates.pdf, zuletzt geprüft am 01.06.2018.

Teufel, J.; Rubik, F.; Scholl, G.; Stratmann, B.; Graulich, K.; Manhart, A. (2009): Untersuchung zur möglichen Ausgestaltung und Marktimplementierung eines Nachhaltigkeitslabels zur Verbraucherinformation. In: Project report of the Öko-Institut e. V. in cooperation with the Institut für ökologische Wirtschaftsforschung (IÖW) GmbH. Freiburg: Öko-Institut e. V. Online verfügbar unter http://download.ble.de/08HS031.pdf.

Umweltbundesamt (Hg.) (2013): Ökodesign-Richtlinie <Computer und Computerserver>. Online verfügbar unter https://www.umweltbundesamt.de/sites/default/files/medien/376/dokumente/datenblatt_oekodesign-richtlinie_computer_und_computerserver.pdf, zuletzt geprüft am 01.06.2018.

Walch, Isabelle (2015): Standard ECMA-370. TED - The ECO Declaration. Hg. v. ecma INTERNATIONAL.

Waller, Lars (2015): TCO Certified Desktops 5.0 - Critera Document and Certification Process. TCO Development. Online verfügbar unter http://tcodevelopment.com/files/2015/12/TCO-Certified-Desktops-5.0.pdf, zuletzt geprüft am 01.06.2018.

Warschun, Mirko; Rühle, Jens (2008): Zwischen Öko-Labels, grüner Logistik und fairem Handel Lebensmitteleinzelhandel auf der Suche nach Wegen zur Nachhaltigkeit, Managementberatung A.T. Kearney und der Universität zu Köln - Lehrstuhl für Unternehmensführung und Logistik.

Anhang 1 Kriterienkatalog für nachhaltige Software

Stand: 31.05.2017

1 Ressourceneffizienz

In welchem Ausmaß werden bei gegebener Funktionserfüllung Hardwarekapazitäten und damit indirekt natürliche Ressourcen beansprucht?

Dieses Hauptkriterium geht von der Vorstellung aus, dass die Erfüllung einer gegebenen Funktionalität durch ein Softwareprodukt mit unterschiedlicher Beanspruchung von Hardwarekapazitäten erfolgen kann, was indirekt zu einem unterschiedlichen Ausmaß in der Beanspruchung von natürlichen Ressourcen führt, die für die Bereitstellung und den Betrieb der Hardware benötigt werden.

Das angestrebte Ideal ist ein Softwareprodukt, das eine gegebene Funktionalität mit einem minimalen Ressourcenaufwand erbringt, also die Ressourceneffizienz (siehe Glossar) maximiert. Die Funktionalität wird durch Standardnutzungsszenarien (siehe Glossar) definiert. Als Näherung zur Abschätzung der beanspruchten natürlichen Ressourcen dienen die bereitzuhaltenden und die tatsächlich genutzten Hardwarekapazitäten sowie die dabei verbrauchte Energie.

1.1 Hardwareeffizienz

Welche Hardwarekapazitäten müssen für den Betrieb des Softwareprodukts bereitgehalten werden und wie werden sie während des Betriebs ausgelastet?

Die Hardwarekapazitäten werden in Prozent der entsprechenden Kapazität eines Referenzsystems[21] gemessen. Sie können nach zwei Dimensionen differenziert werden (Tabelle 18). Auf der einen Dimension werden sie nach lokalen, Netzwerk-, und entfernten Ressourcen differenziert. Hier unterscheiden wir zusätzlich zwischen den empfohlenen (1.1.1), minimalen (1.1.2), im Leerlauf benötigten (1.1.3) und bei einem Standardnutzungsszenario benötigten (1.1.4) Kapazitäten. Auf der anderen Dimension differenzieren wir nach dem Typ der Hardware-Kapazität: Rechenleistung, Arbeitsspeicher, Permanentspeicher, Bandbreite, Displayauflösung. Die Matrix ist offen für die Erweiterung um zusätzliche Spalten, wenn in Zukunft neue Kategorien von Hardware relevant werden sollten.

[21] Zur Anwendung des Kriteriensystems muss periodisch ein Referenzsystem auf dem Stand der technischen Entwicklung bestimmt werden. Dieses dient zur Normierung von Indikatoren.

Tabelle 18: Differenzierung der Hardwarekapazitäten in zwei Dimensionen

		Rechenleistung	Arbeitsspeicher	Permanentspeicher	Bandbreite	Displayauflösung
lokal	empfohlen minimal Leerlauf Standardnutzung	1.1.1 a) 1.1.2 a) 1.1.3 a) 1.1.4 a)	1.1.1 b) 1.1.2 b) 1.1.3 b) 1.1.4 b)	1.1.1 c) 1.1.2 c) 1.1.3 c) 1.1.4 c)	-	1.1.1 d) 1.1.2 d)
Netz	empfohlen minimal Leerlauf Standardnutzung	-	-	-	1.1.1 e) 1.1.2 e) 1.1.3 d) 1.1.4 d)	-
entfernt	empfohlen minimal Leerlauf Standardnutzung	1.1.1 f) 1.1.2 f) 1.1.3 e) 1.1.4 e)	1.1.1 g) 1.1.2 g) 1.1.3 f) 1.1.4 f)	1.1.1 h) 1.1.2 h) 1.1.3 g) 1.1.4 g)	-	-

Jede Zelle der in Tabelle 18 dargestellten Matrix zeigt das zugehörige Kriterium (z. B. 1.1.1) mit dem zugehörigen Indikator (z. B. a)) zur Operationalisierung des Kriteriums. Die Kriterien und Indikatoren werden in den folgenden Abschnitten erläutert, die entsprechend nummeriert sind. Nicht alle Kriterien 1.1.1 – 1.1.4 sind in allen Feldern der Matrix anwendbar, deshalb bleiben einige Zellen leer.

Die Kriterien 1.1.5 und 1.1.6 dienen ebenfalls zur Beurteilung der Hardwareeffizienz, können pauschal beurteilt werden und erfordern keine Differenzierung nach dieser Matrix. Sie sind deshalb in Tabelle 18 nicht erwähnt.

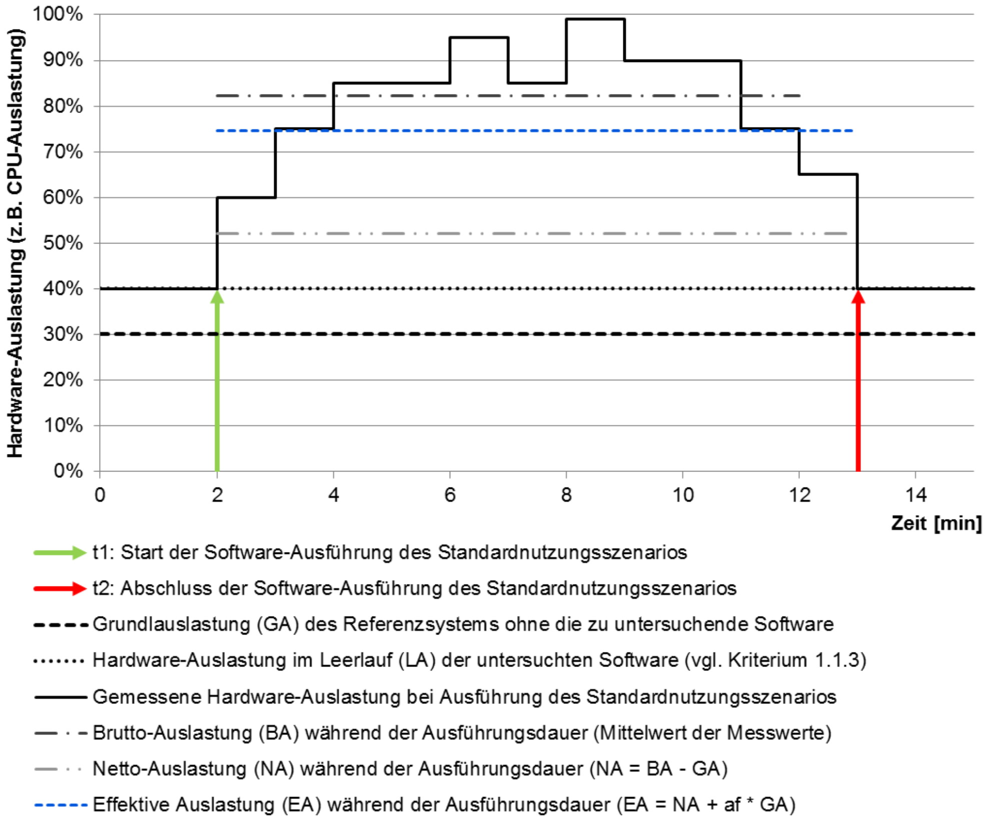
Grundsätzlich stellt sich für die spätere Aggregation von Kriterien das Problem einer Konkurrenz zwischen verschiedenen Hardwarekapazitäten (lokal vs. entfernt, Rechenleistung vs. Arbeitsspeicher, Rechenleistung zur Datenkompression vs. Bandbreite usw.). Könnte man die Hardwarekapazitäten in Form eines ökologischen Fußabdrucks bewerten, könnten sie mit dem Ergebnis dieser Bewertung gewichtet und aggregiert werden. Die Abschätzung des ökologischen Fußabdrucks ist nicht Teil der Arbeiten, die hier vorgestellt werden; wir verweisen den Leser auf existierende Lebenszyklus-Inventare von IKT-Hardware und elektrischer Energie als Grundlage für eine Aggregation.

Tabelle 19: Grundlegende Definitionen für die Messung der Kriterien 1.1.3 und 1.1.4

Bezeichner	Name	Definition	Anmerkung
VA_i	Vollauslastung	Obere Grenze der Auslastung i des Referenzsystems	Bei Prozessorleistungen beträgt die Vollauslastung 100%, bei Arbeitsspeicher die Summe der installierten RAM-Kapazitäten, beim Netzwerk die maximale Übertragungsgeschwindigkeit usw.
GA_i	Grundauslastung	Messwert der mittleren Auslastung i des Referenzsystems ohne die zu untersuchende Software.	
LA_i	Leerlauf-Auslastung	Messwert der mittleren Auslastung i des Referenzsystems im Leerlauf der untersuchten Software	Sie beinhaltet sowohl die Grundauslastung (GA_i) als auch die zusätzliche durch den Leerlaufbetrieb der Software verursachte Hardware-Auslastung.
NLA_i	Netto-Leerlauf-Auslastung	$NLA_i = LA_i - GA_i$	Sie beschreibt die durch den Leerlaufbetrieb der Software über die Grundauslastung hinausgehende beanspruchte Hardware-Auslastung
t	Ausführungsdauer	Laufzeit der untersuchten Software zur Ausführung des Standardnutzungsszenarios auf dem Referenzsystem	Sie beginnt mit dem Start des Standardnutzungsszenarios und endet mit der Erledigung aller dort vorgesehenen Aktionen, inklusive nachgelagerter Prozesse (z.B. Freigeben von Arbeitsspeicher, Löschung temporärer Dateien).
BA_i	Brutto-Auslastung	Mittlere Auslastung des Referenzsystems bei Ausführung eines Standardnutzungsszenarios über dessen Ausführungsdauer	Sie wird als zeitgewichteter Mittelwert der Messwerte über die Ausführungsdauer berechnet.
NA_i	Netto-Auslastung	$NA_i = BA_i – GA_i$	Sie bezeichnet die durch die Software verursachte mittlere Hardware-Auslastung nach Abzug der Grundauslastung des Referenzsystems.
AF_i	Allokationsfaktor	$AF_i = NA_i / (VA_i - GA_i)$ (Allokationsfaktor für die Ausführung des Standardnutzungsszenarios)	Verhältnis der durch die Software beanspruchten Netto-Auslastung zu der maximal zur Verfügung stehenden Auslastung. Dabei ist berücksichtigt, dass maximal nur die Differenz zwischen der Vollauslastung (VA) und der Grundauslastung (GA) zur Verfügung steht.
AFL_i	Leerlauf-Allokationsfaktor	$AFL_i = NLA_i / (VA_i - GA_i)$ (Allokationsfaktor für den Leerlauf der Software)	
EA_i	Effektive Auslastung	$EA_i = NA_i + AF_i * GA_i$	
ELA_i	Effektive Leerlauf-Auslastung	$ELA_i = NLA_i + AFL_i * GA_i$	Genutzt um die Indikatoren zur Berechnung der Hardwareansprüche von Kriterium 1.1.3 zu berechnen
H_i	Hardware-Inanspruchnahme	$H_i = EA_i * t$	Genutzt um die Indikatoren zur Berechnung der Hardwareansprüche von Kriterium 1.1.4 zu berechnen

Die nachfolgende Abbildung stellt den Messzyklus beispielhaft dar und illustriert die verschiedenen Auslastungen und deren Bezeichnung.

Abbildung 34: Beispielhafter Messzyklus zur Bestimmung der Hardware-Auslastung



Quelle: Eigene Darstellung, Öko-Institut

1.1.1 Empfohlene Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)

Welche Systemvoraussetzungen für den Betrieb des Softwareprodukts werden vom Hersteller empfohlen? Wenn andere Softwareprodukte zur Nutzung des betrachteten Produktes vorausgesetzt werden, sind deren empfohlene Systemvoraussetzungen zusätzlich zu berücksichtigen.

Indikatoren:

a) Empfohlene lokale Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems (Produkt aus Taktfrequenz, Anzahl Kerne, Busbreite)
b) Empfohlener lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems
c) Empfohlener lokaler Permanentspeicher laut Herstellerangaben in % des Permanentspeichers des Referenzsystems
d) Empfohlene Displayauflösung laut Herstellerangaben in % der Displayauflösung des Referenzsystems
e) Empfohlene Bandbreite für Netzzugang laut Herstellerangaben in % der Bandbreite des Referenzsystems
f) Empfohlene serverseitige Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems

g) Empfohlener serverseitiger Arbeitsspeicher laut Herstellerangaben in % des serverseitigen Arbeitsspeichers des Referenzsystems
h) Empfohlener serverseitiger Permanentspeicher laut Herstellerangaben in % des serverseitigen Permanentspeichers des Referenzsystems

1.1.2 *Minimale* Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)

Welche Systemvoraussetzungen sind für den Betrieb des Softwareprodukts minimal notwendig?

Indikatoren:

a) Minimale lokale Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems
b) Minimaler lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems
c) Minimaler lokaler Permanentspeicher laut Herstellerangaben in % des Permanentspeichers des Referenzsystems
d) Minimale Displayauflösung laut Herstellerangaben in % der Displayauflösung des Referenzsystems
e) Minimale Bandbreite für Netzzugang laut Herstellerangaben in % der Bandbreite des Referenzsystems
f) Minimale serverseitige Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems
g) Minimaler serverseitiger Arbeitsspeicher laut Herstellerangaben in % des serverseitigen Arbeitsspeichers des Referenzsystems
h) Minimaler serverseitiger Permanentspeicher laut Herstellerangaben in % des serverseitigen Permanentspeichers des Referenzsystems

1.1.3 Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration

Wie hoch ist die mittlere Auslastung der bereitgestellten Hardwarekapazitäten durch das Softwareprodukt, wenn sich dieses im Leerlauf befindet?

Indikatoren:

a) Messung der mittleren Prozessorauslastung im Leerlauf unter Standardkonfiguration (Differenz zum Grundverbrauch der Standardkonfiguration ohne das Softwareprodukt im gleichen Zeitraum)
b) Messung der mittleren Arbeitsspeicherbelegung im Leerlauf unter Standardkonfiguration
c) Messung der mittleren Permanentspeicherbelegung im Leerlauf unter Standardkonfiguration
d) Messung der mittleren beanspruchten Bandbreite für Netzzugang im Leerlauf unter Standardkonfiguration
e) Messung der mittleren serverseitigen Prozessorauslastung im Leerlauf unter Standardkonfiguration
f) Messung der mittleren serverseitigen Arbeitsspeicherbelegung im Leerlauf unter Standardkonfiguration
g) Messung der mittleren serverseitigen Permanentspeicherbelegung im Leerlauf unter Standardkonfiguration

Die mittleren Prozessorauslastungen (Indikatoren a) und e)) sowie die mittleren Arbeitsspeicherbelegungen (Indikatoren b) und f)) werden als Effektive Leerlauf-Auslastungen (ELA) berechnet (siehe Tabelle 19). Sie beinhalten neben der Netto-Leerlauf-Auslastung (NLA) zusätzlich noch einen Anteil an der Grundauslastung (GA) gemäß den Festlegungen in Abschnitt 1.1.

1.1.4 Hardware-Inanspruchnahme bei normaler Nutzung unter der Annahme einer Standardkonfiguration und eines Standardnutzungsszenarios

Wie hoch ist die Inanspruchnahme der bereitgestellten Hardwarekapazitäten beim Betrieb des Softwareprodukts?

Als Hardware-Inanspruchnahme wird hier das Integral der Hardware-Auslastung über die Ausführungsdauer eines Standardnutzungsszenarios verstanden. Die Maßeinheiten für die Hardware-Inanspruchnahme sind Einheiten für Arbeitsleistung, wie %*s (Prozessor- und Arbeitsspeicherarbeit), MByte/s*s = MByte (Permanentspeicherarbeit, Lesen und Schreiben) und MBit/s*s = MBit (im Netzwerk übertragene Datenmenge). Anders als bei den vorangehenden Kriterien 1.1.1 – 1.1.3 wird bei der Hardware-Inanspruchnahme also auch die Ausführungsdauer berücksichtigt. Zur Erläuterung: Wenn ein Programm A doppelt so viel Prozessorleistung, Arbeitsspeicher oder Bandbreite beansprucht wie Programm B, um ein gegebenes Standardnutzungsszenario zu erledigen, aber dafür den Prozessor, Speicher oder die Bandbreite nach der Hälfte der von B benötigten Zeit wieder freigibt, so ist die Hardware-Inanspruchnahme beider Programme gleich hoch.

Die Hardware-Inanspruchnahme wird als Produkt aus mittlerer effektiver Hardware-Auslastung und der zur Ausführung des Standardnutzungsszenarios benötigten Zeit berechnet. Die Ausführungsdauer ist dabei für alle Hardwarekapazitäten gleich hoch.

Indikatoren:

a) Messung der Prozessorarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration
b) Messung der Arbeitsspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration
c) Messung der Permanentspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration
d) Messung der übertragenen Datenmenge für Netzzugang bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration
e) Messung der serverseitigen Prozessorarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration
f) Messung der serverseitigen Arbeitsspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration
g) Messung der serverseitigen Permanentspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration

Die Hardware-Inanspruchnahme (H) für die Prozessorarbeit (Indikatoren a) und e)) sowie die Arbeitsspeicherarbeit (Indikatoren b) und f)) werden berechnet wie in Tabelle 19 angegeben.

1.1.5 Sparsame Hardwarenutzung durch Anpassbarkeit und Unterstützung der Nutzenden bei der Anpassung des Softwareprodukts

Beansprucht das Softwareprodukt nur jene Hardwarekapazitäten, die für die Ausführung der von den jeweiligen Nutzenden nachgefragten Funktionen benötigt werden? Werden die Nutzenden durch die Software ausreichend dabei unterstützt, entsprechende Anpassungen vorzunehmen?[22]

Indikatoren:

a) Geschieht die Minimierung der beanspruchten Kapazitäten automatisch und/oder gibt es bei der Installation der Software entsprechende Optionen? (Skala: ja/nein)
b) Falls die Nutzenden eine entsprechende Wahl treffen, ist die Entscheidung für oder gegen Installationsoptionen später jederzeit revidierbar (Skala: ja/nein)
c) Black-Box-Test: Können hardwareintensive Software-Module abgeschaltet werden? (Skala: können abgeschaltet werden/können nicht abgeschaltet werden)
d) Ist es ohne Nachteil möglich, vorübergehend oder dauerhaft nicht benötigte Peripheriegeräte abzuschalten bzw. gar nicht bereitzustellen? (Skala: Können vorübergehend und dauerhaft abgeschaltet werden/können nur vorübergehend abgeschaltet werden/können nicht abgeschaltet werden)
e) Werden nach der Installation die Dateien gelöscht, die nur zur Installation benötigt werden?

1.1.6 Online-Auslieferung

Kann das Softwareprodukt (inkl. aller Programme, Daten und Dokumentation einschließlich Handbüchern) ohne den Transport physischer Datenträger (inkl. Papier) oder anderer materieller Güter (inkl. Verpackung) erworben, installiert und betrieben werden?

Indikatoren:

a) Sind eine Online-Auslieferung und ein Online-Update der Software möglich?
b) Wird unterstützt, dass das Softwareprodukt und seine Updates in der anwendenden Organisation zentral abgelegt werden?

1.2 Energieeffizienz

Wie viel elektrische Energie verbraucht die Hardware bei Nutzung des Softwareprodukts zur Ausführung eines Standardnutzungsszenarios?

Der Verbrauch elektrischer Energie ist eine Konsequenz der Beanspruchung von Hardwarekapazitäten. Deren Bestimmung wurde unter 1.1.4 bereits behandelt. Deshalb soll die elektrische Leistungsaufnahme der Hardware im Rahmen der unter 1.1.4 beschriebenen Messungen jeweils mitgemessen werden, zumindest für die Summe der jeweils lokal, im Netz oder entfernt genutzten Hardwarekapazitäten.

Indikatoren:

a) Messung der zur Ausführung des Standardnutzungsszenarios unter Standardkonfiguration auf dem lokalen Gerät verbrauchten Energie

[22] keine Belegung von Kapazitäten durch vorübergehend oder dauerhaft nicht nachgefragte Funktionalität.

b) Schätzung der durch die Datenübertragung im Netz verbrauchten Energie aufgrund des bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration erzeugten Datenverkehrs (Verwendung einer aktuellen Schätzung für die Energieintensität des Netzes in kWh/GB basierend auf aktueller Fachliteratur, wenn nötig differenziert nach Zugangsnetzen)
c) Messung der durch die entfernte Speicherung und Verarbeitung in Servern verbrauchten Energie bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration (falls Messung nicht möglich, Schätzung mit Hilfe durchschnittlicher Faktoren für die Energieintensität von Rechenzentrumsdienstleistungen basierend auf aktueller Fachliteratur)

Die verbrauchte elektrische Energie ergibt sich durch Integration der elektrischen Leistungsaufnahme über die Ausführungszeit des Standardnutzungsszenarios. Abweichend von den Festlegungen, die für die Hardware-Auslastung (siehe Abschnitt 1.1.4) getroffen wurden, wird für die Messungen der elektrischen Energie (Indikatoren a) und c)) nur der Nettowert der elektrischen Leistungen ermittelt, also nur die Leistungswerte, die über den elektrischen Grundverbrauch des Referenzsystems hinausgehen. Diese abweichende Berechnung erfolgt sowohl aus Gründen der Praktikabilität (die Berechnung des Allokationsfaktors nach Tabelle 19 ist bei elektrischen Leistungswerten aufgrund fehlender Obergrenzen nicht praktikabel) als auch aus Gründen der Aussagekraft (die Unterschiede im Energieverbrauch von Software werden deutlicher, wenn die Grundauslastung nicht im Indikator enthalten ist).

1.3 Ressourcenmanagement

In welchem Ausmaß trägt das Softwareprodukt während seines Betriebs zu einem effizienten Management der von ihm beanspruchten Ressourcen bei?

Weil das Ausmaß der Nutzung eines gegebenen Softwareprodukts schwanken kann, trägt eine entsprechende adaptive und durch das Softwareprodukt unterstützte Nachfrage nach Hardwarekapazitäten zur Ressourcenschonung bei. Freigegebene Hardwarekapazitäten können potenziell durch andere Prozesse genutzt werden oder ihren Energieverbrauch reduzieren. Beides trägt indirekt zur Schonung natürlicher Ressourcen bei.

Dieses Kriterium bezieht sich im Gegensatz zu den Kriterien 1.1. und 1.2 auf die Anpassung des Bedarfs an Hardwarekapazitäten zur Laufzeit des Programms, insbesondere den Übergang in sparsamere Modi, abhängig von den momentanen Anforderungen der Nutzenden oder dem Angebot an Hardwarekapazitäten und Energie. Während die Ressourceneffizienz in den verschiedenen Modi durch die Kriterien 1.1 und 1.2 adressiert wurde, steht hier also die Fähigkeit zum kontextabhängigen Wechsel zwischen den Modi im Vordergrund.

1.3.1 Anpassung der beanspruchten Kapazitäten an den Bedarf

Kann das Softwareprodukt im laufenden Betrieb Hardwarekapazitäten freigeben (und damit auch seinen Energieverbrauch reduzieren), wenn diese nicht benötigt werden?

Indikatoren:

a) Verfügt das Softwareprodukt über unterschiedliche Modi, deren Wechsel sich messbar auf den Energieverbrauch auswirkt?
b) Wechselt das Softwareprodukt dynamisch jeweils in einen sparsameren Modus, wenn das möglich ist? (z. B. Schlafmodus)

c) Falls für die Software spezielle Einstellungen für einen sparsamen Betriebsmodus vorgenommen werden müssen, sind diese Konfigurationsmöglichkeiten zentral und allgemein verständlich zusammengefasst?[23]

1.3.2 Anpassung des Bedarfs an die verfügbaren Kapazitäten

Kann das Softwareprodukt im laufenden Betrieb seinen Bedarf an Hardwarekapazitäten (und damit auch seinen Energieverbrauch) reduzieren, wenn das Angebot sich dynamisch verringert? (z. B. wenn weniger Bandbreite zur Verfügung steht oder der Akku sich leert)

Indikatoren:

a) Wechselt das Softwareprodukt in einen sparsameren Modus, wenn das Angebot an Hardwarekapazitäten oder Energie sich verringert, ohne dass Fehler oder Datenverluste auftreten? (keine Einschränkung, langsamere Ausführung, Fehler in der Ausführung)
b) Ist die Software auch bei aktiviertem Energiemanagement der darunterliegenden Systemschichten oder der verbundenen Clientsysteme uneingeschränkt funktional nutzbar?[24] (ja, uneingeschränkt nutzbar / eingeschränkt nutzbar / nicht nutzbar)

1.3.3 Ressourcenschonende Standardeinstellungen

Sind die Standardeinstellungen des Softwareprodukts so gewählt, dass sie das Ziel der Ressourcenschonung mitberücksichtigen?[25]

Indikatoren:

a) Einschätzung des Prüfers, ob Standardeinstellungen des Softwareprodukts so gewählt sind, dass sie das Ziel der Ressourcenschonung mitberücksichtigen

1.3.4 Feedback zur Beanspruchung von Hardwarekapazitäten und Energie

Können die vom Softwareprodukt aktuell beanspruchten lokalen und entfernten Hardwarekapazitäten und resultierenden Energieverbräuche angezeigt werden und ist die Anzeige korrekt?

Indikatoren:

a) Werden die beanspruchten Hardwarekapazitäten, der Datenfluss und die Energieverbräuche angezeigt? (funktionsspezifisch vorhanden mit Vorschlägen zur Ressourceneinsparung, funktionsspezifisch vorhanden, für das Gesamtprodukt vorhanden, nicht vorhanden)
b) Einschätzung des Prüfers, ob Anzeige korrekt (korrekt / nicht korrekt)

[23] Beispiele: Hintergrund-/Sleep-Einstellungen, Animationen, rechenintensive Prozesse wie Index etc., Cache-Größen, zeitliche Steuerbarkeit von Prozessen, um ökologisch günstigere Energiezufuhr auszunutzen (demand shaping).

[24] Insbesondere ist bei serverbasierter Software zu fordern, dass das Aktivieren des Energiemanagements auf Client-Seite die Funktionalität nicht beeinträchtigt. Beispielsweise darf es nicht zu einem Verlust von Session-Information führen, wenn der Client in einen Sleep-Mode wechselt.

[25] Beispiel: Standardeinstellung beim Drucken: Wird Papier beidseitig bedruckt, wenn der Drucker dazu in der Lage ist?

2 Potenzielle Hardware-Nutzungsdauer

Zu welchem Grad sind Hardware-Erneuerungszyklen von Software-Erneuerungszyklen entkoppelt?[26]

Software stellt Anforderungen an die Hardware, auf der sie ausgeführt wird. Je schneller diese Anforderungen mit der Weiterentwicklung des Softwareprodukts wachsen und je spezifischer sie sind, desto einschränkender wirken sie sich auf den Einsatz bereits vorhandener Hardwareprodukte aus. Können bereits vorhandene Hardwareprodukte nicht (mehr) eingesetzt werden, um das gegebene Softwareprodukt auszuführen, verkürzt dies die Nutzungsdauer der Hardware.

Das angestrebte Ideal ist ein Softwareprodukt, dessen Entwicklungsdynamik dem Betreiber ein davon unabhängiges, entkoppeltes Management seiner Hardwareprodukte erlaubt.

2.1 Abwärtskompatibilität

Garantiert der Hersteller des Softwareprodukts, dass das aktuelle Release auf einem Referenzsystem von vor *n* Jahren betrieben werden kann?[27]

Indikatoren:

a) Zunächst Herstelleraussage (Hardware, ältere Betriebssysteme, ältere Frameworks), da für zurückliegende Jahre keine Standardkonfigurationen definiert sind
b) Wenn seit der ersten Anwendung dieses Kriteriums auf das Softwareprodukt ausreichend Zeit verstrichen ist, so dass man das Standardnutzungsszenario auch auf früheren Standardkonfigurationen ausführen kann: Ist das Standardnutzungsszenario mit dem aktuellen Release des Softwareprodukts auf einer Konfiguration ausführbar, die vor einer noch festzulegenden Anzahl von Jahren Standardkonfiguration war?

[26] Entkopplung der Software- und Hardware-Erneuerungszyklen bedeutet hohe potenzielle Hardware-Nutzungsdauer. Grundannahme: Jedes Softwareprodukt benötigt eine Systemumgebung als Plattform, auf der es ausgeführt wird. Alle zur Ausführung notwendigen Hard- und Softwarekomponenten des IKT-Systems zählen zu dieser Systemumgebung. Das Softwareprodukt selbst kann Teil der Systemumgebung anderer Softwareprodukte sein. Beispiel: Ein Webbrowser benötigt ein Betriebssystem, weitere Systemsoftware und Hardware als Systemumgebung und bildet zugleich die Systemumgebung für eine Web-Anwendung. Von einem gegebenen Softwareprodukt aus betrachtet interessiert es, welche Ansprüche seine Erneuerung über die verschiedenen Schichten der Systemumgebung an die unterste Schicht – die Hardware – auslöst.

[27] Somit kann das Softwareprodukt mit einer üblichen Hardwarekonfiguration betrieben werden, die schon *n* Jahre in Betrieb ist.

2.2 Plattformunabhängigkeit und Portabilität

Kann das Softwareprodukt auf verschiedenen aktuell verbreiteten produktiven Systemumgebungen (Hardware und Software) betrieben werden und können die Nutzenden zwischen diesen ohne Nachteil wechseln?[28]

Indikatoren:

a) Herstellerangaben (kompatibel mit verschiedenen Betriebssystemen, Laufzeitumgebungen)
b) Standardnutzungsszenario auf verschiedenen aktuell verbreiteten produktiven Systemumgebungen ausführen, dabei die Daten- und Softwareeinstellungsportabilität prüfen

2.3 Hardwaresuffizienz

Bleibt die Menge an beanspruchter Hardwarekapazität bei Weiterentwicklung des Softwareprodukts auch bei Funktionserweiterungen über die Zeit konstant?

Dieses Kriterium belohnt Hersteller von Softwareprodukten, die keine über die Zeit wachsenden Anforderungen an die Hardware stellt, wodurch die Hardware nicht aufgerüstet oder erneuert werden muss. Es berücksichtigt bewusst nicht, ob die Funktionalität erweitert wurde. Suffizienz ist so zu verstehen, dass die beanspruchten Ressourcen auch dann *nicht* zunehmen, wenn man mehr Nutzen aus ihnen zieht (was durch Zunahme der Effizienz möglich ist).

Das angestrebte Ideal ist ein Softwareprodukt, das von Version zu Version zwar höheren Anforderungen der Nutzenden genügt, aber dennoch keine höheren Anforderungen an die Hardware stellt.

Dieses Kriterium ist erst anwendbar, wenn Produkte schon mehrfach beurteilt wurden, also mindestens ein zurückliegendes Ergebnis schon vorliegt.

Indikatoren:

a) Intertemporale Vergleiche mit folgenden Ergebnisstufen:

1. „sehr gut“: Die Versionswechsel haben bisher zu einer Verringerung der benötigten Hardwarekapazitäten geführt.
2. „gut“: Die Versionswechsel haben bisher nicht zu einer Erhöhung der benötigten Hardwarekapazitäten geführt.
3. „genügend“: Die Versionswechsel haben bisher zwar zu einer Erhöhung der benötigten Hardwarekapazitäten geführt, diese blieben jedoch im Rahmen des technisch bedingten Effizienzfortschritts, wie er sich in der zeitlichen Abfolge von Referenzsystemen ausdrückt.
4. „ungenügend“: Die Versionswechsel haben dazu geführt, dass die benötigten Hardwarekapazitäten schneller gewachsen sind als die technische Effizienz entsprechend der Abfolge der Referenzsysteme.

[28] Wir empfehlen, dieses Kriterium nicht zu den Minimalanforderungen zu zählen, da es grundsätzlich sehr ressourcenschonende Software geben kann, die nur auf einer einzigen Plattform läuft. Dennoch ist Plattformunabhängigkeit positiv zu werten, da sie die Freiheitsgrade des Nutzenden bei der Optimierung der Beschaffung Hardware und Systemsoftware vergrößert.

3 Nutzungsautonomie

Respektiert der Hersteller des Softwareprodukts die Autonomie des Nutzenden im Umgang mit dem erworbenen Produkt?

Dieses Hauptkriterium geht von der Annahme aus, dass eine relevante Zahl von Nutzenden an einem ressourcenschonenden Einsatz von Software interessiert ist. Wenn es ihnen ohne funktionelle Nachteile möglich ist, werden sie versuchen, mit geringer Hardwarekapazität auszukommen (für die sie in der Regel bezahlen) und auch den Energieverbrauch gering zu halten (der ebenfalls finanziell relevant ist oder zumindest die Akkulaufzeit mobiler Geräte beeinflusst). Dies ist jedoch nur unter der Voraussetzung möglich, dass die Nutzenden nicht zu unnötiger Beanspruchung von Ressourcen gezwungen werden und verstehen, wie sie sie vermeiden können.

Das angestrebte Ideal ist ein Softwareprodukt, das die Freiheit der Nutzenden über die Beanspruchung von Kapazitäten (und damit indirekt von Ressourcen) bei Nutzung des Produkts selbst zu entscheiden, möglichst weitgehend respektiert.

Die folgenden Kriterien sind aus der Perspektive von technikfernen Zielgruppen zu beurteilen, sie sind also in der Regel nicht dadurch schon erfüllt, dass sie bei Nutzung des Produkts durch eine speziell ausgebildete oder speziell geübte Fachperson erfüllt wären. Eine Ausnahme hiervon bildet Kriterium 3.1.2.

3.1 Transparenz und Interoperabilität

Sind ressourcenrelevante Aspekte des Softwareprodukts für Nutzende mit vernünftigem Aufwand nachvollziehbar? Können Nutzende die Daten, die sie mit dem Softwareprodukt erzeugt haben, mit anderen Softwareprodukten weiterverwenden?

3.1.1 Transparenz der Datenformate und Datenportabilität

Sind die Datenformate (Datei- oder Datenstromformate), die das Softwareprodukt zur Ablage der von Nutzenden erzeugten Daten oder zum Austausch von Daten mit anderen Programmen verwendet, ausreichend dokumentiert, um Interoperabilität zu ermöglichen? Folgen die Datenformate offenen Standards, so dass eine Weiternutzung der Daten mit einem anderen Softwareprodukt möglich ist?[29]

Zur Anwendung dieses Kriteriums muss definiert sein, welche Standards zum Zeitpunkt der Vergabe zu den offenen Standards gezählt werden.

Indikatoren:

a) Überprüfung der Handbücher und technischen Datenblätter, ob Datenformate ausreichend dokumentiert sind
b) Abgleich mit bekannten und offenen Standards

[29] Dies ist entscheidend für Vermeidung einer Abhängigkeit vom Softwareprodukt (Customer Lock-In), welche unnötigen Ressourcenverbrauch erzwingen kann, sowohl im Falle der Beibehaltung eines ineffizienten Produkts als auch im Falle eines (aufwändigen) Umstiegs auf ein anderes Produkt.

3.1.2 Transparenz und Interoperabilität der Programme

Sind Anwendungs-Programmier-Schnittstellen (APIs) klar dokumentiert und wird die Verbreitung und Weiterentwicklung des Programms unterstützt? Folgen die Schnittstellen offenen Standards, die Interoperabilität ermöglichen?

Die Gewichtung der Indikatoren ist möglicherweise stark kontextabhängig. Die Auswirkung von offenem Quellcode und bestimmter Lizenzmodelle auf die Beanspruchung von Ressourcen kann nicht im Sinne einer generellen Regel beurteilt werden.

Indikatoren:

a) Wenn es APIs gibt: Überprüfung der Schnittstellendokumentation anhand der Dokumentation des Softwareprodukts und seiner APIs
b) Ist der Quellcode vollständig offengelegt?
c) Ist das Softwareprodukt unter einer Lizenz veröffentlicht, die es erlaubt, es weiterzuentwickeln?

3.1.3 Kontinuität des Softwareproduktes

Kann das Softwareprodukt über längere Zeiträume genutzt werden, ohne dass schwerwiegende Nachteile (insbesondere Probleme der IT-Sicherheit) auftreten, und hat der Nutzende die Wahl, unnötige Updates zu vermeiden?[30]

Indikatoren:

a) Wie lang ist der Zeitraum, für den der Anbieter die zukünftige Unterstützung des Produkts mit Sicherheitsupdates garantiert?
b) Reagiert der Hersteller zeitnah auf das Bekanntwerden von Sicherheitslücken?
c) Kann der Nutzende durch Konfiguration die Updatehäufigkeit beeinflussen und dabei insbesondere zwischen Sicherheitsupdates und sonstigen Updates differenzieren?
d) Besteht die Möglichkeit, nur differenzielle Updates zu erhalten?[31]

3.1.4 Transparenz des Prozessmanagements

Macht das Softwareprodukt die Nutzenden darauf aufmerksam, dass es im Hintergrund automatisch Prozesse startet oder weiterlaufen lässt, die möglicherweise nicht genutzt werden?

Indikatoren:

a) Prüfung anhand der Installation und der Ausführung von Standardnutzungsmustern, welche Prozesse das Softwareprodukt automatisch startet und ob es darauf aufmerksam macht (es macht auf alles aufmerksam/auf einiges/macht nicht aufmerksam)

[30] Eine hohe Updatefrequenz verursacht Ressourcenverbrauch und erschwert die Aufrechterhaltung von Transparenz. Das Kriterium der Notwendigkeit von Updates ist grundsätzlich schwer objektivierbar; eine Unterscheidung zwischen sicherheitsrelevanten (und damit zweifellos notwendigen) und anderen Updates ist in jedem Fall sinnvoll und wird daher durch Indikator b) adressiert.

[31] Dies vermeidet den Austausch des kompletten Programms, der bei häufigen Updates erheblichen Ressourcenaufwand verursachen kann.

b) Wenn das Softwareprodukt beim Systemstart automatisch gestartet wird („Autostart“): Macht es darauf aufmerksam, dass dies der Fall ist?
c) Wenn der Nutzende eine Aktion durchführt, die als Beendigung des Programms aufgefasst werden kann, aber mindestens einer der Prozesse noch aktiv bleibt: Macht das Softwareprodukt darauf aufmerksam?

3.2 Deinstallierbarkeit

Lässt sich das Softwareprodukt einfach, rückstandsfrei und ohne vermeidbare Nachteile deinstallieren?

3.2.1 Deinstallierbarkeit der Programme

Werden Nutzende ausreichend darin unterstützt, das Softwareprodukt rückstandsfrei zu deinstallieren?

Indikatoren:

a) Deinstallation der Software und Vergleich mit dem Zustand vor der Installation, der gleich sein muss.

3.2.2 Löschbarkeit der Daten

Werden Nutzende ausreichend darin unterstützt, die während des Betriebs des Softwareprodukts generierten Daten, die sie nicht explizit angelegt haben, nach Bedarf zu löschen?

Dieses Kriterium zielt darauf ab, dass die vorübergehende Nutzung eines Softwareprodukts keine Datenspuren hinterlässt, wenn die Nutzenden dies nicht wünschen. Wenn später beispielsweise eine andere Version des Produkts oder ein Konkurrenzprodukt installiert wird, soll die Historie für die neue Software nicht erkennbar sein. Damit soll Lock-In-Effekten und der Missachtung von Datenschutzprinzipien vorgebeugt werden.

Indikatoren:

a) Ist der Zustand nach dem Löschen der nicht durch den Nutzer ausdrücklich gespeicherten Daten mit dem Zustand vor der Installation in relevanter Hinsicht identisch?
b) Macht das Softwareprodukt transparent, wo die explizit angelegten Daten gespeichert werden (Speicherort)?
c) Wird der Nutzende dabei unterstützt, die auf entfernten Speicherkapazitäten angelegten Datenbestände zu löschen?

3.3 Wartungsfunktionen

Bietet das Softwareprodukt einfach zu benutzende Funktionen, die es erlauben, eingetretene Schäden an Daten und Programmen zu beheben?

3.3.1 Datenwiederherstellbarkeit

Lässt sich der letzte Zustand der Daten nach einem unerwünschten Programmabbruch wiederherstellen?

Indikatoren:

a) Macht der Hersteller dazu Angaben, und lassen sich diese im Test bestätigen?
b) Ist der Zeitraum, in dem Änderungen an den Daten zwischengespeichert werden, parametrisierbar?

3.3.2 Selbstreparaturfähigkeit

Lässt sich die installierte Instanz des Softwareprodukts nach dem Eintreten eines inkonsistenten Zustands wiederherstellen?

Indikatoren:

a) Herstellerangaben und Überprüfung durch Test

3.4 Unabhängigkeit von Fremdressourcen

Lässt sich das Softwareprodukt möglichst unabhängig von Hardwarekapazitäten betreiben, die nicht der Kontrolle der Nutzenden unterliegen?

3.4.1 Offlinefähigkeit

Zu welchem Grad vermeidet das Softwareprodukt Zwänge zur Konnektivität, die nicht aus der Funktionserfüllung resultieren?[32]

Indikatoren:

a) Überprüfung durch Standardnutzungsszenario (Offline-Betrieb möglich/mit Einschränkungen möglich/unmöglich)

3.5 Qualität der Produktinformation

Unterstützt die angebotene Information über das Softwareprodukt seine ressourcenschonende Nutzung?

3.5.1 Verständlichkeit und Überschaubarkeit der Produktdokumentation, Lizenz- und Nutzungsbedingungen

Sind alle Angaben für die Nutzenden leicht nachvollziehbar?

Indikatoren:

a) Augenscheinprüfung durch Prüfende; Test mit tatsächlichen Nutzenden

[32] Beispiele für unnötige Konnektivitätszwänge: Verbindung zum Lizenzserver herstellen, wiederholtes Herunterladen von Fonts notwendig.

3.5.2 Ressourcenrelevanz der Produktinformation

Enthält die Produktinformation alle Angaben, die die Nutzenden benötigen, um die Ressourcenbeanspruchung durch das Softwareprodukt gering zu halten, in strukturierter Form, und sind die Angaben korrekt?

Langfristiges Ziel ist die Entwicklung standardisierter Produktbeschreibungen für ressourcenrelevante Produktinformation. Sobald es hierzu einen befriedigenden Standard gibt, kann seine Einhaltung als Indikator aufgenommen werden.

Indikatoren:

a) Qualitative Beurteilung der Vollständigkeit und Verständlichkeit
b) Bezieht sich die Produktinformation auf die aktuelle Produktversion?

Augenscheinprüfung der Korrektheit der Angaben (Angaben sind schlüssig / eingeschränkt schlüssig / nicht schlüssig)

Anhang 2 Beispielhafter Ablauf eines Standardnutzungsszenarios für Textverarbeitungssoftware

Folgender Ablauf zeigt am Beispiel von Textverarbeitungssoftware die Aktionen, die in einem Standardnutzungsszenario durchgeführt werden. Zur Vereinfachung der Standardisierung und Reproduzierbarkeit wurde das englischsprachige Project Gutenberg E-Book „Surgical Anatomy by Joseph Maclise“[33] verwendet, das für die Messungen in das jeweils von der Textverarbeitungssoftware standardmäßig verwendete Format konvertiert wurde.

Vor Start der Messung sicherstellen:

- Von automatisiertem Ablauf erstellte Dateien existieren nicht
- Dokument-Zoom auf 125%, Software öffnet im Fullscreen
- Lineale sind eingeschaltet
- Steuerzeichen sind eingeschaltet
- Einzufügenden Text in Zwischenablage kopieren

Ablauf des Makros auf dem SUT:

1. Dokumentverknüpfung öffnen
2. Warten bis Dokument geöffnet ist
3. Gesamten Text ändern
    a. In den Text klicken
    b. Alles markieren: Strg+a
    c. Schriftart ändern in „Ubuntu Condensed“
    d. Ausrichtung nach Blocksatz ändern
    e. Schriftgröße auf 9 setzen
    f. In Text klicken um Auswahl aufzuheben
4. Als neues Dokument speichern
5. Inhaltsverzeichnis einfügen
    a. Springe zu Kapitel „Table of Contents“
    b. Ändere die Formatvorlage für den Text nach „Überschrift“
    c. Füge ein Inhaltsverzeichnis in die darunterliegende Zeile ein
    d. Scrolle über das Inhaltsverzeichnis
6. Titelseite einfügen
    a. Mittels Seitenleiste zum Dokumentanfang
    b. Neue, leere Seite am Anfang des Dokuments einfügen
    c. Erste Seite formatieren
7. Ansicht anpassen
    a. Lineale aus- und wieder einschalten
    b. Steuerzeichen aus- und wieder einschalten
8. Titel einfügen
    a. „Surgical Anatomy“ als Titel eintippen
    b. Formatvorlage „Titel“ auswählen
    c. Titel Formatvorlage ändern: Schriftart „Ubuntu“-fett, Größe: 48 Pixel
9. Fußzeile einfügen und formatieren
    a. Fußzeile einfügen
    b. In Fußzeile Seitennummer einfügen

[33] Vgl. https://www.gutenberg.org/ebooks/24440

    c. „Page “ vor Seitennummer schreiben
    d. Bearbeitung der Fußzeile beenden
10. Text einfügen
    a. Springe zu „Page 9“
    b. Seitenumbruch hinter Zeile einfügen
    c. Neue Überschrift „The Heart“ eintippen und als Überschrift 1 Formatieren
11. Zoom auf 50%
12. Bild 1 (Herz) einfügen
13. Zoom auf 125%
14. Text eintippen und bearbeiten
    a) Bildunterschrift hinzufügen: „The Heart Muscle“
    b) Darunter einen Querverweis einfügen: Abbildung 1:
    c) Folgenden Text eintippen:
       shows the heart. It is a muscular organ in humans and other animals, which pumps blood through the blood vessels of the circulatory system. Blood provides the body with oxygen and nutrients, as well as assists in the removal of metabolic wastes. The heart is located in the middle compartment of the chest.
    d) Speichern (Strg-s)
    e) Die Wörter „pumps blood through the blood vessels of the circulatory system“ markieren und Schriftstil „fett“ auswählen
    f) Folgenden Text aus Zwischenablage einfügen:
       In humans, other mammals, and birds, the heart is divided into four chambers: upper left and right atria; and lower left and right ventricles. Commonly the right atrium and ventricle are referred together as the right heart and their left counterparts as the left heart. Fish in contrast have two chambers, an atrium and a ventricle, while reptiles have three chambers. In a healthy heart blood flows one way through the heart due to heart valves, which prevent backflow. The heart is enclosed in a protective sac, the pericardium, which also contains a small amount of fluid. The wall of the heart is made up of three layers: epicardium, myocardium, and endocardium.
    g) Zeilenumbruch
15. Rechtschreibprüfung ab soeben eingegebenen Text
16. Bild einfügen
    a) Bild 2 (Rippenkäfig) einfügen
    b) Bildunterschrift hinzufügen: „The Ribcage with Sternum“
    c) Zeilenumbruch
17. Tabelle einfügen
    a) Tabelle einfügen: 5x5 Felder
    b) Überschriften eintippen: Bone name, Count, Classification, Location, Latin name
18. Inhaltsverzeichnis aktualisieren
19. Speichern (Strg-s)
20. Im Dokument suchen + ersetzen: „downwards“ durch „down“ ersetzen → 29 Ersetzungen
21. PDF erzeugen
22. Programm schließen und speichern
23. Bis Minute 10 warten

Nach Ablauf einer Messung werden die erzeugten Dateien wieder mittels eines WinAutomation-Befehls gelöscht um die Konsistenz des Ablaufs sicherzustellen.

Anhang 3 Operationalisierung für die Aufnahme der Indikatoren

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
1	**Ressourceneffizienz**			
1.1	**Hardwareeffizienz**			
1.1.1	Empfohlene Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)	a) Empfohlene lokale Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems (Produkt aus Taktfrequenz, Anzahl Kerne, Busbreite)	Die empfohlenen Anforderungen werden (falls vorhanden) aus Produktverpackung, Benutzerdokumentation, Readme-Datei oder Produktwebseite ermittelt und anschließend in den jeweiligen Prozentanteil des Referenzsystems umgerechnet. Wenn das Softwareprodukt zum Betrieb weitere Softwareprodukte voraussetzt oder empfiehlt, werden deren empfohlene Systemvoraussetzungen zusätzlich analog berücksichtigt. Der höhere Wert wird gewertet.	Für jeden Indikator wird die empfohlene Anforderung notiert, sowie der prozentuelle Anteil des Referenzsystems **Hinweis:** Je nach Software kann das gesamte Kriterium entfallen (wenn keine empfohlenen Anforderungen angegeben sind) **Hinweis:** Kann auch größer 100% sein, wenn Referenzsystem die empfohlenen Anforderungen nicht erreicht. **Hinweis:** 1.1.1.e) bis 1.1.1.h) Entfallen bei lokalen Anwendungen
		b) Empfohlener lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems		
		c) Empfohlener lokaler Permanentspeicher laut Herstellerangaben in % des Permanentspeichers des Referenzsystems		
		d) Empfohlene Displayauflösung laut Herstellerangaben in % der Displayauflösung des Referenzsystems		
		e) Empfohlene Bandbreite für Netzzugang laut Herstellerangaben in % der Bandbreite des Referenzsystems		
		f) Empfohlene serverseitige Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems		
		g) Empfohlener serverseitiger Arbeitsspeicher laut Herstellerangaben in % des serverseitigen Arbeitsspeichers des Referenzsystems		
		h) Empfohlener serverseitiger Permanentspeicher laut Herstellerangaben in % des serverseitigen Permanentspeichers des Referenzsystems		
1.1.2	Minimale Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)	a) Minimale lokale Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems	Die minimalen Anforderungen werden (falls vorhanden) aus Produktverpackung, Benutzerdokumentation, Readme-Datei oder Produktwebseite ermittelt und anschließend in den jeweiligen Prozentanteil des Referenzsystems umgerechnet. Wenn das Softwareprodukt zum Betrieb weitere Softwareprodukte voraussetzt, werden deren minimale Systemvoraussetzungen zusätzlich analog berücksichtigt. Der höhere Wert wird gewertet.	Für jeden Indikator wird die Mindestanforderung notiert, sowie der prozentuelle Anteil des Referenzsystems **Hinweis:** 1.1.2.e) bis 1.1.2.h) entfallen bei lokalen Anwendungen
		b) Minimaler lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems		
		c) Minimaler lokaler Permanentspeicher laut Herstellerangaben in % des Permanentspeichers des Referenzsystems		
		d) Minimale Displayauflösung laut Herstellerangaben in % der Displayauflösung des Referenzsystems		

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
		e) Minimale Bandbreite für Netzzugang laut Herstellerangaben in % der Bandbreite des Referenzsystems		
		f) Minimale serverseitige Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems		
		g) Minimaler serverseitiger Arbeitsspeicher laut Herstellerangaben in % des serverseitigen Arbeitsspeichers des Referenzsystems		
		h) Minimaler serverseitiger Permanentspeicher laut Herstellerangaben in % des serverseitigen Permanentspeichers des Referenzsystems		
1.1.3	Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration	a) Messung der mittleren Prozessorauslastung im Leerlauf unter Standardkonfiguration (Differenz zum Grundverbrauch der Standardkonfiguration ohne das Softwareprodukt im gleichen Zeitraum)	Zunächst wird das angepasste leere Betriebssystem-Image (vgl. Messmethodik) auf das SUT geladen. Darauf werden alle zum Betrieb der Software benötigten Softwareprodukte mit Standardeinstellungen installiert (z.B. Laufzeitumgebungen, Datenbanken etc.). Anschließend wird das eigentliche Softwareprodukt mit Standardeinstellungen installiert. Von diesem Zustand wird erneut ein Image erzeugt, um den Installationsschritt fortan überspringen zu können. Danach wird die Software gestartet und die Messung der Hardwareauslastung des SUT wie in der Messmethodik beschrieben (10 Messdurchläufe á 10 Minuten, 1 Messwert pro Sekunde, 1 Minute Vorlaufzeit) durchgeführt, während das Softwareprodukt aktiv ist, aber keine Nutzerinteraktion stattfindet. Sind alle Werte aufgenommen, wird der Mittelwert der Mittelwerte anhand der Formel $U_P = \frac{1}{n}\sum_{k=1}^{n}\bar{x}_k$ gebildet. Dabei ist x_k das arithmetische Mittel des k-ten Messdurchlaufs in %. Anschließend wird wie im Kriterienkatalog beschrieben die Hardware-Inanspruchnahme als Integral der Hardware-Auslastung über die Ausführungsdauer des Szenarios berechnet. Die Einheiten der Inanspruchnahme sind Einheiten für die Arbeitsleistung	Für jeden Indikator wird der ermittelte Wert der Hardware-Inanspruchnahme notiert.
		b) Messung der mittleren Arbeitsspeicherbelegung im Leerlauf unter Standardkonfiguration		
		c) Messung der mittleren Permanentspeicherbelegung im Leerlauf unter Standardkonfiguration		
		d) Messung der mittleren beanspruchten Bandbreite für Netzzugang im Leerlauf unter Standardkonfiguration		
		e) Messung der mittleren serverseitigen Prozessorauslastung im Leerlauf unter Standardkonfiguration		
		f) Messung der mittleren serverseitigen Arbeitsspeicherbelegung im Leerlauf unter Standardkonfiguration		
		g) Messung der mittleren serverseitigen Permanentspeicherbelegung im Leerlauf unter Standardkonfiguration		

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
			(also %*s, MB*s*s=MB, MBit/s*s=MBit)	
1.1.4	Hardware-Inanspruchnahme bei normaler Nutzung unter der Annahme einer Standardkonfiguration und eines Standardnutzungsszenarios	a) Messung der Prozessorarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration b) Messung der Arbeitsspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration c) Messung der Permanentspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration d) Messung der übertragenen Datenmenge für Netzzugang bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration e) Messung der serverseitigen Prozessorarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration f) Messung der serverseitigen Arbeitsspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration g) Messung der serverseitigen Permanentspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	Ablauf der Installation analog 1.1.3. Anschließend Start der Software und Messung der Hardwareauslastung des SUT wie in der Messmethodik beschrieben (30 Messdurchläufe á ca. 10 Minuten, 1 Messwert pro Sekunde, 1 Minute Vorlaufzeit), während das Standardnutzungsszenario automatisiert abgespielt wird. Sind alle Werte aufgenommen wird der Mittelwert der Mittelwerte anhand der Formel $U_P = \frac{1}{n}\sum_{k=1}^{n}\bar{x}_k$ gebildet. Dabei ist x_k das arithmetische Mittel des k-ten Messdurchlaufs in %. Anschließend wird wie im Kriterienkatalog beschrieben die Hardware-Inanspruchnahme als Integral der Hardware-Auslastung über die Ausführungsdauer des Szenarios berechnet. Die Einheiten der Inanspruchnahme sind Einheiten für die Arbeitsleistung (also %*s, MByte/s*s = MByte, MBit/s*s = MBit)	Für jeden Indikator wird der ermittelte Wert der Hardware-Inanspruchnahme notiert.
1.1.5	Sparsame Hardwarenutzung durch Anpassbarkeit und Unterstützung der Nutzenden bei der Anpassung des Softwareprodukts		Das Softwareprodukt wird mit Standardeinstellungen installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Geschieht die Minimierung der beanspruchten Kapazitäten automatisch und/oder gibt es bei der Installation der Software entsprechende Optionen? (Skala: ja/nein)	Während der Installation wird geprüft, ob entsprechende Optionen angeführt werden. Bsp: Automatische Updates, Wörterbücher, Sprachen, Zusatzmodule	Es wird notiert, ob die Minimierung der Kapazitäten automatisch geschieht und ob entsprechende Optionen existieren. Zusätzlich wird notiert, welche Optionen existieren.
		b) Falls die Nutzenden eine entsprechende Wahl treffen, ist die Entscheidung für oder gegen Installationsoptionen später jederzeit revidierbar (Skala: ja/nein)	Nach Abschluss der Installation wird geprüft, ob die in 1.1.5.a) aufgeführten Optionen in der Softwarekonfiguration verändert werden können und welche Voraussetzungen es dazu gibt (z.B. Installationsmedium wird benötigt etc.).	Es wird notiert, ob die während der Installation getroffene Wahl später geändert werden kann und welche Voraussetzungen bestehen. **Hinweis:** Entfällt ggf. abhängig von 1.1.5.a)
		c) Black-Box-Test: Können hardwareintensive Software-Module abgeschaltet werden? (Skala: können abgeschaltet werden/können nicht abgeschaltet werden)	Nach Einschätzung des Prüfenden und anhand einer Analyse des Verbrauchsverlaufs im Standardnutzungsszenario werden hardwareintensive Optionen, Module oder Funktionen der	Die identifizierten Optionen, Module oder Funktionen werden, zusammen mit der Aussage ob sie deaktiviert werden können, notiert.

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
			Software identifiziert. Anschließend wird geprüft, ob diese deaktiviert werden können.	
		d) Ist es ohne Nachteil möglich, vorübergehend oder dauerhaft nicht benötigte Peripheriegeräte abzuschalten bzw. gar nicht bereitzustellen? (Skala: Können vorübergehend und dauerhaft abgeschaltet werden/können nur vorübergehend abgeschaltet werden/können nicht abgeschaltet werden)	Zunächst werden die von dem Produkt unterstützten Peripheriegeräte identifiziert (z.B. Webcam, Drucker, Scanner, Mikrofon, etc.). Für alle Geräte, die nicht für das Standardnutzungsszenario nötig sind wird geprüft, ob das Szenario auch ohne das Gerät durchgeführt werden kann. Für alle Geräte, die eigentlich für das Standardnutzungsszenario nötig sind wird geprüft, ob eine eingeschränkte Nutzung nach Abschaltung weiterhin möglich ist.	Die unterstützten Geräte werden, zusammen mit der Aussage zur Notwendigkeit notiert.
		e) Werden nach der Installation die Dateien gelöscht, die nur zur Installation benötigt werden?	Es wird geprüft, ob die ggf. während der Installation angelegten Dateien und Ordner, die nur für den Installationsprozess notwendig sind, nach Abschluss der Installation automatisch gelöscht werden.	Es wird notiert ob nach der Installation ausschließlich Dateien und Ordner vorhanden sind, die für den Betrieb des Produkts notwendig sind.
1.1.6	Online-Auslieferung		Suche auf Website/Webshop des Anbieters und Festhalten der Angaben	
		a) Sind eine Online-Auslieferung und ein Online-Update der Software möglich?	Aus den Herstellerangaben in der offiziellen Dokumentation oder Produktwebseite wird ermittelt ob Auslieferung und Update online möglich sind. Anschließend wird die Downloadfunktionalität geprüft und die Größe des Downloads notiert. Falls eine ältere Version verfügbar ist, wird diese abschließend installiert und geprüft, ob ein nachträgliches Update auf neuste Version möglich ist.	Es wird notiert, ob Online-Auslieferung und -Update angeboten werden und, falls dies der Fall ist, wird zusätzlich die Größe des Downloads notiert.
		b) Wird unterstützt, dass das Softwareprodukt und seine Updates in der anwendenden Organisation zentral abgelegt werden?	Es wird geprüft, ob die heruntergeladenen Installationsdateien so abgelegt werden können, dass das Softwareprodukt ohne weiteren Download auch auf anderen Geräten installiert werden kann.	Es wird notiert ob dies möglich ist. **Hinweis:** Entfällt ggf. abhängig von 1.1.6.a)

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
1.2	**Energieeffizienz**			
		a) Messung der zur Ausführung des Standardnutzungsszenarios unter Standardkonfiguration auf dem lokalen Gerät verbrauchten Energie	Ablauf der Installation analog 1.1.3. Anschließend Start der Software und Messung des Energieverbrauchs des SUT wie in der Messmethodik beschrieben (30 Messdurchläufe á ca. 10 Minuten, 1 Messwert pro Sekunde, 1 Minute Vorlaufzeit), während das Standardnutzungsszenario automatisiert abgespielt wird. Sind alle Werte aufgenommen wird der Mittelwert des Energieverbrauchs, gemäß der Beschreibung im Katalog, anhand der Formel $E = \frac{1}{3600 \cdot n} \sum_{k=1}^{n} \sum_{i=1}^{m} P_i$ berechnet als Summe der pro Sekunde gemittelten Leistungsaufnahme P_i des SUT. Durch die Division durch 3600 s/h erhält man die elektrische Arbeit in Wh.	Der berechnete Absolutwert der elektrischen Arbeit wird notiert.
		b) Schätzung der durch die Datenübertragung im Netz verbrauchten Energie aufgrund des bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration erzeugten Datenverkehrs (Verwendung einer aktuellen Schätzung für die Energieintensität des Netzes in kWh/GB basierend auf aktueller Fachliteratur, wenn nötig differenziert nach Zugangsnetzen)	Anhand der übertragenen Datenmenge, die in Indikator 1.1.4.d) ermittelt wurde und der gemessenen maximalen Datenübertragungsrate (D) des SUT wird die Zeit t(S) geschätzt, die die Übertragung der Daten GB(S) des Standardnutzungsszenarios (S) unter voller Auslastung der Bandbreite benötigt hätte: t(S) = GB(S) / D Anhand der von Schien et al. (2014) entwickelten Formel W(S) = t(S) * 52W + GB(S) * 0,052 kWh/GB wird anschließend der Energieverbrauch (in Wh) abgeschätzt. Die Formel wird durch die angenommene Verbesserung der Energieeffizienz der Geräte in 2017 wie folgt modifiziert: W(S) = t(S) * 52W + GB(S) * 0,052 kWh/GB * 0.25	Der mittels der Formel angenäherte Energieverbrauch wird notiert. **Hinweis:** Entfällt bei lokalen Anwendungen
		c) Messung der durch die entfernte Speicherung und Verarbeitung in Servern verbrauchten Energie bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration (falls Messung nicht möglich, Schätzung mit Hilfe durchschnittlicher Faktoren für die Energieintensität von Rechenzentrumsdienstleistungen basierend auf aktueller Fachliteratur)	Schätzung des durch die entfernte Speicherung und Verarbeitung auf nicht messbaren, entfernten Servern Energieverbrauchs mittels der übertragenen Datenmenge GB(S) anhand des in Valancius et al. (2009) entwickelten Schätzers: 211,1 Ws / GB	Es wird der geschätzte Energieverbrauch notiert. **Hinweis:** Entfällt bei lokalen Anwendungen

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
1.3	**Ressourcenmanagement**			
1.3.1	Anpassung der beanspruchten Kapazitäten an den Bedarf		Das Softwareprodukt wird mit Standardeinstellungen installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Verfügt das Softwareprodukt über unterschiedliche Modi, deren Wechsel sich messbar auf den Energieverbrauch auswirkt?	Zunächst werden verschiedene Modi des Produkts (z.B. Lesemodus, Bearbeitungsmodus in Textverarbeitungsprogrammen, Privater Modus in Browsern etc.) vom Prüfenden identifiziert. Anschließend wird der Energieverbrauch analog 1.2.1 für die Ausführung eines Szenarios in den identifizierten Modi gemessen und die Resultate verglichen.	Es wird notiert, ob das Produkt über verschiedene Modi verfügt und ob deren Wechsel sich auf den Energieverbrauch auswirkt.
		b) Wechselt das Softwareprodukt dynamisch jeweils in einen sparsameren Modus, wenn das möglich ist? (z. B. Schlafmodus)	Es wird beobachtet, ob und nach welcher Zeit das Produkt in einen sparsamen Modus wechselt.	Es wird notiert, ob ein solcher Modus existiert und, falls dies der Fall ist, nach welcher Zeit das Produkt in diesen wechselt.
		c) Falls für die Software spezielle Einstellungen für einen sparsamen Betriebsmodus vorgenommen werden müssen, sind diese Konfigurationsmöglichkeiten zentral und allgemein verständlich zusammengefasst?	Es wird eine Recherche in Produktdokumentation, Benutzerhandbuch oder Hersteller- bzw. Produktwebseite, sowie in den Optionen des Produkts nach dem Vorhandensein entsprechender energierelevanter Optionen durchgeführt. Sollte dies der Fall sein wird geprüft, ob diese übersichtlich zusammengefasst sind (z.B. eigener Menüpunkt oder gruppiert als Unteroptionen)	Es wird notiert ob solche Einstellungsmöglichkeiten existieren und, falls dies der Fall ist, ob sie übersichtlich zusammengefasst sind.
1.3.2	Anpassung des Bedarfs an die verfügbaren Kapazitäten		Das Softwareprodukt wird mit Standardeinstellungen auf einer virtuellen Maschine installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Wechselt das Softwareprodukt in einen sparsameren Modus, wenn das Angebot an Hardwarekapazitäten oder Energie sich verringert, ohne dass Fehler oder Datenverluste auftreten? (keine Einschränkung, langsamere Ausführung, Fehler in der Ausführung)	Die zur Verfügung gestellten Hardwarekapazitäten werden durch die Ausführung des Produktes in einer virtuellen Maschine eingeschränkt. Die Prozessorleistung und Größe des Hauptspeichers wird reduziert und es wird eine Augenscheinprüfung durchgeführt, ob bei der Ausführung des Standardnutzungsszenarios Funktionseinschränkungen auftreten.	Das Ergebnis der Augenscheinprüfung wird notiert.
		b) Ist die Software auch bei aktiviertem Energiemanagement der darunterliegenden Systemschichten oder der verbundenen Clientsysteme uneingeschränkt funktional nutzbar? (ja, uneinge-	Es wird eine Augenscheinprüfung durchgeführt um festzustellen, ob Funktionseinschränkungen auftreten wenn die Energiespareinstellungen der darunterliegenden Systemschichten (z.B.	Das Ergebnis der Augenscheinprüfung wird notiert.

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
		schränkt nutzbar / eingeschränkt nutzbar / nicht nutzbar)	Betriebssystem) aktiviert werden. Insbesondere bei Client-Software wird geprüft, ob das Aktivieren des Energiemanagements die Funktionalität beeinträchtigt. Z.B. darf es nicht zu einem Verlust von Session-Information führen, wenn der Client in einen Sleep-Mode wechselt.	
1.3.3	Ressourcenschonende Standardeinstellungen	a) Einschätzung des Prüfers, ob Standardeinstellungen des Softwareprodukts so gewählt sind, dass sie das Ziel der Ressourcenschonung mitberücksichtigen	Während und nach der Installation des Produkts wird eine Augenscheinprüfung durchgeführt in der die Standardeinstellungen dahingehend überprüft werden, ob das Ziel der Ressourcenschonung berücksichtigt wird, z.B. Sleep Modus vorhanden und aktiviert, welche Zeitspanne ist gewählt (vgl. 1.3.2.a), Werden nicht benötigte Komponenten standardmäßig mitinstalliert?, Welche Startseite wird bei Browsern standardmäßig geöffnet? etc.	Das Ergebnis der Augenscheinprüfung wird notiert.
1.3.4	Feedback zur Beanspruchung von Hardwarekapazitäten und Energie		Das Softwareprodukt wird mit Standardeinstellungen installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Werden die beanspruchten Hardwarekapazitäten, der Datenfluss und die Energieverbräuche angezeigt? (funktionsspezifisch vorhanden mit Vorschlägen zur Ressourceneinsparung, funktionsspezifisch vorhanden, für das Gesamtprodukt vorhanden, nicht vorhanden)	Es wird eine Recherche in der Produktdokumentation, dem Benutzerhandbuch, der Hilfe-Funktion oder der Hersteller- bzw. Produktwebseite (mögliche Schlagwörter: Energie, Energiemonitor, Monitoring, Ressourcen, Verbrauch, Strom, Akku, Datenvolumen, Traffic, Auslastung etc.), sowie in den Optionen des Produkts (z.B. ggf. vorhandene energierelevante Einstellungen vgl. 1.3.1.a) nach dem Vorhandensein entsprechender Anzeigen durchgeführt.	Es wird notiert, ob entsprechende Anzeigen verfügbar sind und welche Daten visualisiert werden.
		b) Einschätzung des Prüfers, ob Anzeige korrekt (korrekt / nicht korrekt)	Falls die beanspruchten Kapazitäten dargestellt werden können werden sie mit den Messdaten aus 1.1.3, 1.1.4, 1.2.1) verglichen und ihre Korrektheit eingeschätzt.	Das Ergebnis der Einschätzung wird notiert. **Hinweis:** Entfällt ggf. abhängig von 1.3.4.a)

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
2	**Potenzielle Hardware-Nutzungsdauer**			
2.1	Abwärtskompatibilität		Das Softwareprodukt wird mit Standardeinstellungen installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Zunächst Herstelleraussage (Hardware, ältere Betriebssysteme, ältere Frameworks), da für zurückliegende Jahre keine Standardkonfigurationen definiert sind	Es wird eine Recherche in der Produktdokumentation, dem Benutzerhandbuch und der Hersteller- bzw. Produktwebseite sowie der Dokumentation und Hersteller- bzw. Produktwebseite von weiterer Software, die der Betrieb des Produkts bedingt, durchgeführt um festzustellen, vor wie vielen Jahren das jüngste der mindestens benötigten Anforderungen noch lauffähig war. Falls der Hersteller Angaben zur Hardware macht, werden diese mit den vorhandenen Referenzsystemen verglichen und anschließend das Alter des ältesten vergleichbaren Referenzsystems als Wert des Indikators verwendet. Wenn Angaben zur Hardware nicht zur Verfügung stehen, wird das Veröffentlichungsdatum des jüngsten Betriebssystems verwendet, auf welchem sowohl das Softwareprodukt selbst, als auch die benötigten Softwareprodukte lauffähig sind. Beispiel: Benötigt ein Produkt als Mindestanforderung Windows 2000 (2000 erschienen) und Oracle 11g (2007 erschienen, aber lt. Hersteller lauffähig unter Windows 2000 => lauffähig auf System aus dem Jahr 2000), beträgt die Abwärtskompatibilität im Jahr 2017 17 Jahre.	Es wird notiert, welche weitere Software zum Betrieb des Produkts notwendig ist. Zusätzlich wird die älteste vom Produkt unterstützte Version der bedingenden Software, sowie das Release-Datum dieser Version. Schließlich wird die Abwärtskompatibilität als die vergangene Zeit seit dem jüngsten so notierten Release-Datum festgelegt. **Hinweis:** Auch zu beachten sind z.B. Release-Datumsangaben von verwendeten Dokumentformaten, Schnittstellendefinitionen etc.
		b) Wenn seit der ersten Anwendung dieses Kriteriums auf das Softwareprodukt ausreichend Zeit verstrichen ist, so dass man das Standardnutzungsszenario auch auf früheren Standardkonfigurationen ausführen kann: Ist das Standardnutzungsszenario mit dem aktuellen Release des Softwareprodukts auf einer Konfiguration ausführbar, die vor einer noch festzulegenden Anzahl von Jahren Standardkonfiguration war?	Nach Ablauf der Frist wird das Softwareprodukt in der dann aktuellen Version auf einem Gerät in der früheren Standardkonfiguration installiert. Dabei wird geprüft, ob Installation und Standardnutzungsszenario auf dem Gerät erfolgreich durchgeführt werden können. Das Vorgehen wird solange mit der jeweils nächst älteren Standardkonfiguration wiederholt, bis das Szenario nicht mehr durchgeführt werden kann, oder die Konfiguration erreicht ist, die vor einer noch festzulegenden Anzahl von Jahren Standardkonfiguration war.	Es wird notiert bis zu welcher ehemaligen Standardkonfiguration die aktuelle Version lauffähig ist. **Hinweis:** Hierdurch wird eine Obergrenze für das Alter der Standardkonfiguration festgelegt.

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
2.2	Plattformunabhängigkeit und Portabilität		Das Softwareprodukt wird mit Standardeinstellungen installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Herstellerangaben (kompatibel mit verschiedenen Betriebssystemen, Laufzeitumgebungen)	Es wird eine Recherche in der Produktdokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite durchgeführt um die Anforderungen des Produkts für die Benutzung auf verschiedenen Betriebssystemen, bzw. Laufzeitumgebungen festzustellen.	Die vorausgesetzten Betriebssysteme, bzw. Laufzeitumgebungen werden mit der mindestens benötigten Version notiert.
		b) Standardnutzungsszenario auf verschiedenen aktuell verbreiteten produktiven Systemumgebungen ausführen, dabei die Daten- und Softwareeinstellungsportabilität prüfen	Die Software wird auf den verschiedenen Systemumgebungen (z.B. Betriebssysteme) installiert und eine Augenscheinprüfung durchgeführt um festzustellen, ob zwischen den Umgebungen Abweichungen bei Funktionsumfang, Kompatibilität und Usability auftreten.	Es wird notiert ob (und welche) Unterschiede zwischen den Systemumgebungen beobachtet werden können. **Hinweis:** Entfällt ggf. abhängig von 2.2.a
2.3	**Hardwaresuffizienz**			
		a) Intertemporale Vergleiche mit folgenden Ergebnisstufen: 1. „sehr gut“: Die Versionswechsel haben bisher zu einer Verringerung der benötigten Hardwarekapazitäten geführt. 2. „gut“: Die Versionswechsel haben bisher nicht zu einer Erhöhung der benötigten Hardwarekapazitäten geführt. 3. „genügend“: Die Versionswechsel haben bisher zwar zu einer Erhöhung der benötigten Hardwarekapazitäten geführt, diese blieben jedoch im Rahmen des technisch bedingten Effizienzfortschritts, wie er sich in der zeitlichen Abfolge von Referenzsystemen ausdrückt. 4. „ungenügend“: Die Versionswechsel haben dazu geführt, dass die benötigten Hardwarekapazitäten schneller gewachsen sind als die technische Effizienz entsprechend der Abfolge der Referenzsysteme.	Die in 1.1.4 gemessenen benötigten Hardwarekapazitäten werden mit Messungen älterer Produktversionen verglichen. Dabei wird die Ergebnisstufe durch den Prüfer eingeschätzt.	Die Ergebnisstufen werden notiert. **Hinweis:** Entfällt bei erster Messung

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
3	**Nutzungsautonomie**			
3.1	**Transparenz und Interoperabilität**			
3.1.1	Transparenz der Datenformate und Datenportabilität	a) Überprüfung der Handbücher und technischen Datenblätter, ob Datenformate ausreichend dokumentiert sind b) Abgleich mit bekannten und offenen Standards	Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts, bzw. Augenscheinprüfung in den Optionen und der Hilfe-Funktionalität durchgeführt um festzustellen welche Datenformate von dem Produkt unterstützt werden und ob der Datenaustausch über offene Datenformate möglich ist. Anschließend wird für die Datenformate recherchiert, ob diese ausreichend dokumentiert sind. **Hinweis:** Ggf. existiert kein eindeutiges Datenaustauschformat.	Es wird notiert, welche Datenformate unterstützt werden und inwiefern diese offengelegt sind, bzw. den Datenaustausch mit anderen Softwareprodukten ermöglichen.
3.1.2	Transparenz und Interoperabilität der Programme		Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts, bzw. Augenscheinprüfung in den Optionen und der Hilfe-Funktionalität durchgeführt. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Wenn es APIs gibt: Überprüfung der Schnittstellendokumentation anhand der Dokumentation des Softwareprodukts und seiner APIs	Es wird geprüft, ob APIs existieren. Falls dies der Fall ist wird die Art der APIs recherchiert und nach Einschätzung des/der Prüfenden beurteilt, ob diese gut dokumentiert und erreichbar sind. Dabei orientiert sich der/die Prüfende an der Übersichtlichkeit der Strukturierung, der Einhaltung von Dokumentationsstandards, sowie dem Vorhandensein von Suchfunktionalität, Entwicklersupport, Support-Foren etc.	Es wird notiert ob APIs existieren und welche Funktionen diese bereitstellen. Zusätzlich wird notiert, ob die Schnittstellen gut dokumentiert sind.
		b) Ist der Quellcode vollständig offengelegt?	Es wird geprüft, ob der Quellcode heruntergeladen werden kann. Dazu können auch ggf. gängiger Sourcecodeverwaltungsplattformen wie GitHub, GitLab, Bitbucket, Sourceforge etc. durchsucht werden.	Es wird notiert, ob der Quellcode offengelegt ist. **Hinweis:** Auch Quellcode-Teile können Open Source sein. Diese werden dann auch notiert.
		b) Ist die Software unter einer Lizenz veröffentlicht, die es erlaubt, die Software weiterzuentwickeln?	Die Lizenzbedingungen des Produkts werden geprüft und die Rechte recherchiert, die die Lizenz einräumt.	Es wird notiert, welche Lizenz verwendet wird und ob diese es erlaubt die Software weiterzuentwickeln.
3.1.3	Kontinuität des Softwareproduktes		Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Pro-	

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
			dukts, bzw. Augenscheinprüfung in den Optionen und der Hilfe-Funktionalität durchgeführt. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Wie lang ist der Zeitraum, für den der Anbieter die zukünftige Unterstützung des Produkts mit Sicherheitsupdates garantiert?	Es wird geprüft, ob Angaben zum Unterstützungszeitraum des Produkts vorhanden sind. Ist dies nicht der Fall, wird eine Recherche zu den Unterstützungsintervallen durchgeführt. Dazu wird entweder das Versprechen des Herstellers herangezogen oder, falls nicht vorhanden, für letzte nicht mehr supportete Version festgestellt, wie viel Zeit zwischen Release und Ende des Supports vergangen ist. **Hinweis:** Gibt es mehrere Betriebssysteme, bzw. Laufzeitumgebungen, die das Produkt unterstützt, werden die Angaben zum Supportzeitraum für alle Systeme recherchiert.	Die Supportzeiträume werden zusammen mit der Art wie sie recherchiert wurden notiert.
		b) Reagiert der Hersteller zeitnah auf das Bekanntwerden von Sicherheitslücken?	Es wird eine Recherche zu bekannten, offenen und geschlossenen Sicherheitslücken des Produkts durchgeführt. Dabei wird geprüft ob Sicherheits-Lücken bei der entsprechenden Software bekannt sind und ob der Hersteller diese zeitnah behoben hat (z.B. mittels Web-Recherchen, CVSS Score von http://www.cvedetails.com/ etc.) und ob ein öffentlicher Issue-Tracker existiert. Abschließend schätzt der/die Prüfende ein, ob Hersteller zuverlässig auf Sicherheitslücken reagiert.	Die Ergebnisse der Recherche, sowie die Einschätzung des/der Prüfenden werden notiert.
		c) Kann der Nutzende durch Konfiguration die Updatehäufigkeit beeinflussen und dabei insbesondere zwischen Sicherheitsupdates und sonstigen Updates differenzieren?	Die Konfigurationsoptionen werden auf Update-Optionen geprüft. Falls dort keine entsprechenden Optionen vorhanden sind wird eine Suche in offizieller Dokumentation und ggf. Hilfe-Funktion. Zusätzlich wird geprüft, ob Sicherheitsupdates und sonstige Updates separat aktiviert oder deaktiviert werden können. **Hinweis:** Auch bereits während der Installation können entsprechende Optionen vorhanden sein.	Die Update-Verwaltung, verfügbare Optionen und Aussagen zur Möglichkeit der Differenzierung zwischen Sicherheitsupdates und sonstigen Updates werden notiert.
		d) Besteht die Möglichkeit, nur differenzielle Updates zu erhalten?	Das Softwareprodukt wird in einer älteren Version installiert und ein Softwareupdate durchgeführt. Dabei wird geprüft, ob das Produkt komplett neu her-	Es wird notiert ob das Produkt beim Update komplett heruntergeladen wird oder differenzielle Updates durchge-

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
			untergeladen werden muss, oder ob das Update differenziell erfolgt. Falls dies nicht ersichtlich ist wird eine Traffic-Messung durchgeführt und der gemessene Wert mit der Größe des Downloads des gesamten Produkts verglichen (vgl. 1.1.6.a)	führt werden. Im Falle der Überprüfung durch eine Traffic-Messung wird deren Ergebnis notiert.
3.1.4	Transparenz des Prozessmanagements		Das Softwareprodukt wird mit Standardeinstellungen installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Prüfung anhand der Installation und der Ausführung von Standardnutzungsmustern, welche Prozesse das Softwareprodukt automatisch startet und ob es darauf aufmerksam macht (es macht auf alles aufmerksam/auf einiges/macht nicht aufmerksam)	Es wird bei der Installation geprüft, ob auf den Start von Prozessen aufmerksam gemacht wird und ob Zusatzmodule mit installiert werden, die ggf. weitere Prozesse starten. Anschließend wird das Produkt gestartet und die dadurch geöffneten Prozesse im Prozessmanager des Betriebssystems beobachtet.	Es wird notiert, ob das Produkt auf Prozesse und Zusatzmodule aufmerksam macht, welche Prozesse gestartet werden und ob dies nur solche Prozesse sind auf die aufmerksam gemacht wurde
		b) Wenn das Softwareprodukt beim Systemstart automatisch gestartet wird („Autostart“): Macht es darauf aufmerksam, dass dies der Fall ist?	Es wird geprüft, ob bei der Installation darauf aufmerksam gemacht wird, ob die Software oder Untermodule beim Systemstart geöffnet werden. Anschließend wird die Autostart-Konfiguration des Betriebssystems auf neue Einträge durch die Installation überprüft. Falls Einträge angelegt wurden wird geprüft, ob während der Installation darauf aufmerksam gemacht wurde. Falls neue Autostarteinträge vorhanden sind, auf die nicht aufmerksam gemacht wurde wird ein Neustart des Betriebssystems durchgeführt und geprüft ob der Autostart der Software für Nutzer ersichtlich ist (z.B. Benachrichtigung oder geöffnetes Fenster)	Das Ergebnis der Beobachtungen wird notiert. **Hinweis:** Dieser Indikator ist ggf. bei Server-Softwareprodukten eine ausdrücklich erwünschte Funktionalität.
		c) Wenn der Nutzende eine Aktion durchführt, die als Beendigung des Programms aufgefasst werden kann, aber mindestens einer der Prozesse noch aktiv bleibt: Macht das Softwareprodukt darauf aufmerksam?	Bei Produkten mit graphischer Benutzeroberfläche werden alle Fenster mittels Schließkreuz geschlossen, Produkte ohne graphische Benutzeroberfläche werden durch einen entsprechenden Befehl (siehe Dokumentation) beendet. Dabei wird geprüft, ob eine Berichtigung angezeigt wird, dass Teile der Software im Hintergrund geöffnet bleiben. Falls keine Benachrichtigung angezeigt wird. Im Prozessmanager des Betriebssystems überprüft, ob alle Prozesse der Software beendet wurden.	Das Ergebnis der Beobachtungen wird notiert. **Hinweis:** Dieser Indikator ist ggf. bei Server-Softwareprodukten unzutreffend, da sie ggf. nicht per se beendet werden können (z.B. Content Management Systeme bestehen lediglich aus Dateien, die durch den Webserver ausgeführt, bzw. bereitgestellt werden).

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
3.2	**Deinstallierbarkeit**			
3.2.1	Deinstallierbarkeit der Programme	a) Deinstallation der Software und Vergleich mit dem Zustand vor der Installation, der gleich sein muss.	Es wird ein Black-Box Test wie folgt durchgeführt: Zunächst wird eine Kopie des standardisierten Speicherabbilds des SUT vor der Installation des Softwareproduktes erstellt. Anschließend wird das Produkt installiert, das Standardnutzungsszenario durchgeführt und das Softwareprodukt wieder (falls vorhanden, anhand der Angaben in der Dokumentation) deinstalliert. Von dem jetzigen Zustand des SUT wird erneut ein Speicherabbild erstellt und mit dem vor der Installation erstellten Abbild verglichen. Dadurch werden alle Dateien und Änderungen festgestellt, die durch die Software hinterlassen werden. Der/die Prüfende stellt dann fest, welche Dateien durch das Softwareprodukt erstellt wurden. **Hinweis:** Im Fall von Windows-Betriebssystemen wird außerdem die Registry nach verbleibenden Einträgen des Produktes durchsucht.	Es wird das Vorgehen bei der Deinstallation, sowie die verbliebenen Daten und ggf. weitere Dateien wie Registry- oder Datenbankeinträge notiert.
3.2.2	Löschbarkeit der Daten		Das Softwareprodukt wird mit Standardeinstellungen installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Ist der Zustand nach dem Löschen der nicht durch den Nutzer ausdrücklich gespeicherten Daten mit dem Zustand vor der Installation in relevanter Hinsicht identisch?	Es wird eine Augenscheinprüfung durchgeführt um festzustellen, ob und welche Daten von dem Softwareprodukt, neben den explizit vom Nutzer erzeugten Daten, angelegt werden (z.B. Browserverlauf, Cookies, "zuletzt geöffnete Dokumente" etc.) und ob diese transparent gemacht und gelöscht werden können.	Das Ergebnis der Augenscheinprüfung wird notiert.
		b) Macht das Softwareprodukt transparent, wo die explizit angelegten Daten gespeichert werden (Speicherort)?	Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts, bzw. Augenscheinprüfung in den Optionen und der Hilfe-Funktionalität durchgeführt um den Speicherort der vom Nutzer angelegten Dateien herauszufinden.	Es wird notiert, ob das Produkt den Speicherort transparent macht oder nicht.
		c) Wird der Nutzende dabei unterstützt, die auf entfernten Speicherkapazitäten angelegten Datenbestände zu löschen?	Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts, bzw. Augenscheinprüfung in den Optionen und der Hilfe-	Das Ergebnis der Recherche wird notiert.

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
			Funktionalität durchgeführt um festzustellen ob das Produkt den/die Nutzer/in beim Löschen von Daten auf entfernten Speicherkapazitäten unterstützt.	
3.3	**Wartungsfunktionen**			
3.3.1	Datenwiederherstellbarkeit		Das Softwareprodukt wird mit Standardeinstellungen installiert. Die Beurteilung erfolgt entlang der Indikatoren:	
		a) Macht der Hersteller dazu Angaben, und lassen sich diese im Test bestätigen?	Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts zu Angaben zur Datenwiederherstellbarkeit durchgeführt. Außerdem wird eine Augenscheinprüfung durchgeführt bei der durch gezielte Fehlerverursachung (z.B. Strom ausschalten, Prozess über Taskmanager beenden) das Verhalten des Softwareprodukts im Fehlerfall bewertet wird.	Das Ergebnis der Recherche und Augenscheinprüfung wird notiert.
		b) Ist der Zeitraum, in dem Änderungen an den Daten zwischengespeichert werden, parametrisierbar?	Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts, bzw. Augenscheinprüfung in den Optionen und der Hilfe-Funktionalität durchgeführt um festzustellen ob der Zeitraum für das automatische zwischenspeichern parametrisierbar ist.	Das Ergebnis der Recherche und Augenscheinprüfung wird notiert.
3.3.2	Selbstreparaturfähigkeit	a) Herstellerangaben und Überprüfung durch Test	Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts zu Angaben zur Selbstreparaturfähigkeit durchgeführt. Außerdem wird eine Augenscheinprüfung durchgeführt bei der durch gezielte Fehlerverursachung (z.B. entfernen einer Datei aus dem Installationsverzeichnis und erneutes Ausführen des Setups) geprüft ob eine Reparaturoption angeboten wird.	Das Ergebnis der Recherche und Augenscheinprüfung wird notiert.

#	Kriterium	Indikator	Operationalisierung	Darstellungsart
3.4	**Unabhängigkeit von Fremdressourcen**			
3.4.1	Offlinefähigkeit	a) Überprüfung durch Standardnutzungsszenario (Offline-Betrieb möglich/mit Einschränkungen möglich/unmöglich)	Das Softwareprodukt wird mit Standardeinstellungen installiert und sichergestellt, dass das Standardnutzungsszenario ohne Einschränkungen durchführbar ist. Anschließend wird die Netzwerkverbindung deaktiviert und das Standardnutzungsszenario erneut ausgeführt. Dabei wird das Verhalten der Software beobachtet.	Es wird notiert, ob der Offline-Betrieb möglich, mit Einschränkungen möglich oder unmöglich ist. **Hinweis:** Entfällt bei Server-Softwareprodukten.
3.5	**Qualität der Produktinformation**			
3.5.1	Verständlichkeit und Überschaubarkeit der Produktdokumentation, Lizenz- und Nutzungsbedingungen	a) Augenscheinprüfung durch Prüfende; Test mit tatsächlichen Nutzenden	Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts und der Hilfe-Funktionalität durchgeführt um festzustellen ob Dokumentation, Lizenz und Nutzungsbedingungen verständlich und übersichtlich sind. Für den Test mit den Nutzenden werden bestimmte Fragestellungen definiert (z.B. Deinstallation) und geprüft ob dieser Vorgang schnell in der Dokumentation gefunden wird und verständlich beschrieben ist.	Die Ergebnisse der Recherche und der Tests werden notiert.
3.5.2	Ressourcenrelevanz der Produktinformation	a) Qualitative Beurteilung der Vollständigkeit und Verständlichkeit	Es wird eine Recherche in der Dokumentation, dem Benutzerhandbuch oder der Hersteller- bzw. Produktwebseite des Produkts und der Hilfe-Funktionalität durchgeführt um festzustellen ob die Produktinformation alle Angaben enthalten, die die Nutzenden benötigen, um die Ressourcenbeanspruchung durch das Softwareprodukt gering zu halten und ob diese Angaben verständlich sind.	Das Ergebnis der Recherche wird notiert.
		b) Bezieht sich die Produktinformation auf die aktuelle Produktversion?	Es wird recherchiert, ob sich die in 3.5.2.a erfassten Angaben auf die aktuelle Version des Produkts beziehen.	Das Ergebnis der Recherche wird notiert.
		c) Augenscheinprüfung der Korrektheit der Angaben (Angaben sind schlüssig / eingeschränkt schlüssig / nicht schlüssig)	Es wird eine Augenscheinprüfung durchgeführt um festzustellen ob die in 3.5.2.a erfassten Angaben korrekt sind.	Das Ergebnis der Augenscheinprüfung wird notiert.

Anhang 4 Ergebnisse der Kriterienerfassung im Detail

Tabelle 20: Ergebnisse der Kriterienerfassung für die Fallbeispiele „Textverarbeitung (TVP)“ und „Browser (B)“ für die Fallbeispiele „Textverarbeitung“, „Browser“, „Content Management Systeme“ und „Datenbanken“

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
1	**Ressourceneffizienz**						
1.1	**Hardwareeffizienz**						
1.1.1	Empfohlene Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)	a) Empfohlene lokale Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems (Produkt aus Taktfrequenz, Anzahl Kerne, Busbreite)	k.A.	37,6 % (1 GHz oder schneller)	k.A.	k.A.	k.A.
		b) Empfohlener lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems	k.A.	25 % (1 GB)	k.A.	k.A.	k.A.
		c) Empfohlener lokaler Permanentspeicher laut Herstellerangaben in % des Permanentspeichers des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.
		d) Empfohlene Displayauflösung laut Herstellerangaben in % der Displayauflösung des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.
		e) Empfohlene Bandbreite für Netzzugang laut Herstellerangaben in % der Bandbreite des Referenzsystems	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		f) Empfohlene serverseitige Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		g) Empfohlener serverseitiger Arbeitsspeicher laut Herstellerangaben in % des serverseitigen Arbeitsspeichers des Referenzsystems	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		h) Empfohlener serverseitiger Permanentspeicher laut Herstellerangaben in % des serverseitigen Permanentspeichers des Referenzsystems	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
1.1.2	Minimale Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)	a) Minimale lokale Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems	37,6 % (1 GHz oder schneller)	k.A. (Pentium III, Athlon oder aktueller)	k.A. (Intel Pentium 4-Prozessor mit SSE2)	k.A. (Intel Pentium 4-Prozessor mit SSE2)	37,6 % (1 GHz oder schneller)
		b) Minimaler lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems	50 % (2 GB RAM)	6,4 % (256 MB RAM)	3,125 % (128 MB RAM)	12,5 % (512 MB RAM)	12,5 % (512 MB RAM)
		c) Minimaler lokaler Permanentspeicher laut Herstellerangaben in % des Permanentspeichers des Referenzsystems	0,9 % (3,0 GB)	0,5 % (1,5 GB)	0,03 % (100 MB)	0,1 % (200 MB)	0,036 % (120 MB)
		d) Minimale Displayauflösung laut Herstellerangaben in % der Displayauflösung des Referenzsystems	37,93 % (1024 x 768 Pixel)	37,93 % (1024 x 768 Pixel)	k.A.	k.A.	23,15 % (800 x 600 Pixel)
		e) Minimale Bandbreite für Netzzugang laut Herstellerangaben in % der Bandbreite des Referenzsystems	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		f) Minimale serverseitige Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		g) Minimaler serverseitiger Arbeitsspeicher laut Herstellerangaben in % des serverseitigen Arbeitsspeichers des Referenzsystems	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		h) Minimaler serverseitiger Permanentspeicher laut Herstellerangaben in % des serverseitigen Permanentspeichers des Referenzsystems	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
1.1.3	Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration	a) Messung der mittleren Prozessorauslastung im Leerlauf unter Standardkonfiguration (Differenz zum Grundverbrauch der Standardkonfiguration ohne das Softwareprodukt im gleichen Zeitraum)	4,06 %	1,73 %	0,761 %	9,109 %	12,299 %
		b) Messung der mittleren Arbeitsspeicherbelegung im Leerlauf unter Standardkonfiguration	2,09 %	1,4 %	0,9779 %	0 % (Abweichung zur Grundauslastung nicht messbar)	0,0825 %
		c) Messung der mittleren Permanentspeicherbelegung im Leerlauf unter Standardkonfiguration	0,223 MB/s	0,068 MB/s	0,069 MB/s	0,028 MB/s	0,009 MB/s

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
		d) Messung der mittleren beanspruchten Bandbreite für Netzzugang im Leerlauf unter Standardkonfiguration	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	0,0086 MB/s	0,0002 MB/s	0,0022 MB/s
		e) Messung der mittleren serverseitigen Prozessorauslastung im Leerlauf unter Standardkonfiguration	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		f) Messung der mittleren serverseitigen Arbeitsspeicherbelegung im Leerlauf unter Standardkonfiguration	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		g) Messung der mittleren serverseitigen Permanentspeicherbelegung im Leerlauf unter Standardkonfiguration	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
1.1.4	Hardware-Inanspruchnahme bei normaler Nutzung unter der Annahme einer Standardkonfiguration und eines Standardnutzungsszenarios	a) Messung der Prozessorarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	17.366,965 %s	4.727,954 %s	5.487,986 %s	12.556,107 %s	8.220,312 %s
		b) Messung der Arbeitsspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	1.869,952 %s	69,438 %s	5.018,386 %s	6.461,971 %s	7.656,674 %s
		c) Messung der Permanentspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	192,252 MByte	553,747 MByte	112,124 MByte	399,162 MByte	199,367 MByte
		d) Messung der übertragenen Datenmenge für Netzzugang bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	44,771 MByte	75,63 MByte	58,3 MByte
		e) Messung der serverseitigen Prozessorarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		f) Messung der serverseitigen Arbeitsspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.
		g) Messung der serverseitigen Permanentspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	entfällt, da lokale Anwendung	entfällt, da lokale Anwendung	k.A.	k.A.	k.A.

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
1.1.5	Sparsame Hardwarenutzung durch Anpassbarkeit und Unterstützung der Nutzenden bei der Anpassung des Softwareprodukts	a) Geschieht die Minimierung der beanspruchten Kapazitäten automatisch und/oder gibt es bei der Installation der Software entsprechende Optionen? (Skala: ja/nein)	ja	ja	nein	nein	nein
		b) Falls die Nutzenden eine entsprechende Wahl treffen, ist die Entscheidung für oder gegen Installationsoptionen später jederzeit revidierbar (Skala: ja/nein)	ja	ja	nicht zutreffend	nicht zutreffend	nicht zutreffend
		c) Black-Box-Test: Können hardwareintensive Software-Module abgeschaltet werden? (Skala: können abgeschaltet werden/können nicht abgeschaltet werden)	können abgeschaltet werden (z.B. Autosave und Rechtschreibprüfung)	können abgeschaltet werden (z.B. Autosave und Rechtschreibprüfung)	können nicht abgeschaltet werden	können nicht abgeschaltet werden	können nicht abgeschaltet werden
		d) Ist es ohne Nachteil möglich, vorübergehend oder dauerhaft nicht benötigte Peripheriegeräte abzuschalten bzw. gar nicht bereitzustellen? (Skala: Können vorübergehend und dauerhaft abgeschaltet werden/können nur vorübergehend abgeschaltet wer-den/können nicht abgeschaltet werden)	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden
		e) Werden nach der Installation die Dateien gelöscht, die nur zur Installation benötigt werden?	nein	nein	nein	nein	nein
1.1.6	Online-Auslieferung	a) Sind eine Online-Auslieferung und ein Online-Update der Software möglich?	ja	ja	ja	ja	ja
		b) Wird unterstützt, dass das Softwareprodukt und seine Updates in der anwendenden Organisation zentral abgelegt werden?	ja	ja	ja	ja	nein
1.2	**Energieeffizienz**						
		a) Messung der zur Ausführung des Standardnutzungsszenarios unter Standardkonfiguration auf dem lokalen Gerät verbrauchten Energie	3,6 Wh	0.93 Wh	0,66 Wh (gemessen)	1,95 Wh (gemessen)	0,91 Wh (gemessen)

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
		b) Schätzung der durch die Datenübertragung im Netz verbrauchten Energie aufgrund des bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration erzeugten Datenverkehrs (Verwendung einer aktuellen Schätzung für die Energieintensität des Netzes in kWh/GB basierend auf aktueller Fachliteratur, wenn nötig differenziert nach Zugangsnetzen)	0 Wh	0 Wh	0,593 Wh (Schätzwert)	1,002 Wh (Schätzwert)	0,772 Wh (Schätzwert)
		c) Messung der durch die entfernte Speicherung und Verarbeitung in Servern verbrauchten Energie bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration (falls Messung nicht möglich, Schätzung mit Hilfe durchschnittlicher Faktoren für die Energieintensität von Rechenzentrumsdienstleistungen basierend auf aktueller Fachliteratur)	0 Wh	0 Wh	0,00256 Wh (Schätzwert)	0,00433 Wh (Schätzwert)	0,00334 Wh (Schätzwert)
1.3	**Ressourcenmanagement**						
1.3.1	Anpassung der beanspruchten Kapazitäten an den Bedarf	a) Verfügt das Softwareprodukt über unterschiedliche Modi, deren Wechsel sich messbar auf den Energieverbrauch auswirkt?	nein	nein	Datensparmodus vorhanden	nein	nein
		b) Wechselt das Softwareprodukt dynamisch jeweils in einen sparsameren Modus, wenn das möglich ist? (z. B. Schlafmodus)	ja, Schlafmodus nach 26 Sekunden	ja, Schlafmodus nach 18 Sekunden	ja, Schlafmodus nach 40 Sekunden	ja, Schlafmodus nach 65 Sekunden	ja, Schlafmodus nach 30 Sekunden
		c) Falls für die Software spezielle Einstellungen für einen sparsamen Betriebsmodus vorgenommen werden müssen, sind diese Konfigurationsmöglichkeiten zentral und allgemein verständlich zusammengefasst?	nein	nein	nein	nein	nein
1.3.2	Anpassung des Bedarfs an die verfügbaren Kapazitäten	a) Wechselt das Softwareprodukt in einen sparsameren Modus, wenn das Angebot an Hardwarekapazitäten oder Energie sich verringert, ohne dass Fehler oder Datenverluste auftreten? (keine Einschränkung,	nein	nein	nein	nein	nein

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
		langsamere Ausführung, Fehler in der Ausführung)					
		b) Ist die Software auch bei aktiviertem Energiemanagement der darunterliegenden Systemschichten oder der verbundenen Clientsysteme uneingeschränkt funktional nutzbar? (ja, uneingeschränkt nutzbar / eingeschränkt nutzbar / nicht nutzbar)	ja	ja	Größtenteils keine Einschränkungen. Fenster, Tabs und Formulareinträge bleiben erhalten. Downloads müssen neu gestartet werden.	Keine Einschränkungen. Fenster, Tabs und Formulareinträge bleiben erhalten. Downloads werden (wenn möglich) automatisch fortgesetzt.	Keine Einschränkungen. Fenster, Tabs und Formulareinträge bleiben erhalten. Downloads werden (wenn möglich) automatisch fortgesetzt.
1.3.3	Ressourcenschonende Standardeinstellungen	a) Einschätzung des Prüfers, ob Standardeinstellungen des Softwareprodukts so gewählt sind, dass sie das Ziel der Ressourcenschonung mitberücksichtigen	nein	nein	nein	nein	nein
1.3.4	Feedback zur Beanspruchung von Hardwarekapazitäten und Energie	a) Werden die beanspruchten Hardwarekapazitäten, der Datenfluss und die Energieverbräuche angezeigt? (funktionsspezifisch vorhanden mit Vorschlägen zur Ressourceneinsparung, funktionsspezifisch vorhanden, für das Gesamtprodukt vorhanden, nicht vorhanden)	nicht vorhanden	nicht vorhanden	Eingeschränkt für das Gesamtprodukt vorhanden: Tool zur Überwachung der Speicher-belegung, CPU-Auslastung und Netzwerknutzung	Eingeschränkt für das Gesamtprodukt vorhanden: Tools zur Überwachung der Netzwerkauslastung und Speicherbelegung	Eingeschränkt für das Gesamtprodukt vorhanden: Tools zur Überwachung der Netzwerkauslastung und Speicherbelegung
		b) Einschätzung des Prüfers, ob Anzeige korrekt (korrekt / nicht korrekt)	nicht vorhanden	nicht vorhanden	ja	ja	ja
2	**Potenzielle Hardware-Nutzungsdauer**						
2.1	**Abwärtskompatibilität**						
		a) Zunächst Herstelleraussage (Hardware, ältere Betriebssysteme, ältere Frameworks), da für zurückliegende Jahre keine Standardkonfigurationen definiert sind	Auf 8 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 9 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 8 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 8 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 8 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
		b) Wenn seit der ersten Anwendung dieses Kriteriums auf das Softwareprodukt ausreichend Zeit verstrichen ist, so dass man das Standardnutzungsszenario auch auf früheren Standardkonfigurationen ausführen kann: Ist das Standardnutzungsszenario mit dem aktuellen Release des Softwareprodukts auf einer Konfiguration ausführbar, die vor einer noch festzulegenden Anzahl von Jahren Standardkonfiguration war?	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung
2.2	**Plattformunabhängigkeit und Portabilität**						
		a) Herstellerangaben (kompatibel mit verschiedenen Betriebssystemen, Laufzeitumgebungen)	- Windows ab 7 - Windows Server ab 2008R2 - OS X ab 10.10 Yosemite	- Windows ab XP - Windows Server ab 2008 - OS X ab 10.8 Mountain Lion - Diverse Linux-Distributionen	- Windows ab 7 - OS X ab 10.9 Mavericks - Diverse Linux-Distributionen	- Windows ab XP - Windows Server ab 2003 - OS X ab 10.9 Mavericks - Diverse Linux-Distributionen	- Windows ab 7 - Windows Server ab 2008R2
		b) Standardnutzungsszenario auf verschiedenen aktuell verbreiteten produktiven Systemumgebungen ausführen, dabei die Daten- und Softwareeinstellungsportabilität prüfen	Teilweise andere Visualisierung unter Windows und MacOS	Identische Funktionalität	Identische Funktionalität	Identische Funktionalität	Nicht verfügbar
2.3	**Hardwaresuffizienz**						
		a) Intertemporale Vergleiche mit folgenden Ergebnisstufen: 1. „sehr gut“: Die Versionswechsel haben bisher zu einer Verringerung der benötigten Hardwarekapazitäten geführt. 2. „gut“: Die Versionswechsel haben bisher nicht zu einer Erhöhung der benötigten Hardwarekapazitäten geführt. 3. „genügend“: Die Versionswechsel haben bisher zwar zu einer Erhöhung der benötigten Hardwarekapazitäten geführt, diese blieben jedoch im Rahmen des technisch bedingten Effizienzfortschritts, wie er sich	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
		in der zeitlichen Abfolge von Referenzsystemen ausdrückt. 4. „ungenügend“: Die Versionswechsel haben dazu geführt, dass die benötigten Hardwarekapazitäten schneller gewachsen sind als die technische Effizienz entsprechend der Abfolge der Referenzsysteme.					
3	**Nutzungsautonomie**						
3.1	**Transparenz und Interoperabilität**						
3.1.1	Transparenz der Datenformate und Datenportabilität	a) Überprüfung der Handbücher und technischen Datenblätter, ob Datenformate ausreichend dokumentiert sind	Datenformate sind ausreichend dokumentiert.	Datenformate sind ausreichend dokumentiert.	Datenformate sind ausreichend dokumentiert.	Datenformate sind ausreichend dokumentiert.	Datenformate sind ausreichend dokumentiert.
		b) Abgleich mit bekannten und offenen Standards	Standards des proprietären Formats wurden offengelegt. Es gibt jedoch teilweise Zweifel an der Offenheit	Datenformate sind offene Standards	Datenformate sind offene Standards	Datenformate sind offene Standards	Datenformate sind offene Standards
3.1.2	Transparenz und Interoperabilität der Programme	a) Wenn es APIs gibt: Überprüfung der Schnittstellendokumentation anhand der Dokumentation des Softwareprodukts und seiner APIs	Umfangreich und übersichtlich dokumentierte APIs vorhanden. Entwicklersupport vorhanden.	Umfangreich und übersichtlich dokumentierte APIs vorhanden. Entwicklersupport vorhanden.	Umfangreich und übersichtlich dokumentierte APIs vorhanden. Community-Support.	Umfangreich und übersichtlich dokumentierte APIs vorhanden. Community-Support.	Umfangreich und übersichtlich dokumentierte APIs vorhanden. Entwicklersupport vorhanden.
		b) Ist der Quellcode vollständig offengelegt?	nein	ja	nein	ja	nein
		b) Ist die Software unter einer Lizenz veröffentlicht, die es erlaubt, die Software weiterzuentwickeln?	nein	ja (MPL, LGPL v. 3, GPL v. 3+)	Nein, große Teile werden aber in einem open source Projekt verfügbar gemacht.	ja (MPL, GPL, LGPL)	nein
3.1.3	Kontinuität des Softwareproduktes	a) Wie lang ist der Zeitraum, für den der Anbieter die zukünftige Unterstützung des Produkts mit Sicherheitsupdates garantiert?	ca. 11 Jahre	Nicht zutreffend. Kontinuierliche, kostenlose Veröffentlichung von Updates.	Nicht zutreffend. Kontinuierliche, kostenlose Veröffentlichung von Updates.	Nicht zutreffend. Kontinuierliche, kostenlose Veröffentlichung von Updates.	Keine genauen Angaben vorhanden
		b) Reagiert der Hersteller zeitnah auf das Bekanntwerden von Sicherheitslücken?	ja	Hersteller reagiert teilweise erst spät auf Sicherheitslücken	Hersteller reagiert teilweise erst spät auf Sicherheitslücken	Hersteller reagiert teilw. erst spät auf Sicherheitslücken	Hersteller reagiert teilweise erst spät auf Sicherheitslücken

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
		c) Kann der Nutzende durch Konfiguration die Updatehäufigkeit beeinflussen und dabei insbesondere zwischen Sicherheitsupdates und sonstigen Updates differenzieren?	Updates können nur generell aktiviert oder deaktiviert werden.	nein	nein	Updates können nur generell aktiviert oder deaktiviert werden.	Updates können nur generell aktiviert oder deaktiviert werden.
		d) Besteht die Möglichkeit, nur differenzielle Updates zu erhalten?	Neue Versionen sind i.a. nicht differenziell, Updates innerhalb einer Version sind i.a. differenziell.	nein	ja	ja	Neue Versionen sind i.a. nicht differenziell, Updates innerhalb einer Version sind i.a. differenziell.
3.1.4	Transparenz des Prozessmanagements	a) Prüfung anhand der Installation und der Ausführung von Standardnutzungsmustern, welche Prozesse das Softwareprodukt automatisch startet und ob es darauf aufmerksam macht (es macht auf alles aufmerksam/auf einiges/macht nicht aufmerksam)	Macht auf einiges aufmerksam.	Macht auf alles aufmerksam.	Macht auf einiges aufmerksam.	Macht auf alles aufmerksam.	Macht auf alles aufmerksam.
		b) Wenn das Softwareprodukt beim Systemstart automatisch gestartet wird („Autostart“): Macht es darauf aufmerksam, dass dies der Fall ist?	Keine Einträge in Autostart, aber: In der "Aufgabenplanung" von Windows werden drei neue Aufgaben angelegt über die der Nutzer nicht in Kenntnis gesetzt wird.	Keine Einträge in Autostart.	Keine Einträge in Autostart, aber: In der "Aufgabenplanung" von Windows werden zwei neue Aufgaben angelegt über die der Nutzer nicht in Kenntnis gesetzt wird.	Keine Einträge in Autostart.	Keine genauen Angaben vorhanden
		c) Wenn der Nutzende eine Aktion durchführt, die als Beendigung des Programms aufgefasst werden kann, aber mindestens einer der Prozesse noch aktiv bleibt: Macht das Softwareprodukt darauf aufmerksam?	Das Programm wird vollständig geschlossen.	Das Programm wird vollständig geschlossen.	Das Programm wird vollständig geschlossen.	Das Programm wird vollständig geschlossen.	Das Programm wird vollständig geschlossen.
3.2	**Deinstallierbarkeit**						
3.2.1	Deinstallierbarkeit der Programme	a) Deinstallation der Software und Vergleich mit dem Zustand vor der Installation, der gleich sein muss.	Es verbleiben 14 leere Ordner, eine Datei (Größe vernachlässigbar) sowie über 100 Einträge in der Windows Registry.	Zustand vor der Installation und nach der Deinstallation identisch.	Es verbleiben 29 Dateien (Größe vernachlässigbar).	Es verbleiben 19 Dateien (Größe vernachlässigbar).	Durchführung des Tests nicht möglich.

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
3.2.2	Löschbarkeit der Daten	a) Ist der Zustand nach dem Löschen der nicht durch den Nutzer ausdrücklich gespeicherten Daten mit dem Zustand vor der Installation in relevanter Hinsicht identisch?	Generelle Löschfunktion nicht vorhanden, Produkt macht teilweise auf nicht explizit angelegte Daten aufmerksam.	Generelle Löschfunktion nicht vorhanden, Produkt macht teilweise auf nicht explizit angelegte Daten aufmerksam.	Teilweise Funktionen zum Löschen vorhanden, privater Modus vorhanden, Produkt macht teilweise auf nicht explizit angelegte Daten aufmerksam.	Teilweise Funktionen zum Löschen vorhanden, privater Modus vorhanden, Produkt macht teilweise auf nicht explizit angelegte Daten aufmerksam.	Teilweise Funktionen zum Löschen vorhanden, privater Modus vorhanden, Produkt macht teilweise auf nicht explizit angelegte Daten aufmerksam.
		b) Macht das Softwareprodukt transparent, wo die explizit angelegten Daten gespeichert werden (Speicherort)?	ja	ja	nur teilweise (z.B. nicht für Lesezeichen, Chronik, etc.)	nur teilweise (z.B. nicht für Lesezeichen, Chronik, etc.)	nur teilweise (z.B. nicht für Lesezeichen, Chronik, etc.)
		c) Wird der Nutzende dabei unterstützt, die auf entfernten Speicherkapazitäten angelegten Datenbestände zu löschen?	nein	nein	nein	nein	nein
3.3	**Wartungsfunktionen**						
3.3.1	Datenwiederherstellbarkeit	a) Macht der Hersteller dazu Angaben, und lassen sich diese im Test bestätigen?	Funktionalität vorhanden und im Test bestätigt.	Funktionalität vorhanden und im Test bestätigt.	Funktionalität vorhanden und im Test bestätigt.	Funktionalität vorhanden und im Test bestätigt.	Funktionalität vorhanden und im Test bestätigt.
		b) Ist der Zeitraum, in dem Änderungen an den Daten zwischengespeichert werden, parametrisierbar?	ja	ja	nein	nein	nein
3.3.2	Selbstreparaturfähigkeit	a) Herstellerangaben und Überprüfung durch Test	Funktionalität vorhanden	Funktionalität vorhanden (Installationsmedium wird benötigt)	Funktionalität nicht vorhanden	Funktionalität nicht vorhanden	Durchführung des Tests nicht möglich.
3.4	**Unabhängigkeit von Fremdressourcen**						
3.4.1	Offlinefähigkeit	a) Überprüfung durch Standardnutzungsszenario (Offline-Betrieb möglich/mit Einschränkungen möglich/unmöglich)	Offline-Betrieb möglich	Offline-Betrieb möglich	Offline-Betrieb unmöglich	Offline-Betrieb unmöglich	Offline-Betrieb unmöglich
3.5	**Qualität der Produktinformation**						
3.5.1	Verständlichkeit und Überschaubarkeit der Produktdokumentation, Lizenz- und Nutzungsbedingungen	a) Augenscheinprüfung durch Prüfende; Test mit tatsächlichen Nutzenden	Angaben überschaubar und verständlich.	Angaben in großen Teilen überschaubar und verständlich.	Angaben überschaubar und verständlich.	Angaben überschaubar und verständlich.	Angaben überschaubar und verständlich.

#	Kriterium	Indikator	TVP1	TVP2	B1	B2	B3
3.5.2	Ressourcenrelevanz der Produktinformation	a) Qualitative Beurteilung der Vollständigkeit und Verständlichkeit	Keine Informationen über Möglichkeiten zur Senkung der Ressourcenbeanspruchung.	Keine Informationen über Möglichkeiten zur Senkung der Ressourcenbeanspruchung	Hinweise zur Speicherreduzierung	Hinweise zur Speicherreduzierung	Keine Informationen über Möglichkeiten zur Senkung der Ressourcenbeanspruchung
		b) Bezieht sich die Produktinformation auf die aktuelle Produktversion?	ja	ja	ja	ja	ja
		c) Augenscheinprüfung der Korrektheit der Angaben (Angaben sind schlüssig / eingeschränkt schlüssig / nicht schlüssig)	nicht zutreffend, da keine Angaben	nicht zutreffend, da keine Angaben	ja	ja	nicht zutreffend, da keine Angaben

Tabelle 21: Ergebnisse der Kriterienerfassung für die Fallbeispiele „Content Management Systeme (CMS)“ und „Datenbanken (DB)“

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
1	**Ressourcen-effizienz**							
1.1	**Hardwareeffizienz**							
1.1.1	Empfohlene Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)	a) Empfohlene lokale Rechenleistung laut Herstellerangaben in % d er Rechenleistung des Referenzsystems (Produkt aus Taktfrequenz, Anzahl Kerne, Busbreite)	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		b) Empfohlener lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		c) Empfohlener lokaler Permanentspeicher laut Herstellerangaben in % des Permanentspeichers des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		d) Empfohlene Displayauflösung laut Herstellerangaben in % der Displayauflösung des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		e) Empfohlene Bandbreite für Netzzugang laut Herstellerangaben in % der Bandbreite des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		f) Empfohlene serverseitige Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		g) Empfohlener serverseitiger Arbeitsspeicher laut Herstellerangaben in % des serverseitigen Arbeitsspeichers des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		h) Empfohlener serverseitiger Permanentspeicher laut Herstellerangaben in % des serverseitigen Permanentspeichers des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
1.1.2	Minimale Systemvoraussetzungen und resultierende Hardwareanforderungen (inkl. Peripheriegeräte)	a) Minimale lokale Rechenleistung laut Herstellerangaben in % der Rechenleistung des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		b) Minimaler lokaler Arbeitsspeicher laut Herstellerangaben in % des Arbeitsspeichers des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		c) Minimaler lokaler Permanentspeicher laut Herstellerangaben in % des Permanentspeichers des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		d) Minimale Displayauflösung laut Herstellerangaben in % der Displayauflösung des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	37,93 % (1024 x 768 Pixel)
		e) Minimale Bandbreite für Netzzugang laut Herstellerangaben in % der Bandbreite des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.
		f) Minimale serverseitige Rechenleistung laut Her-	k.A.	k.A.	k.A.	k.A.	k.A.	k.A.

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		stellerangaben in % der Rechenleistung des Referenzsystems						
		g) Minimaler serverseitiger Arbeitsspeicher laut Herstellerangaben in % des serverseitigen Arbeitsspeichers des Referenzsystems	k.A.	k.A.	0,001 % (256 MB)	k.A.	k.A.	50 % (4 GB)
		h) Minimaler serverseitiger Permanentspeicher laut Herstellerangaben in % des serverseitigen Permanentspeichers des Referenzsystems	k.A.	k.A.	k.A.	k.A.	k.A.	3,125 % (10 GB)
1.1.3	Hardware-Auslastung im Leerlauf unter der Annahme einer Standardkonfiguration	a) Messung der mittleren Prozessorauslastung im Leerlauf unter Standardkonfiguration (Differenz zum Grundverbrauch der Standardkonfiguration ohne das Softwareprodukt im gleichen Zeitraum)	9,109 %	9,109 %	9,109 %	entfällt, da Serversystem	entfällt, da Serversystem	entfällt, da Serversystem
		b) Messung der mittleren Arbeitsspeicherbelegung im Leerlauf unter Standardkonfiguration	0 % (Abweichung zur Grundauslastung nicht messbar)	0 % (Abweichung zur Grundauslastung nicht messbar)	0 % (Abweichung zur Grundauslastung nicht messbar)	entfällt, da Serversystem	entfällt, da Serversystem	entfällt, da Serversystem
		c) Messung der mittleren Permanentspeicherbelegung im Leerlauf unter Standardkonfiguration	0,028 MB/s	0,028 MB/s	0,028 MB/s	entfällt, da Serversystem	entfällt, da Serversystem	entfällt, da Serversystem
		d) Messung der mittleren beanspruchten Bandbreite für Netzzugang im Leerlauf unter Standardkonfiguration	0 MB/s	0 MB/s	0 MB/s	0 MB/s	0 MB/s	0 MB/s
		e) Messung der mittleren serverseitigen Prozessorauslastung im Leerlauf	0,005 %	0,0069 %	0,0044 %	0 % (Abweichung zur Grundauslas-	0 % (Abweichung zur Grundauslastung nicht mess-	0 % (Abweichung zur Grundauslas-

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		unter Standardkonfiguration				tung nicht messbar)	bar)	tung nicht messbar)
		f) Messung der mittleren serverseitigen Arbeitsspeicherbelegung im Leerlauf unter Standardkonfiguration	4,4576 %	2,2285 %	4,6412 %	0 % (Abweichung zur Grundauslastung nicht messbar)	0 % (Abweichung zur Grundauslastung nicht messbar)	35,43 %
		g) Messung der mittleren serverseitigen Permanentspeicherbelegung im Leerlauf unter Standardkonfiguration	0 MB/s	0 MB/s	0 MB/s	0 MB/s	0 MB/s	0 MB/s
1.1.4	Hardware-Inanspruchnahme bei normaler Nutzung unter der Annahme einer Standardkonfiguration und eines Standardnutzungsszenarios	a) Messung der Prozessorarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	4.094,405 %s	2.801,137 %s	4.079,559 %s	entfällt, da Serversystem	entfällt, da Serversystem	entfällt, da Serversystem
		b) Messung der Arbeitsspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	631,451 %s	845,858 %s	1.313,191 %s	entfällt, da Serversystem	entfällt, da Serversystem	entfällt, da Serversystem
		c) Messung der Permanentspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	32,465 MByte	3,807 MByte	26,100 MByte	entfällt, da Serversystem	entfällt, da Serversystem	entfällt, da Serversystem
		d) Messung der übertragenen Datenmenge für Netzzugang bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	8.668,209 MByte	1.529,961 MByte	1.631,802 MByte	6.909,072 MByte	7.252,695 MByte	6.976,531 MByte
		e) Messung der serverseitigen Prozessorarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	3.952,3 %s	5.048,2 %s	2.453,2 %s	8.321,7 %s	5.494,7 %s	7.087,4 %s

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		f) Messung der serverseitigen Arbeitsspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	4.530,9 %s	3.165,4 %s	5.029,2 %s	24.154,05 %s	3.338,33 %s	16.906,15 %s
		g) Messung der serverseitigen Permanentspeicherarbeit bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration	714,255 MByte	19,181 MByte	59,032 MByte	2.054,60 MByte	1.030,50 MByte	504,511 MByte
1.1.5	Sparsame Hardwarenutzung durch Anpassbarkeit und Unterstützung der Nutzenden bei der Anpassung des Softwareprodukts	a) Geschieht die Minimierung der beanspruchten Kapazitäten automatisch und/oder gibt es bei der Installation der Software entsprechende Optionen? (Skala: ja/nein)	ja	nein	ja	ja	ja	ja
		b) Falls die Nutzenden eine entsprechende Wahl treffen, ist die Entscheidung für oder gegen Installationsoptionen später jederzeit revidierbar (Skala: ja/nein)	ja	nicht zutreffend	ja	ja	ja	ja
		c) Black-Box-Test: Können hardwareintensive Software-Module abgeschaltet werden? (Skala: können abgeschaltet werden/können nicht abgeschaltet werden)	können nicht abgeschaltet werden	können nicht abgeschaltet werden	können nicht abgeschaltet werden	können abgeschaltet werden (z.B. Indizierung)	können abgeschaltet werden (z.B. Indizierung)	können abgeschaltet werden (z.B. Indizierung)
		d) Ist es ohne Nachteil möglich, vorübergehend oder dauerhaft nicht benötigte Peripheriegeräte abzuschalten bzw. gar nicht bereitzustellen?	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden	Können vorübergehend und dauerhaft abgeschaltet werden

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		(Skala: Können vorübergehend und dauerhaft abgeschaltet werden/können nur vorübergehend abgeschaltet wer-den/können nicht abgeschaltet werden)						
		e) Werden nach der Installation die Dateien gelöscht, die nur zur Installation benötigt werden?	nein	nein	nein	nein	nein	nein
1.1.6	Online-Auslieferung	a) Sind eine Online-Auslieferung und ein Online-Update der Software möglich?	ja	ja	ja	ja	ja	ja
		b) Wird unterstützt, dass das Softwareprodukt und seine Updates in der anwendenden Organisation zentral abgelegt werden?	ja	ja	ja	ja	ja	ja
1.2	**Energieeffizienz**							
		a) Messung der zur Ausführung des Standardnutzungsszenarios unter Standardkonfiguration auf dem lokalen Gerät verbrauchten Energie	0,73 Wh (gemessen)	0,608 Wh (gemessen)	0,657 Wh (gemessen)	entfällt, da Serversystem	entfällt, da Serversystem	entfällt, da Serversystem
		b) Schätzung der durch die Datenübertragung im Netz verbrauchten Energie aufgrund des bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration erzeugten Datenverkehrs (Verwendung einer aktuellen Schätzung für die Energieintensität des	114,132 Wh (Schätzwert)	20,265 Wh (Schätzwert)	21,614 Wh (Schätzwert)	91,515 Wh (Schätzwert)	96,066 Wh (Schätzwert)	92,408 Wh (Schätzwert)

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		Netzes in kWh/GB basierend auf aktueller Fachliteratur, wenn nötig differenziert nach Zugangsnetzen)						
		c) Messung der durch die entfernte Speicherung und Verarbeitung in Servern verbrauchten Energie bei Ausführung des Standardnutzungsszenarios unter Standardkonfiguration (falls Messung nicht möglich, Schätzung mit Hilfe durchschnittlicher Faktoren für die Energieintensität von Rechenzentrumsdienstleistungen basierend auf aktueller Fachliteratur)	10,316 Wh (gemessen)	0,741 Wh (gemessen)	0,288 Wh (gemessen)	1,603 Wh (gemessen)	1,044 Wh (gemessen)	1,355 Wh (gemessen)
1.3	**Ressourcen-management**							
1.3.1	Anpassung der beanspruchten Kapazitäten an den Bedarf	a) Verfügt das Softwareprodukt über unterschiedliche Modi, deren Wechsel sich messbar auf den Energieverbrauch auswirkt?	nein	nein	nein	nein	nein	nein
		b) Wechselt das Softwareprodukt dynamisch jeweils in einen sparsameren Modus, wenn das möglich ist? (z. B. Schlafmodus)	nein	nein	nein	nein	nein	nein
		c) Falls für die Software spezielle Einstellungen für einen sparsamen Betriebsmodus vorgenommen werden müssen, sind diese Konfigurationsmög-	nein	nein	nein	nein	nein	nein

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		lichkeiten zentral und allgemein verständlich zusammengefasst?						
1.3.2	Anpassung des Bedarfs an die verfügbaren Kapazitäten	a) Wechselt das Softwareprodukt in einen sparsameren Modus, wenn das Angebot an Hardwarekapazitäten oder Energie sich verringert, ohne dass Fehler oder Datenverluste auftreten? (keine Einschränkung, langsamere Ausführung, Fehler in der Ausführung)	Keine Einschränkungen	Keine Einschränkungen	Keine Einschränkungen	Keine Einschränkungen	Keine Einschränkungen	Keine Einschränkungen
		b) Ist die Software auch bei aktiviertem Energiemanagement der darunterliegenden Systemschichten oder der verbundenen Clientsysteme uneingeschränkt funktional nutzbar? (ja, uneingeschränkt nutzbar / eingeschränkt nutzbar / nicht nutzbar)	nicht zutreffend	nicht zutreffend	nicht zutreffend	ja	ja	ja
1.3.3	Ressourcenschonende Standardeinstellungen	a) Einschätzung des Prüfers, ob Standardeinstellungen des Softwareprodukts so gewählt sind, dass sie das Ziel der Ressourcenschonung mitberücksichtigen	nein	nein	nein	ja	ja	ja
1.3.4	Feedback zur Beanspruchung von Hardwarekapazitäten und Energie	a) Werden die beanspruchten Hardwarekapazitäten, der Datenfluss und die Energieverbräuche angezeigt? (funktionsspezifisch vorhanden mit Vorschlägen zur Ressourceneinsparung, funktions-	nein	nein	nein	Eingeschränkt für das Gesamtprodukt vorhanden	nein	Funktionsspezifisch vorhanden bei Nutzung eines Zusatztools.

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		spezifisch vorhanden, für das Gesamtprodukt vorhanden, nicht vorhanden)						
		b) Einschätzung des Prüfers, ob Anzeige korrekt (korrekt / nicht korrekt)	nicht vorhanden	nicht vorhanden	nicht vorhanden	ja	nicht vorhanden	ja
2	**Potenzielle Hardware-Nutzungsdauer**							
2.1	**Abwärtskompatibilität**							
		a) Zunächst Herstelleraussage (Hardware, ältere Betriebssysteme, ältere Frameworks), da für zurückliegende Jahre keine Standardkonfigurationen definiert sind	Auf 15 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 10 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 10 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 22 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 14 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)	Auf 8 Jahre altem System lauffähig (Aufnahmezeitpunkt: 04/2017)
		b) Wenn seit der ersten Anwendung dieses Kriteriums auf das Softwareprodukt ausreichend Zeit verstrichen ist, so dass man das Standardnutzungsszenario auch auf früheren Standardkonfigurationen ausführen kann: Ist das Standardnutzungsszenario mit dem aktuellen Release des Softwareprodukts auf einer Konfiguration ausführbar, die vor einer noch festzulegenden Anzahl von Jahren Standardkonfiguration war?	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
2.2	**Plattformunabhängigkeit und Portabilität**							
		a) Herstellerangaben (kompatibel mit verschiedenen Betriebssystemen, Laufzeitumgebungen)	- Windows Server ab 2003 - Diverse Linux Distributionen	- Windows Server ab 2008 - Diverse Linux Distributionen	- Windows Server ab 2003 - Diverse Linux Distributionen	- Windows ab 95 - Windows Server ab 2003 - OS X ab 10.5 Leopard - Alle gängigen Linux-Distributionen	- Windows ab 2000 SP4 - Windows Server ab 2003 - OS X ab 10.5 Leopard - Alle gängigen Linux-Distributionen	- Windows ab 7 - Windows Server ab 2008
		b) Standardnutzungsszenario auf verschiedenen aktuell verbreiteten produktiven Systemumgebungen ausführen, dabei die Daten- und Softwareeinstellungsportabilität prüfen	Identische Funktionalität	Identische Funktionalität	Identische Funktionalität	Identische Funktionalität	Identische Funktionalität	Identische Funktionalität
2.3	**Hardware-suffizienz**							
		a) Intertemporale Vergleiche mit folgenden Ergebnisstufen: 1. „sehr gut“: Die Versionswechsel haben bisher zu einer Verringerung der benötigten Hardwarekapazitäten geführt. 2. „gut“: Die Versionswechsel haben bisher nicht zu einer Erhöhung der benötigten Hardwarekapazitäten geführt. 3. „genügend“: Die Versionswechsel haben bisher	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung	entfällt, da erste Messung

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		zwar zu einer Erhöhung der benötigten Hardware-kapazitäten geführt, diese blieben jedoch im Rahmen des technisch bedingten Effizienzfortschritts, wie er sich in der zeitlichen Ab-folge von Referenzsyste-men ausdrückt. 4. „ungenügend“: Die Versionswechsel haben dazu geführt, dass die benötigten Hardwareka-pazitäten schneller ge-wachsen sind als die tech-nische Effizienz entspre-chend der Abfolge der Referenzsysteme.						
3	**Nutzungsauto-nomie**							
3.1	**Transparenz und Interoperabilität**							
3.1.1	Transparenz der Datenformate und Datenportabilität	a) Überprüfung der Hand-bücher und technischen Datenblätter, ob Daten-formate ausreichend dokumentiert sind	Datenformate sind ausreichend doku-mentiert.	Datenformate sind ausreichend doku-mentiert.	Datenformate sind ausreichend doku-mentiert.	Datenformate sind ausrei-chend doku-mentiert.	Datenformate sind ausreichend doku-mentiert.	Datenformate sind ausrei-chend doku-mentiert.
		b) Abgleich mit bekannten und offenen Standards	Datenformate sind offene Standards	Datenformate sind offene Standards	Datenformate sind offene Standards	Datenformate sind offene Standards	Datenformate sind offene Standards	Datenformate sind offene Standards
3.1.2	Transparenz und Interoperabilität der Programme	a) Wenn es APIs gibt: Überprüfung der Schnitt-stellendokumentation anhand der Dokumentati-on des Softwareprodukts und seiner APIs	Umfangreich und übersichtlich doku-mentierte APIs vor-handen. Community-Support.	Umfangreich und übersichtlich doku-mentierte APIs vor-handen. Community-Support.	Umfangreich und übersichtlich doku-mentierte APIs vor-handen. Entwickler-support vorhanden.	Umfangreich und übersicht-lich dokumen-tierte APIs vor-handen. Com-munity-Support.	Umfangreich und übersichtlich do-kumentierte APIs vorhanden. Com-munity-Support.	Umfangreich und übersicht-lich dokumen-tierte APIs vor-handen. Ent-wicklersupport vorhanden.
		b) Ist der Quellcode voll-ständig offengelegt?	ja	ja	ja	ja	ja	nein

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		b) Ist die Software unter einer Lizenz veröffentlicht, die es erlaubt, die Software weiterzuentwickeln?	ja (GPL v2+)	ja (GPL v2+)	ja (GPL)	ja (GPL v2)	ja (eigene Lizenz, kompatibel mit GPL)	nein
3.1.3	Kontinuität des Softwareproduktes	a) Wie lang ist der Zeitraum, für den der Anbieter die zukünftige Unterstützung des Produkts mit Sicherheitsupdates garantiert?	Nicht zutreffend. Kontinuierliche, kostenlose Veröffentlichung von Updates.	Nicht zutreffend. Kontinuierliche, kostenlose Veröffentlichung von Updates.	3 Jahre für LTS und kontinuierliche, kostenlose Veröffentlichung von Updates.	Nicht zutreffend. Kontinuierliche, kostenlose Veröffentlichung von Updates.	5 Jahre (ohne Garantie), kontinuierliche, kostenlose Veröffentlichung von Updates.	unbegrenzt
		b) Reagiert der Hersteller zeitnah auf das Bekanntwerden von Sicherheitslücken?	Hersteller reagiert teilweise erst spät auf Sicherheitslücken	Hersteller reagiert teilweise erst spät auf Sicherheitslücken	Hersteller reagiert teilweise erst spät auf Sicherheitslücken	Hersteller reagiert teilweise erst spät auf Sicherheitslücken	Hersteller reagiert teilweise erst spät auf Sicherheitslücken	Hersteller reagiert teilweise erst spät auf Sicherheitslücken
		c) Kann der Nutzende durch Konfiguration die Updatehäufigkeit beeinflussen und dabei insbesondere zwischen Sicherheitsupdates und sonstigen Updates differenzieren?	ja (keine automatischen Updates)	ja (keine automatischen Updates)	ja (keine automatischen Updates)	ja (keine automatischen Updates)	ja (keine automatischen Updates)	ja (keine automatischen Updates)
		d) Besteht die Möglichkeit, nur differenzielle Updates zu erhalten?	ja	nein	nein	ja	Neue Versionen sind i.a. nicht differenziell, Updates innerhalb einer Version sind i.a. differenziell.	konnte nicht festgestellt werden
3.1.4	Transparenz des Prozessmanagements	a) Prüfung anhand der Installation und der Ausführung von Standardnutzungsmustern, welche Prozesse das Softwareprodukt automatisch startet und ob es darauf aufmerksam macht (es macht auf alles aufmerksam/auf einiges/macht	macht nicht aufmerksam	macht nicht aufmerksam	macht nicht aufmerksam	Macht auf alles aufmerksam	Macht auf alles aufmerksam	Macht auf einiges aufmerksam

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		nicht aufmerksam)						
		b) Wenn das Softwareprodukt beim Systemstart automatisch gestartet wird („Autostart“): Macht es darauf aufmerksam, dass dies der Fall ist?	nicht zutreffend	nicht zutreffend	nicht zutreffend	nicht zutreffend	nicht zutreffend	nicht zutreffend
		c) Wenn der Nutzende eine Aktion durchführt, die als Beendigung des Programms aufgefasst werden kann, aber mindestens einer der Prozesse noch aktiv bleibt: Macht das Softwareprodukt darauf aufmerksam?	nicht zutreffend	nicht zutreffend	nicht zutreffend	nicht zutreffend	nicht zutreffend	nicht zutreffend
3.2	**Deinstallierbarkeit**							
3.2.1	Deinstallierbarkeit der Programme	a) Deinstallation der Software und Vergleich mit dem Zustand vor der Installation, der gleich sein muss.	Zustand vor der Installation und nach der Deinstallation identisch.	Zustand vor der Installation und nach der Deinstallation identisch.	Zustand vor der Installation und nach der Deinstallation identisch.	Zustand vor der Installation und nach der Deinstallation identisch.	Zustand vor der Installation und nach der Deinstallation identisch.	Zustand vor der Installation und nach der Deinstallation identisch.
3.2.2	Löschbarkeit der Daten	a) Ist der Zustand nach dem Löschen der nicht durch den Nutzer ausdrücklich gespeicherten Daten mit dem Zustand vor der Installation in relevanter Hinsicht identisch?	Generelle Löschfunktion nicht vorhanden, Produkt macht teilweise auf nicht explizit angelegte Daten aufmerksam.	Generelle Löschfunktion nicht vorhanden, Produkt macht teilweise auf nicht explizit angelegte Daten aufmerksam.	Teilweise Funktionen zum Löschen vorhanden, Produkt macht teilweise auf nicht explizit angelegte Daten aufmerksam.	nicht zutreffend	nicht zutreffend	nicht zutreffend
		b) Macht das Softwareprodukt transparent, wo die explizit angelegten Daten gespeichert werden (Speicherort)?	ja	ja	ja	nicht zutreffend	nicht zutreffend	nicht zutreffend

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
		c) Wird der Nutzende dabei unterstützt, die auf entfernten Speicherkapazitäten angelegten Datenbestände zu löschen?	ja	ja	ja	nicht zutreffend	nicht zutreffend	nicht zutreffend
3.3	**Wartungsfunktionen**							
3.3.1	Datenwiederherstellbarkeit	a) Macht der Hersteller dazu Angaben, und lassen sich diese im Test bestätigen?	Funktionalität nicht vorhanden.	Funktionalität vorhanden und im Test bestätigt.	Funktionalität nicht vorhanden.	nicht zutreffend	nicht zutreffend	nicht zutreffend
		b) Ist der Zeitraum, in dem Änderungen an den Daten zwischengespeichert werden, parametrisierbar?	nicht vorhanden	nein	nicht vorhanden	nicht zutreffend	nicht zutreffend	nicht zutreffend
3.3.2	Selbstreparaturfähigkeit	a) Herstellerangaben und Überprüfung durch Test	Funktionalität vorhanden	Funktionalität nicht vorhanden	Funktionalität nicht vorhanden	Funktionalität nicht vorhanden	Funktionalität nicht vorhanden	Funktionalität nicht vorhanden
3.4	**Unabhängigkeit von Fremdressourcen**							
3.4.1	Offlinefähigkeit	a) Überprüfung durch Standardnutzungsszenario (Offline-Betrieb möglich/mit Einschränkungen möglich/unmöglich)	Offline-Betrieb unmöglich	Offline-Betrieb unmöglich	Offline-Betrieb unmöglich	Offline-Betrieb unmöglich	Offline-Betrieb unmöglich	Offline-Betrieb unmöglich
3.5	Qualität der Produktinformation							
3.5.1	Verständlichkeit und Überschaubarkeit der Produktdokumentation, Lizenz- und Nutzungsbedingungen	a) Augenscheinprüfung durch Prüfende; Test mit tatsächlichen Nutzenden	Angaben überschaubar und verständlich.	Angaben überschaubar und verständlich.	Angaben in großen Teilen überschaubar und verständlich.	Angaben überschaubar und verständlich.	Angaben überschaubar und verständlich.	Angaben überschaubar und verständlich.

#	Kriterium	Indikator	CMS1	CMS2	CMS3	DB1	DB2	DB3
3.5.2	Ressourcenrelevanz der Produktinformation	a) Qualitative Beurteilung der Vollständigkeit und Verständlichkeit	Keine Informationen über Möglichkeiten zur Senkung der Ressourcenbeanspruchung	Keine Informationen über Möglichkeiten zur Senkung der Ressourcenbeanspruchung	Keine Informationen über Möglichkeiten zur Senkung der Ressourcenbeanspruchung	Hinweise zur Effizienzsteigerung vorhanden.	Hinweise zur Effizienzsteigerung vorhanden.	Hinweise zur Effizienzsteigerung vorhanden.
		b) Bezieht sich die Produktinformation auf die aktuelle Produktversion?	ja	ja	größtenteils ja	ja	ja	ja
		c) Augenscheinprüfung der Korrektheit der Angaben (Angaben sind schlüssig / eingeschränkt schlüssig / nicht schlüssig)	nicht zutreffend, da keine Angaben	nicht zutreffend, da keine Angaben	nicht zutreffend, da keine Angaben	ja	ja	ja

Anhang 5 XML-Schema „ssd-info“


```xml
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="ufoplan-ssd-kriterien">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Produkt">
          <xs:complexType>
            <xs:sequence>
              <xs:element type="xs:string" name="ProduktBezeichnung"/>
              <xs:element type="xs:string" name="ProduktVersion"/>
              <xs:element type="xs:string"  name="ProduktHersteller"/>
              <xs:element type="xs:string" name="SoftwareArchitektur"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="MessungPersonelleAngaben">
          <xs:complexType>
            <xs:sequence>
              <xs:element type="xs:string" name="MessungDatum"/>
              <xs:element name="Messender">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element type="xs:string" name="MessenderName"/>
                    <xs:element type="xs:string" name="MessenderMail"/>
                    <xs:element type="xs:string" name="MessenderSonst"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="MessenderInstitut">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element type="xs:string" name="InstitutBezeichnung"/>
                    <xs:element type="xs:string" name="InstitutAdresse"/>
                    <xs:element type="xs:string" name="InstitutWebsite"/>
                    <xs:element type="xs:string" name="InstitutSonst"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element type="xs:string" name="MessungBemerkung"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="MessungTechnischeAngaben">
          <xs:complexType>
            <xs:sequence>
              <xs:element type="xs:string" name="Messgeraet"/>
              <xs:element type="xs:string" name="Abtastfrequenz"/>
              <xs:element type="xs:string" name="SzenarioLaenge"/>
              <xs:element type="xs:string" name="Stichprobenumfang"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="SystemUnderTestDetails">
          <xs:complexType>
            <xs:sequence>
              <xs:element type="xs:string" name="sutSystem"/>
              <xs:element type="xs:string" name="sutBetriebssystem"/>
              <xs:element type="xs:string" name="sutKonfiguration"/>
              <xs:element type="xs:string" name="sutMainboard"/>
              <xs:element type="xs:string" name="sutProzessor"/>
              <xs:element type="xs:string" name="sutCache"/>
              <xs:element type="xs:string" name="sutRAM"/>
              <xs:element type="xs:string" name="sutGrafikkarte"/>
              <xs:element type="xs:string" name="sutSSD"/>
              <xs:element type="xs:string" name="sutHDD"/>
              <xs:element type="xs:string" name="sutNetzwerk"/>
              <xs:element type="xs:string" name="sutStromversorgung"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="Ressourceneffizienz">
```

```
<xs:complexType>
  <xs:sequence>
    <xs:element name="Hardwareeffizienz">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="MinimaleSystemvoraussetzungen">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element type="xs:string" name="IndikatorID"/>
                      <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                      <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                      <xs:element type="xs:string" name="IndikatorErgebnis"/>
                      <xs:element type="xs:string" name="IndikatorEinheit"/>
                      <xs:element type="xs:string" name="IndikatorBemerkung"/>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="HWAuslastungLeerlauf">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element type="xs:string" name="IndikatorID"/>
                      <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                      <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                      <xs:element type="xs:string" name="IndikatorErgebnis"/>
                      <xs:element type="xs:string" name="IndikatorEinheit"/>
                      <xs:element type="xs:string" name="IndikatorBemerkung"/>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="HWInanspruchnahmeNormal">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element type="xs:string" name="IndikatorID"/>
                      <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                      <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                      <xs:element type="xs:string" name="IndikatorErgebnis"/>
                      <xs:element type="xs:string" name="IndikatorEinheit"/>
                      <xs:element type="xs:string" name="IndikatorBemerkung"/>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Energieeffizienz">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="EnergieeffizienzInd">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element type="xs:string" name="IndikatorID"/>
                      <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                      <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                      <xs:element type="xs:string" name="IndikatorErgebnis"/>
                      <xs:element type="xs:string" name="IndikatorEinheit"/>
```

```
                        <xs:element type="xs:string" name="IndikatorBemerkung"/>
                      </xs:sequence>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="HWNutzungsdauer">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Abwaertskompatibilitaet">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="AbwaertskompatibilitaetInd">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                    <xs:complexType>
                      <xs:sequence>
                        <xs:element type="xs:string" name="IndikatorID"/>
                        <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                        <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                        <xs:element type="xs:string" name="IndikatorErgebnis"/>
                        <xs:element type="xs:string" name="IndikatorEinheit"/>
                        <xs:element type="xs:string" name="IndikatorBemerkung"/>
                      </xs:sequence>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Plattformunabhaengigkeit">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="PlattformunabhaengigkeitPortabiliaet">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                    <xs:complexType>
                      <xs:sequence>
                        <xs:element type="xs:string" name="IndikatorID"/>
                        <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                        <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                        <xs:element type="xs:string" name="IndikatorErgebnis"/>
                        <xs:element type="xs:string" name="IndikatorEinheit"/>
                        <xs:element type="xs:string" name="IndikatorBemerkung"/>
                      </xs:sequence>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="Hardwaresuffizienz">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="HardwaresuffizienzInd">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                    <xs:complexType>
                      <xs:sequence>
                        <xs:element type="xs:string" name="IndikatorID"/>
                        <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                        <xs:element type="xs:string" name="IndikatorBeschreibung"/>
```

```xml
                        <xs:element type="xs:string" name="IndikatorErgebnis"/>
                        <xs:element type="xs:string" name="IndikatorEinheit"/>
                        <xs:element type="xs:string" name="IndikatorBemerkung"/>
                      </xs:sequence>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="Nutzungsautonomie">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="TransparenzInteroperabilitaet">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="TransparenzDaten">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                    <xs:complexType>
                      <xs:sequence>
                        <xs:element type="xs:string" name="IndikatorID"/>
                        <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                        <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                        <xs:element type="xs:string" name="IndikatorErgebnis"/>
                        <xs:element type="xs:string" name="IndikatorEinheit"/>
                        <xs:element type="xs:string" name="IndikatorBemerkung"/>
                      </xs:sequence>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="TransparenzInteroperabilitaetInd">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                    <xs:complexType>
                      <xs:sequence>
                        <xs:element type="xs:string" name="IndikatorID"/>
                        <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                        <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                        <xs:element type="xs:string" name="IndikatorErgebnis"/>
                        <xs:element type="xs:string" name="IndikatorEinheit"/>
                        <xs:element type="xs:string" name="IndikatorBemerkung"/>
                      </xs:sequence>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="Kontinuitaet">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                    <xs:complexType>
                      <xs:sequence>
                        <xs:element type="xs:string" name="IndikatorID"/>
                        <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                        <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                        <xs:element type="xs:string" name="IndikatorErgebnis"/>
                        <xs:element type="xs:string" name="IndikatorEinheit"/>
                        <xs:element type="xs:string" name="IndikatorBemerkung"/>
                      </xs:sequence>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
```

```xml
          </xs:complexType>
        </xs:element>
        <xs:element name="Deinstallierbarkeit">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="DeinstallierbarkeitInd">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element type="xs:string" name="IndikatorID"/>
                          <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                          <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                          <xs:element type="xs:string" name="IndikatorErgebnis"/>
                          <xs:element type="xs:string" name="IndikatorEinheit"/>
                          <xs:element type="xs:string" name="IndikatorBemerkung"/>
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="UnabhaengigkeitFremdressourcen">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Offlinefaehigkeit">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element type="xs:string" name="IndikatorID"/>
                          <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                          <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                          <xs:element type="xs:string" name="IndikatorErgebnis"/>
                          <xs:element type="xs:string" name="IndikatorEinheit"/>
                          <xs:element type="xs:string" name="IndikatorBemerkung"/>
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="Produktinformation">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Verstaendlichkeit">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Indikator" maxOccurs="unbounded" minOccurs="0">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element type="xs:string" name="IndikatorID"/>
                          <xs:element type="xs:string" name="IndikatorBezeichnung"/>
                          <xs:element type="xs:string" name="IndikatorBeschreibung"/>
                          <xs:element type="xs:string" name="IndikatorErgebnis"/>
                          <xs:element type="xs:string" name="IndikatorEinheit"/>
                          <xs:element type="xs:string" name="IndikatorBemerkung"/>
                        </xs:sequence>
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
```

```
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```