TEKNOLOGISK INSTITUT

DigitalLead.

Grøn softwareudvikling og datahåndtering

Innovationsmuligheder for danske IKT-virksomheder

Udarbejdet af
Teknologisk Institut, Erhverv og Samfund
Kongsvang Allé 29, 8000 Aarhus C

Februar 2025

Forfattere
Andreas Bjerre Lunkeit
Christina Løth Andersen
Melissa Joelsen
Asbjørn Veilskov Friis

Kontaktperson
Seniorkonsulent Andreas Bjerre Lunkeit
Mail: ablu@teknologisk.dk, Tlf.: 7220 1553

ISBN: 978-87-91461-88-0

Indhold

Resumé

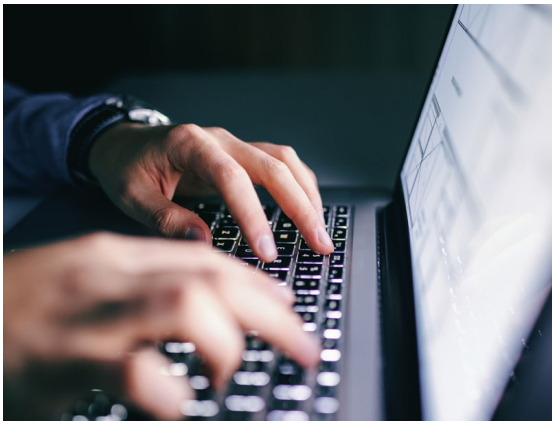
Informations- og kommunikationsteknologi (IKT) spiller en afgørende rolle i det moderne samfund, men den voksende anvendelse af digitale enheder og løsninger medfører en betydelig belastning af klimaet. IKT-sektoren er ansvarlig for mellem 2 og 4 procent af de globale drivhusgasemissioner, en andel, der kan stige til op mod 14 procent i 2040.[1]

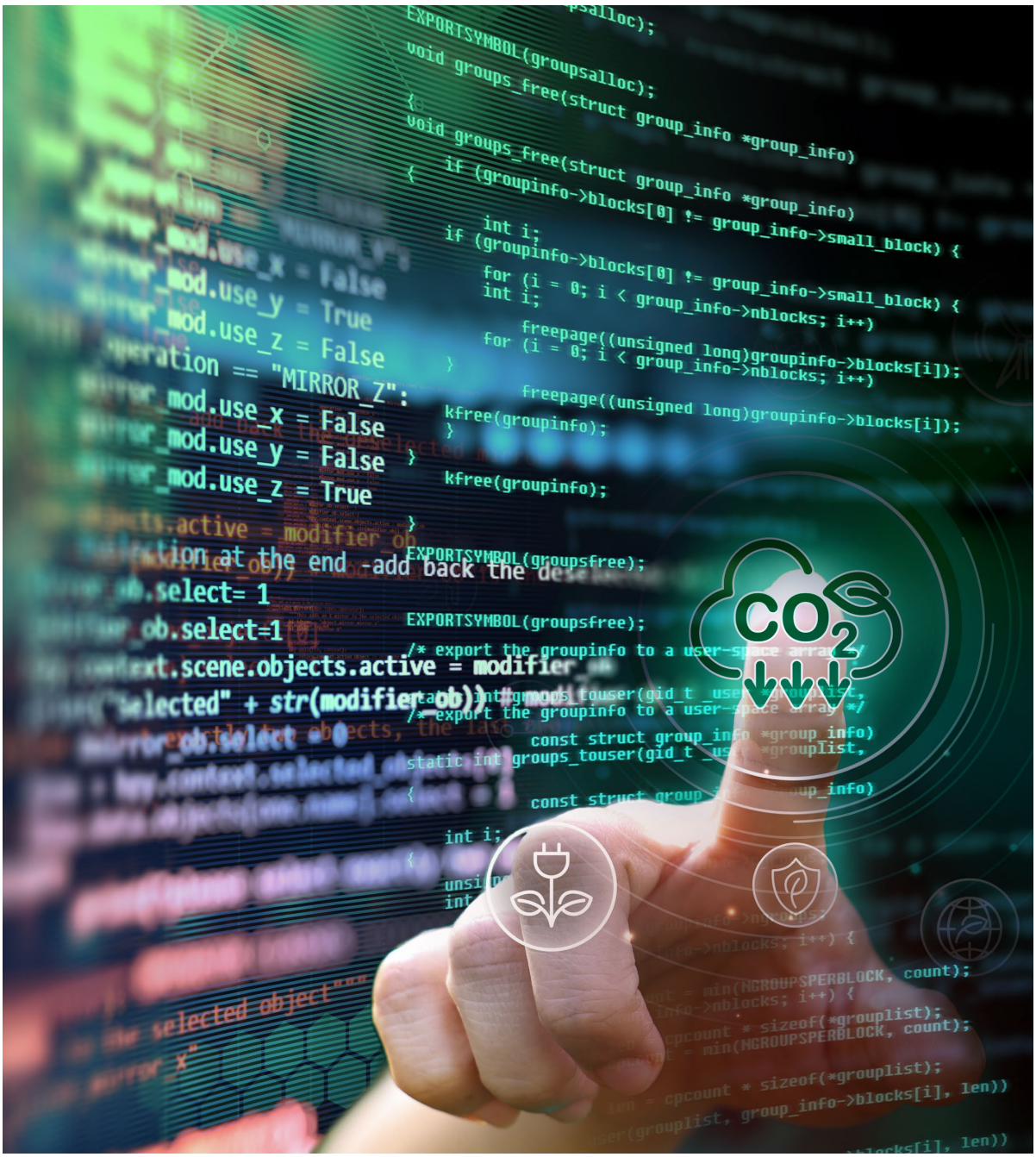
Sektorens klimabelastning stammer bl.a. fra forbruget af ressourcer og energi ifm. produktion af ny hardware og affaldshåndtering af udtjent hardware. Den største kilde er dog energiforbruget ifm. anvendelsen af digitale enheder og digital infrastruktur. Når fx computere, mobiltelefoner og servere bruger konventionel strøm fra fossile brændstoffer, er de kilder til CO_2-udledninger. En tilgang til at reducere klimaaftrykket fra IKT er derfor at arbejde med energioptimering af den software, der kører på vores forskellige digitale enheder, og styrer, hvordan de håndterer data.

Grøn softwareudvikling indebærer anvendelsen af bæredygtige designprincipper og kodningsteknikker til at reducere energiforbruget. Dette kan fx omfatte optimering af softwarearkitektur eller eliminering af unødvendig kode. Grøn datahåndtering beskæftiger sig med energioptimering af processer vedr. behandling, lagring og overførsel af data. Her kan der fx være tale om komprimering af data for at begrænse netværkstrafik eller anvendelsen af cloudløsninger, der i højere grad kører på vedvarende energi og er mere effektive end lokale datacentre på grund af dynamisk ressourcefordeling.

DigitalLead – Danmarks nationale klynge for digitale teknologier – ønsker at fremme viden inden for grøn softwareudvikling og datahåndtering for at styrke danske IKT-virksomheders konkurrenceevne og miljøprofil. Der er stor opmærksomhed på bæredygtighed i vores samfund, og bl.a. klimamål og miljøreguleringer bidrager til, at IKT-virksomhederne i højere grad skal levere produkter, der kan hjælpe kunder med at begrænse eller reducere deres klimabelastning.

På denne baggrund har Teknologisk Institut udarbejdet en rapport med fokus på, hvordan virksomheder, der udvikler, sælger og rådgiver om softwareløsninger, konkret kan arbejde med at gøre disse løsninger mere energieffektive. Rapporten er baseret på desk research og en række supplerende interviews med fire forskere og to større IT-virksomheder. Den drøfter vigtige elementer for en mere klimabevidst virksomhedspraksis på software- og dataområdet og præsenterer en bred vifte af principper og specifikke teknikker inden for grøn softwareudvikling og datahåndtering.





Danske IKT-virksomheder har betydelige muligheder på dette område, som må forventes at spille en voksende rolle i de kommende år. Rapporten præsenterer derfor også anbefalinger i form af en køreplan mod grøn softwareudvikling og datahåndtering. Den fremhæver bl.a. viden om relevante problemstillinger og muligheder for at bidrage til løsninger som en forudsætning for det videre arbejde med energieffektivisering på software- og dataområdet.

Virksomhederne bør derudover også sætte sig konkrete mål, sådan at der i samspil mellem udviklere, sælgere og kunder fx kan arbejdes strategisk med at energioptimere et eller flere softwareprojekter. Endelig er der behov for implementering og opfølgning, fx ved hjælp af de grønne kodeteknikker og bæredygtige tilgange til datahåndtering, der uddybes i denne rapport.

> **IKT anses for at være ansvarlig for mellem 2 og 4 procent af de globale drivhusgasemissioner – hvilket er omtrent på samme størrelse som flytrafikkens klimaaftryk.**

Kapitel 1

Indledning

Informations- og kommunikationsteknologi (IKT) spiller en central og voksende rolle i samfundet. I dag er der flere mobiltelefoner end mennesker på jorden,[2] mens antallet af forbundne enheder og sensorer, der udgør det såkaldte Internet of Things (IoT), er vokset fra ca. 10 milliarder i 2019 til 18,8 milliarder i 2024.[3] Dertil kommer vores stadig stigende behov for mere netværksinfrastruktur og flere datacentre, fordi vi producerer, transmitterer og forbruger voksende mængder data. Det er derfor ikke overraskende, at IKT anses for at være ansvarlig for mellem 2 og 4 procent af de globale drivhusgasemissioner[4] – hvilket er omtrent på samme størrelse som flytrafikkens klimaaftryk.[5]

De IKT-relaterede emissioner skyldes primært hardware. Der er bl.a. miljøomkostninger forbundet med produktionen af ny hardware og affaldshåndtering af udtjent hardware. Den største kilde til IKT's emissioner er dog energiforbruget i brugsfasen. Medmindre computere, mobiltelefoner, servere osv. udelukkende kører på vedvarende energi, er deres brug en kilde til CO_2-udledninger.

For at reducere klimaaftrykket fra IKT i drift, er der – ud over mindre forbrug eller større udbredelse af vedvarende energi – primært to muligheder. For det første kan man arbejde med energieffektivisering af hardware og sørge for, at energiforbruget er mindre ved samme ydeevne. Her har markante fremskridt bidraget til at begrænse stigningen i energiforbruget fra IKT på trods af den voksende efterspørgsel efter computerkraft. For det andet kan man energioptimere softwaren, der foretager beregninger og styrer data igennem kommunikationsnetværk. Sidstnævnte er genstand for denne rapport, der beskæftiger sig med innovationsmuligheder inden for grøn softwareudvikling og datahåndtering.

Med det voksende fokus på bæredygtighed i samfundet, fx i form af klimamål og miljøreguleringer, stiger behovet samtidig for softwareudvikling og datahåndtering med mindre strømforbrug fra hardware. IKT-virksomheder, der udvikler software, bliver derfor i stigende grad nødt til at udvikle mere bæredygtige løsninger og produkter, som bl.a. kan hjælpe kunder med at reducere deres klimaaftryk. Det kan fx være mere energieffektiv software, der er udviklet vha. grønne kodningsteknikker, eller energioptimerede cloudløsninger til lagring og håndtering af data.

DigitalLead, som er Danmarks nationale klynge for digitale teknologier, ønsker at belyse innovationsmuligheder inden for grøn softwareudvikling og datahåndtering. Formålet er at styrke danske IKT-virksomheders konkurrenceevne og miljøprofil. Der er knap 16.000 virksomheder i Danmark, der arbejder med softwareudvikling og datahåndtering,[6] så potentialet for at understøtte grønne tiltag på området er stort.

Teknologisk Institut har udarbejdet et idékatalog over teknologiske muligheder for udviklingen af energioptimeret softwareudvikling og datahåndtering. De grønne principper og teknikker, der præsenteres i denne rapport, er baseret på desk research og en række kvalita-

tive interviews med nøgleaktører, herunder to større IT-virksomheder og fire forskere inden for bæredygtighed i IKT. Indholdet i idékataloget skal inspirere IKT-virksomheder til mere konkret at arbejde med grøn softwareudvikling og datahåndtering.

Det er vigtigt at understrege, at rapportens emne på mange måder befinder sig på et tidligt stadie. Der er forskning og fællesskabsdrevne initiativer inden for grøn softwareudvikling og datahåndtering (se fx Green Software Foundation), men der mangler evidens for effekten af relevante tiltag i praksis.[7] Det betyder også, at de tilgange og teknikker, der præsenteres i rapporten, ikke nødvendigvis er universelle og helstøbte løsninger, men snarere muligheder for grønne tiltag, som kan prøves af og arbejdes videre med. Typisk vil anvendeligheden af bestemte principper eller teknikker afhænge af det enkelte projekt og individuelle krav.

1.1 Overblik og begrebsafklaring

I forlængelse af ovenstående fremhæver denne rapport en række muligheder for energioptimering af software og håndtering af data. Figur 1 giver et overblik over centrale principper for grøn softwareudvikling og datahåndtering, der udfoldes og belyses nærmere i kapitlerne 3 og 4. Som vist i figuren er en klimabevidst tilgang en forudsætning for arbejdet. It-konsulenter, softwareudviklere og andre relevante faggrupper skal have en grundlæggende forståelse af klimaaftrykket af deres produkter og være bevidste om, at de aktivt kan bidrage til at reducere det. Kapitel 2 udfolder denne pointe.

Rapportens to hovedbegreber – softwareudvikling og datahåndtering – defineres som følger:

- **Softwareudvikling** dækker over forskellige aktiviteter med det formål at designe, distribuere og vedligeholde computersoftware. Der kan grundlæggende skelnes mellem systemsoftware (fx operativsystemer og værktøjer til hardwarehåndtering), programmeringssoftware (værktøjer til at skabe kode, såsom teksteditorer og compilere) samt applikationer, der hjælper slutbrugere med at løse opgaver (alt fra skriveprogrammer og medieafspillere til web- og mobilbaserede apps).[8]
- **Datahåndtering** omhandler praksisser, processer og værktøjer til indsamling, organisering, lagring og behandling af data. Formålet med datahåndtering er typisk, at data er korrekt organiseret, let tilgængelige og lagret på en sikker måde gennem hele livscyklussen. Dette kan opnås ved brug af en bred vifte af teknologier, herunder fx databaser, datavarehuse, datacentre, cloudløsninger samt redskaber til analyse, integration, kvalitetssikring og kryptering af data.[9]

For at skelne mellem de to begreber dækker softwareudvikling over processen med at designe, kode, teste og vedligeholde computersoftware, mens datahåndtering fokuserer på at organisere, lagre og hente data på en effektiv og sikker måde. Det skal fremhæves, at der er stort overlap mellem begreberne, primært fordi data er en kernekomponent i software.

Set i dette lys er opdelingen af grønne tiltag i kategorierne softwareudvikling og datahåndtering primært analytisk. Mange af de teknikker til grøn softwareudvikling, der omtales i denne rapport, vil have en effekt på et givent systems datahåndtering, mens tiltag til grøn datahåndtering ofte omfatter tilpasninger af software. Når rapporten alligevel bruger denne opdeling, er det for at gøre det nemmere at kategorisere og drøfte forskellige måder, hvorpå softwareudvikling på den ene side og datahåndtering på den anden side kan gøres grønnere.

Figur 1. Centrale principper for grøn softwareudvikling og datahåndtering

Klimabevidst virksomhedspraksis

Tværgående principper

Sigt efter energieffektivitet

Mindre energiforbrug betyder ofte færre CO_2-udledninger.

Husk energimikset

Udfør flere processer, når strømmen er grønnere, og færre, når den ikke er det.

Mål energiforbruget

Etablér udgangspunkter og brug målinger til at afgøre effekten af tiltag.

Softwareudvikling

Bæredygtigt design

Tænk energieffektivitet ind i valg af softwarearkitektur (vælg fx mindre krævende programmeringssprog og databaser eller undgå unødvendige funktioner).

Grøn kodning

Implementér kodningsteknikker, der reducerer energiforbrug (eliminér fx død kode og brug open source-biblioteker mere ansvarligt mhp. energioptimering).

Effektiv udnyttelse af hardware

Udvikl fx algoritmer til udførelse af flere processer parallelt for at fordele arbejdsbyrden jævnt over tilgængelige ressourcer.

Datahåndtering

Minimering af datahåndtering

Undgå overførsel og behandling af unødvendige data (fx ved at bruge mindre energitunge filformater eller optimere installations- og netværkspakker).

Optimering af datahåndtering

Udskyd dataindlæsning til det er nødvendigt (fx via paginering og "lazy loading") og/eller til tidspunkter med stor tilgængelighed af grøn strøm.

Potentialer ved cloud-baserede løsninger

Undersøg mulige klimagevinster ved (delvist) at flytte arbejdsopgaver til skyen (fx ved at udnytte de store udbyderes muligheder inden for dynamisk ressourceallokering).

Kapitel 2

Klimabevidst virksomhedspraksis

En klimabevidst tilgang er afgørende for, at IT-konsulenter, softwareudviklere og andre relevante faggrupper kan arbejde målrettet med grøn softwareudvikling og datahåndtering. Ligesom i andre brancher kræver den grønne omstilling på software- og dataområdet, at erfaringer og specialviden kombineres med grønne tankegange og innovationskraft.

Der er især tre aspekter, der er vigtige at forstå, for at kunne gå til softwareudvikling og datahåndtering fra en grøn vinkel. Som beskrevet i dette kapitel er der tale om:

- grundlæggende forståelse af klimaaftrykket fra informations- og kommunikationsteknologi.
- opmærksomhed på egne og kunders fordele forbundet med grønne tiltag.
- viden om egne muligheder for at begrænse klimaaftrykket.

Viden om klimaaftrykket fra IKT

Det første skridt mod et mere klimavenligt forretningsfokus er at være bevidst om IKT-sektorens energiforbrug og udledning af drivhusgasser. Det er væsentligt at have en forståelse af, hvordan softwareudvikling og datahåndtering påvirker klimaet, hvis man ønsker at kunne ændre det.

I takt med at efterspørgslen på digitale tjenester stiger (tænk fx på den hurtige udvikling inden for generativ kunstig intelligens) vokser energiforbruget og dermed klimaaftrykket fra IKT-sektoren. Der er studier, der estimerer, at de samlede drivhusgasudledninger fra IKT kan stige fra 2-4 procent i dag til op til 14 procent af de globale udledninger i 2040.[10] Fra både et klimamæssigt og et økonomisk perspektiv er der derfor behov for handling, som kan begrænse IKT's energiforbrug via grøn softwareudvikling og datahåndtering.

Innovation på området kræver grundlæggende viden om problematikken. Det er fx nødvendigt at forstå forskellen mellem klimaaftrykket relateret til fremstilling og bortskaffelse af hardware på den ene side og klimaaftrykket fra brugsfasen af hardware på den anden side. Med hensyn til det førstnævnte kan IT-professionelle med fordel tilegne sig viden om vigtigheden af bæredygtigt design. Det indebærer fx, at computere og mobiltelefoner udvikles og produceres således, at de bruger mindre energi, i højere grad består af genbrugsmaterialer og

har et modulært design, som bl.a. gør reparationer nemmere og forlænger levetiden.

I forhold til brugsfasen af hardware er forholdet mellem udnyttelsen af computerressourcer (processor, hukommelse, harddisk, netværk osv.) og deres energiforbrug en vigtig faktor. Hvor meget energi der forbruges, afhænger bl.a. af den software og det styresystem, der anvendes, samt intensiteten af systemets aktivitet, men selv når computere og servere er inaktive, forbruger de stadig strøm.[11]

Fordele ved at gå den grønne vej – for IKT-virksomheder og deres kunder

En direkte fordel ved udviklingen af grønne tilgange til softwareudvikling og datahåndtering kan være reducerede energiomkostninger. I en tid med stigende og ustabile energipriser er det formentlig en særlig motivation for mange virksomheder. Med mere effektive kodningsprocesser kan der fx spares på energien under softwareudviklingen, mens energioptimeret software kan hjælpe kunder med at opnå den samme effekt i softwarens driftsfase.

Derudover har en CEO-undersøgelse af IBM vist, at ledere, der implementerer bæredygtigheds- og digitaliseringsinitiativer som grøn softwareudvikling, oplever en højere gennemsnitlig driftsmargin sammenlignet med deres konkurrenter, som ikke har taget disse skridt. Det skyldes bl.a., at effektivitet og innovation ofte går hånd i hånd med bæredygtige praksisser, og dermed fører til bedre økonomiske resultater.[12]

Mange virksomheder har egne klimamål, og de bliver i stigende grad også mødt af eksterne krav om at redegøre for deres miljøbelastning. Store og/eller børsnoterede virksomheder i Danmark er forpligtet til at arbejde med ESG-rapportering (Environment, Social and Governance), som indebærer, at deres påvirkning af miljø og samfund dokumenteres. Herunder hører udarbejdelsen af et klimaregnskab, der bl.a. opgør virksomhedernes CO_2-udledninger. Derudover skal bl.a. industrivirksomheder i Danmark forholde sig til obligatoriske CO_2-afgifter.[13]

Med implementering af grøn softwareudvikling og datahåndtering kan IKT-virksomheder

Træningsmodul om grøn softwareudvikling og datahåndtering

Green Software Foundation er en nonprofitorganisation, der fokuserer på bæredygtig softwareudvikling. Næsten 70 organisationer har tilsluttet sig foreningen, herunder bl.a. GitHub, Google, Intel, Microsoft og Siemens som styrende medlemmer.

På hjemmesiden tilbyder Green Software Foundation en bred vifte af online ressourcer, herunder et onlinekursus * om grøn softwareudvikling rettet mod IT-professionelle. Kurset præsenterer grundlæggende principper og begreber inden for området og uddyber, hvordan disse kan anvendes til design og udvikling af egne softwareapplikationer. Gennem de forskellige elementer kan kursister tilegne sig viden om:

- sammenhængen mellem software og drivhusgasudledninger, herunder en introduktion til forskellige typer udledninger.

- principperne bag grøn softwareudvikling og datahåndtering (fx reduktion af energibehov, effektiv udnyttelse af hardware og brug af vedvarende energi).
- metoder til beregningen af klimabelastning fra software, herunder anvendelse af den såkaldte SCI-specifikation (se tekstboksen s. 15).

* Onlinekurset kan tilgås her:
learn.greensoftware.foundation

komme tættere på at realisere både deres egne og kundernes klimamålsætninger, hvilket ikke blot formindsker deres generelle klimaaftryk, men også fremmer deres ansvarlige virksomhedsprofil. En konkret mulighed for at klæde sine egne medarbejdere på til dette arbejde er et træningsmodul udviklet af Green Software Foundation, der formidler grundlæggende viden om udvikling af grøn software (se tekstboksen).

Mange muligheder for grønne tiltag på software- og dataområdet

IT-konsulenter, softwareudviklere og andre relevante faggrupper har mange muligheder for at begrænse klimaaftrykket fra software. Rapportens følgende kapitler præsenterer både mindre komplekse løsningsforslag, såsom princippet om automatisk at slukke for hardware, der ikke bruges aktivt, og mere komplicerede tilgange, såsom automatisk distribution af datatunge processer ift. steder og tidspunkter med lavest kulstofintensitet i energisystemet.

Overgangen til en mere klimavenlig tilgang til softwareudvikling og datahåndtering kræver, at en række udfordringer overkommes. IT-ledere kan fx være overbeviste om, at udvikling af grøn software fører til større omkostninger, enten fordi softwaren bliver mere kompleks, performer dårligere, eller kræver stigende infrastrukturomkostninger. De kan også have en holdning til, at energiforbruget fra software

er ubetydeligt, at det bliver opvejet af løbende effektiviseringer af hardware, eller at software allerede er optimeret til at være så energieffektivt som muligt.[14] Hvis man både tror, det er omkostningstungt at lave forbedringer, og at forbedringen er ubetydelig, så bliver der formentlig heller ikke handlet.

For at opnå grøn softwareudvikling og datahåndtering er det desuden vigtigt for IKT-virksomheder at udfordre kunderne og deres forventninger til produkter fra et grønt perspektiv. Kunderne er ikke nødvendigvis klar over, hvor meget tilføjelsen af enkelte funktioner til en software kan koste i strøm. Som det uddybes i kapitel 3.1, bliver grøn kundevejledning et nyt kompetenceområde for udviklere og sælgere af softwareløsninger. Og med stigende opmærksomhed på bæredygtighed og klimaomkostninger hos kunderne vil det formentlig også blive et konkurrenceparameter.

Det gælder altså om at inkludere både softwareudviklere, ledere, sælgere og kunder i overvejelserne om, hvordan grøn softwareudvikling og datahåndtering kan føres ud i livet. Vidensdeling er et vigtigt skridt for at kunne skabe denne forandring, og denne rapports præsentation og drøftelse af de følgende tiltag og teknikker er et bidrag hertil.

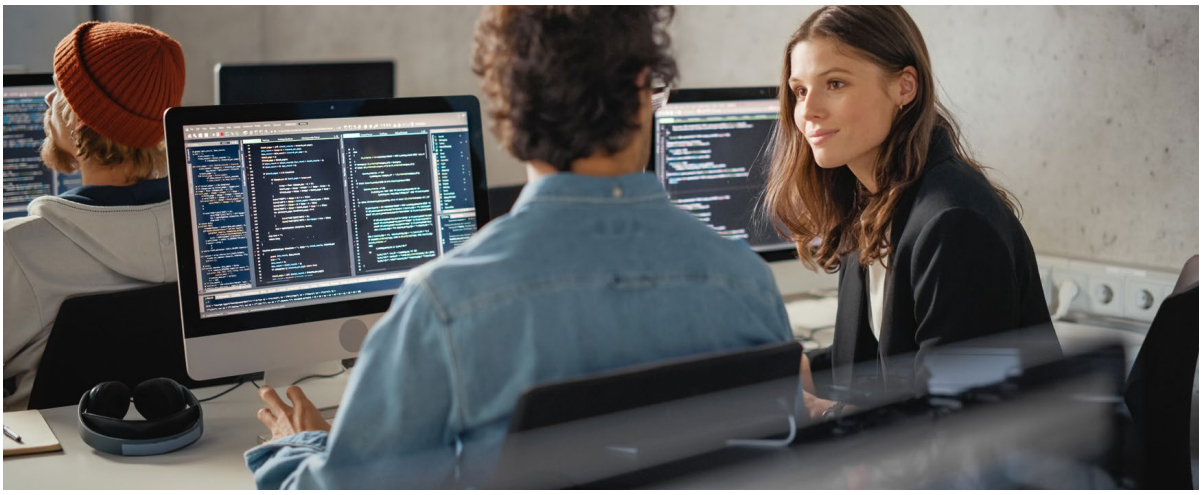
Kapitel 3

Grøn softwareudvikling

Grønne tilgange til softwareudvikling er, som nævnt, et relativt nyt område – både i forskningen og hos virksomhederne. En af de mest almindelige anbefalinger i litteraturen er at starte med at måle energiforbruget på hardware. Ved at etablere en baseline for energiforbruget kan udviklere løbende vurdere effekten af de ændringer, som foretages for at gøre software mere bæredygtigt. Når først en baseline er etableret, bliver det med andre ord lettere at undersøge og implementere tiltag, der kan reducere energiforbruget.

Der findes flere måder at måle energiforbruget af hardware med forskellig nøjagtighed. Den mest præcise metode er at anvende et fysisk måleinstrument, et multimeter, der kan integreres mellem computeren og stikkontakten og måle strømforbruget. Der kan dog være en række udfordringer forbundet med denne form for energimåling. Det er fx ikke muligt at isolere softwaren fra det system, den kører på, hvorfor det er vigtigt at minimere "baggrundsstøj" – bl.a. ved at lukke ned for andre applikationer og processer. Dertil kommer, at det kan være særligt udfordrende at opnå tidsmæssig synkronisering mellem indsamling af energidata og eksekveringen af den software, der undersøges.[15]

En enklere metode er at benytte sig af interne loggere til energimåling. Her er der tale om softwarebaserede løsninger, som henter og anvender data fra sensorer på forskellige hardwarekomponenter. Moderne CPU'er, (central processing units) gør det muligt at tilgå data om energiforbruget for CPU, RAM (intern hukommelse) og GPU (graphics processing unit), når den er en integreret del af computerens bundkort. Der er evidens for, at disse målinger korrelerer fint med fysiske målinger af en computers samlede strømforbrug.[16] Tekstboksen på næste side indeholder flere oplysninger om konkrete redskaber til softwarebaseret måling,



hvis nøjagtighed dog også afhænger af, at man formår at minimere støj fra øvrige applikationer og processer i systemet.

Tid kan også anvendes som en enklere og mere grovkornet proxy for energiforbrug. Ved at måle den tid det tager for en applikation eller algoritme at køre, får man en indikation af energiforbruget, forudsat at det øvrige aktivitetsniveau på hardwaren holdes konstant. Fordi energiforbruget er højere, når CPU'en og andre ressourcer er aktive, gælder det som udgangspunkt, at jo længere tid en opgave tager at udføre, desto mere energi vil den typisk

Redskaber til måling af energiforbrug

Hvis det er upraktisk eller irrelevant at foretage fysiske målinger af energiforbruget ved stikkontakten, kan der i stedet bruges softwarebaserede løsninger (på engelsk kaldet for energy profilers). Luís Cruz, fra det Tekniske Universitet Delft, har kortlagt fordele og ulemper ved at anvende disse interne energiloggere.* Han har udviklet en guide til, hvordan man kan måle energiforbrug fra software vha. eksperimenter med fokus på energieffektivitet. Guiden beskriver, hvordan man opsætter energimålinger med minimal bias, og hvordan man bedst analyserer og drager konklusioner fra sine energimålinger.**

Et eksempel på en intern logger er Intel PCM, der består af en API (snitflade til kommunikation mellem forskellige systemer) og flere redskaber til monitorering af energiforbruget af Intels processorer. Intel PCM kan avendes på en bred vifte af styresystemer, fx til at fremkalde data i realtid om, hvor effektivt processoren eksekverer instruktioner, og hvor meget data der sendes frem og tilbage mellem computerens hukommelse.***

På Linux, der bruges som fundament for mange serversystemer, kan man bruge kommandoprompten *perf* for at tilgå et detaljeret analyseredskab til monitorering af systemperformance, herunder brug af CPU og hukommelse samt aktiviteter ifm. interne og eksterne dataoverførsler.**** Endnu et eksempel er HWiNFO (for Windows og DOS), der giver adgang til dybdegående informationer om forskellige systemkomponenter og efter egne udsagn bl.a. bruges af NASA til monitorering af deres computersystemer.*****

* Cruz (2021). Tools to Measure Software Energy Consumption from your Computer. tinyurl.com/3z4d9vp4
** Cruz (2021). Green Software Engineering Done Right: a Scientific Guide to Set Up Energy Efficiency Experiments. tinyurl.com/yhx52s8r
*** Intel (u.d.). Intel®Performance Counter Monitor – A Better Way to Measure CPU Utilization. tinyurl.com/nhe2p35n
**** Kuldeepkumawat (2024). What is the Linux perf Command? tinyurl.com/3sxhneku
***** HWiNFO (u.d.). Professional System Information and Diagnostics. tinyurl.com/yddtr3tj

forbruge. Ved denne metode skal der dog også tages højde for bias – dvs. skævvridning af måleresultater som følge af fx andre kørende applikationer og baggrundsprocesser.

De forskellige målemetoder deler dog en blind vinkel. Det er nemlig ikke muligt at foretage nøjagtige målinger af energiforbruget relateret til netværkstrafik, hvilket bl.a. skyldes kompleksiteten i netværksinfrastrukturen og de mange veje, data kan tage igennem den, når der fx er tale om webbaserede systemer. Denne problemstilling danner bl.a. baggrund for principper og teknikker ifm. grønne cloudløsninger i afsnit 4.2.

Helt overordnet er det vigtigt at måle energiforbruget i alle faser af softwareudviklingen: i de indledende testfaser, under udviklingen af softwaren samt i de efterfølgende evaluerings- og optimeringsfaser. Målinger er nødvendige for at identificere, hvor forbedringer kan være mest effektive, og for at sikre, at de implementerede optimeringer opnår de ønskede resultater. Beslutninger om, hvilke komponenter i softwaren der helst skal optimeres, bør træffes på baggrund af: 1) hvor der bruges mest energi (ift. de mest ressourcekrævende komponenter), og 2) hvilke komponenter, der køres ofte og/eller over lange perioder.[17]

I de følgende afsnit præsenteres mere konkrete anbefalinger til, hvilke tiltag der kan tages for at skabe mere energieffektiv software.

3.1 Bæredygtigt design

I den første fase af softwareudviklingen skal der bl.a. træffes beslutninger om, hvilke funktioner softwaren skal have, hvordan den skal designes, og hvilke(n) database(r), der skal benyttes. Beslutninger om softwarearkitektur kan have stor indflydelse på softwarens energieffektivitet og kan være vanskelig at ændre senere i processen. Undersøgelser tyder på, at nogle programmeringssprog kan føre til et højere energiforbrug end andre (dette uddybes i afsnit 3.2), samt at brugen af mere ressourcekrævende databaser, der understøtter avancerede forespørgsler, med fordel kan erstattes med enklere og mere energieffektive alternativer.[18] For softwareudviklere, der vil udvikle og sælge mere bæredygtige softwareprodukter, er det derfor essentielt at være opmærksom på betydningen af indledende valg i designfasen for softwarens fremtidige energieffektivitet.

Undgå unødvendige funktioner og rådgiv kunder om bæredygtighed

IT-professionelle, der udvikler og markedsfører software, skal navigere mellem kundernes ønsker og faktiske behov. Med bevidstheden om behovet for mindre klimabelastende software tilføjes der et yderligt lag til denne udfordring. Ud over faktorer som ydeevne, sikkerhed og pris skal softwareudviklere også drøfte med deres kunder, hvad ønskede funktioner og egenskaber betyder for klimaprofilen af en given softwareløsning. Det behøver dog ikke nødvendigvis at være en øvelse i at spare funktioner væk, men det kan være en oplagt mulighed for IT-professionelle til at demonstrere deres ekspertise inden for grøn softwareudvikling. Det er her, de kan foreslå at gennemføre et projekt efter principper om lav kulstofintensitet (se tekstboksen næste side) og komme med bud på, hvordan kundens målsætninger kan opnås i samspil med konkrete grønne tiltag.

En del af dette arbejde indebærer, at softwareudviklere gør deres kunder bevidste om konsekvenserne af mere traditionelle fremgangsmåder. Det kan være, at energiomkostningerne, ved ikke at optimere datahåndtering, hurtigt kan komme til at overstige prisen for at implementere netop den del i udviklingsfasen – fx grundet stigende udgifter for strøm og CO_2-udledninger eller et behov for investeringer i ny hardware som følge af den mindre effektive datahåndtering. I denne sammenhæng kan det

også være passende at udfordre sine kunder til at nøje overveje, hvilke funktioner der er nødvendige, og hvilke der kan undværes.

Sagt med andre ord handler det om at rådgive kunderne i en mere bæredygtig retning, samt at fremhæve energieffektivitet som et attraktivt valg der bl.a. kan hjælpe med at reducere kundernes og slutbrugernes klimaaftryk.

Brug SCI-redskabet til at skabe systematik

Den føromtalte nonprofitorganisation, Green Software Foundation, har udviklet et redskab ved navn Software Carbon Intensity Specification (forkortet SCI-specifikation), der kan bruges til at arbejde systematisk med reduktionen af klimaaftrykket fra software. Redskabet definerer en systematisk og transparent metode til at beregne CO_2-emissioner for et softwaresystem. Formålet er bl.a. at understøtte softwareudviklere i at træffe mere bæredygtige beslutninger, fx når det kommer til valg af softwarearkitektur.

I april 2024 blev SCI-specifikationen anerkendt som en international standard af ISO (International Organization for Standardization), der beskriver redskabet som en "pålidelig, fair og sammenlignelig protokol til måling og reduktion af klimaaftrykket fra software".[19] SCI-specifikationen er alsidig og kan bruges ifm. forskellige former for softwareapplikationer og miljøer (fra private computere og datacentre til offentligt tilgængelige cloudløsninger).[20] Derfor kan IT-professionelle med fordel anvende redskabet til deres strategiske arbejde med grøn softwareudvikling, herunder også dokumentationen af deres egen indsats på området (se tekstboksen for uddybende informationer).

ISO-standard til beregning af klimaaftrykket fra software

Redskabet Software Carbon Intensity Specification (SCI-specifikation) kan bruges til at beregne kulstofintensiteten af forskellige typer software på en systematisk måde. Resultatet af beregningen er en specifik SCI-karakter. Her er lave tal bedre end høje, men man kan ikke opnå et 0.

Man finder frem til SCI-karakteren ved at beregne softwarens samlede CO_2-emission over en periode pr. enhed af "arbejde" eller "funktionalitet". Afhængig af softwaretypen kan der være tale om emissioner pr. yderligere bruger, pr. træningscyklus inden for maskinlæring eller pr. API-kald (anmodninger fra en softwareapplikation til en anden). Princippet er enkelt: jo lavere SCI-karakter, jo mindre CO_2-udledning pr. arbejde. En software vurderes således efter dens energieffektivitet og ikke dens samlede CO_2-udledning. Derfor er det muligt at sammenligne bl.a. store distribuerede og cloudbaserede løsninger med små monolitiske softwaresystemer.

Et vigtigt princip ved SCI-specifikationen er, at man kun kan forbedre softwarens SCI-karakter ved at reducere de faktiske CO_2-udledninger – fx ved at optimere softwaren til at kræve mindre computerkraft eller i højere grad satse på vedvarende energi. Det er ikke muligt at forbedre karakteren vha. andre foranstaltninger ifm. klimakompensationer, såsom at plante træer eller købe CO_2-kvoter.

Læs mere om SCI-specifikationen her: tinyurl.com/2wwbrc54

Prioritér tiltag med størst mulig effekt

At gøre kode grønnere og mere klimavenlig kan være en tidskrævende proces. Det er derfor vigtigt at overveje, hvor man bedst muligt kan bruge egne kræfter. Her er bl.a. produktets levetid og anvendelse en afgørende faktor. Hvis et program fx kun anvendes på 50-100 enheder i cirka 10 minutter om måneden, og desuden forventes at blive udfaset inden for 6 måneder, vil fordelene ved at optimere softwaren sandsynligvis være minimale. Derimod vil en lille forbedring i energiforbruget af en særlig udbredt applikation, som anvendes mere regelmæssigt af mange og over et længere tidsrum, føre til betydelige klimamæssige fordele.

Derudover er en god rettesnor ift. samtlige teknikker, der præsenteres i denne rapport, at man prioriterer de tiltag med det største potentiale for at opnå energibesparelser. Det kan fx være, at det giver mest mening at fokusere alle sine kræfter på optimering af en hyppig anvendt database, mens der i andre tilfælde med fordel kan foretages små justeringer i kildekoden for at udnytte hardware mere effektivt.

Tænk bæredygtighed ind i valget af arkitekturen

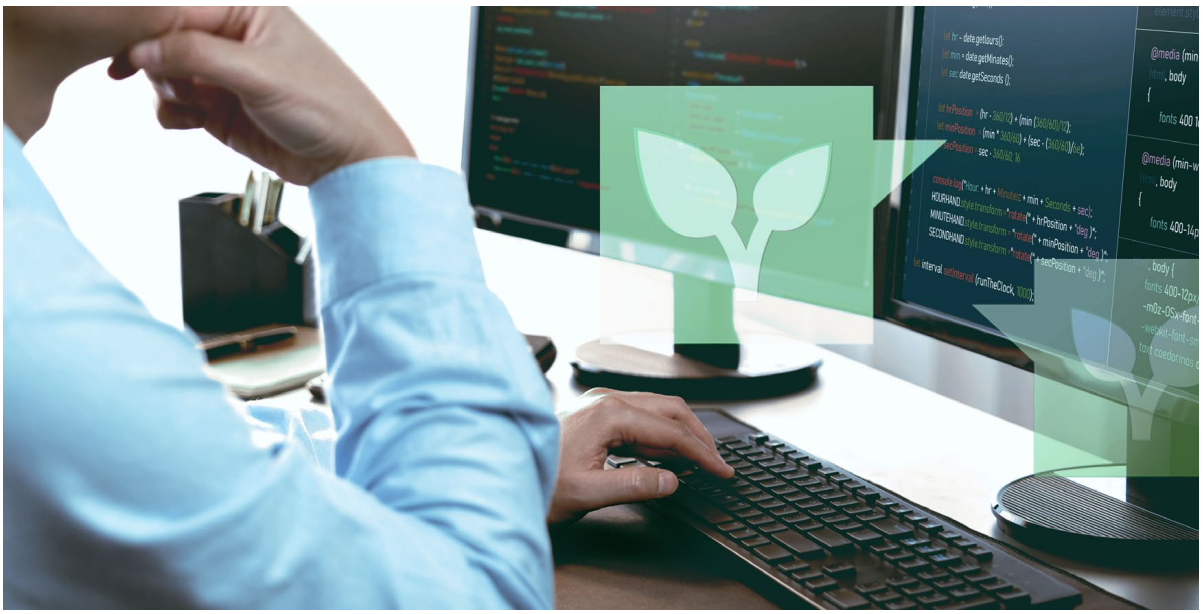
Valget af arkitektur kan have stor betydning for klimaprofilen af ens endelige produkt. Der findes forskellige måder at strukturere softwareapplikationer på, og blandt den seneste tids mest fremtrædende arkitekturmønstre er en modulær tilgang, også kaldet microservices.

Opdelingen af softwarens arkitektur vha. microservices er ensbetydende med brugen og kombinationen af en række mindre, selvstændige tjenester, som hver især er ansvarlige for bestemte funktioner og kommunikerer med hinanden via snitflader i form af API'er. Et konkret eksempel er e-handelsplatforme, der består af og forener en bred vifte af microservices med hver deres funktionalitet, herunder fx produktkatalog, brugergodkendelse, ordregennemførelse, betalingsservice og brugeranmeldelser. Det står i kontrast til en monolitisk struktur, hvor en softwareapplikation er bygget op som en samlet og meget tættere koblet enhed.

Der findes mange arkitekturmønstre med hver deres fordele og ulemper, og valget af det ene frem for det andet vil afhænge af rammerne for udviklingen af softwaren og kravene til slutproduktet. Sammenlignet med microservices er en monolitisk arkitektur fx nemmere at udvikle og distribuere, fordi der netop ikke skal tages højde for kommunikation og gensidig afhængighed mellem forskellige underordnede applikationer. Omvendt kan det være svært at tilpasse eller udskifte enkelte funktioner i den monolitiske arkitektur, fordi den ikke er opbygget i moduler, og fordi ændringer i et område kan have en uønsket effekt i andre.[21]

På samme måde kan der være fordele og ulemper set fra et grønt perspektiv. En klar fordel ved microservices er muligheden for automatisk op- og nedskalering af individuelle ressourcer efter behov. Den modulære opbygning tillader også, at man fx vælger det bedst egnede programmeringssprog eller den mest effektive database for hver microservice, hvilket samlet set kan føre til en mere energioptimeret softwareløsning. Samtidig kan den ofte cloudbaserede og distribuerede microservices-arkitektur medføre øget netværkstrafik (grundet hyppig kommunikation mellem modulerne) og et større behov for lagring og håndtering af data, bl.a. fordi uafhængigheden af de enkelte microservices ofte sikres ved at duplikere data for hver service.[22]

For at opsummere vil det altid afhænge af den enkelte brugscase, hvilke arkitekturmønstre softwareudviklere vil kunne vælge imellem. For at fremme grøn softwareudvikling er det imidlertid vigtigt at overveje og reflektere over de forskellige arkitekturers styrker og svagheder mhp. energieffektivitet.



Vælg programmeringssprog med omhu

Valg af programmeringssprog kan for mange softwareudviklere være et spørgsmål om vane, men meget tyder på, at alene valget af sprog kan gøre koden betydeligt mere energieffektiv. Litteraturen indikerer bl.a., at de statiske sprog, der anvender såkaldte compilere, er mest energieffektive. Her er der tale om softwarebaserede redskaber, der forenklet sagt forstår, kvalitetssikrer og oversætter mere abstrakt kildekode direkte til binær maskinkode, der kan udføres af computeren. Denne proces finder sted inden eksekvering af kildekoden. Det er modsat dynamiske sprog, hvor abstrakt og højniveau-kode fortolkes under selve udførelsen af koden.[23]

Et konkret forsøg har set nærmere på energiforbruget for 27 forskellige programmeringssprog vha. tre benchmarktests. Forsøget viste, at forskellen i energiforbruget mellem det mindst effektive sprog (det dynamiske sprog Python) og det mest effektive sprog (compilersproget C) lå på gennemsnitlig 4.000 procent ved gennemførelsen af de samme beregninger.[24]

Det er vigtigt at påpege, at de belyste programmeringssprog typisk anvendes på forskellige områder og til forskellige formål. Med andre ord kan de ikke frit udskiftes eller erstatte hinanden. Afhængig af brugscase kan der være gode grunde til at vælge et sprog frem for et andet. Men litteraturen viser også, at dynamiske sprog anvendes i server eller "backend"-applikationer, hvor dette hverken er nødvendigt eller energieffektivt (især når der er tale om hyppigt anvendte moduler).[25] Her kan der hentes klimagevinster ved at vælge programmeringssprog med omhu. Endnu en mulighed er at undersøge muligheden for at bruge et foretrukket sprog med en anden type compiler, hvilket kan bidrage til øget energieffektivitet. Anbefalingen herfra lyder, at man tester de forskellige muligheder, da det bl.a. kan afhænge af den anvendte hardware, hvilken compiler der er bedst egnet.

Grundlæggende gælder, at det er vigtigt at overveje relationen mellem valget af programmeringssprog og softwarens (fremtidige) energiprofil tidligt i udviklingsfasen. Når et softwareprojekt har opnået en vis størrelse, kan det være forbundet med større omkostninger at flytte store mængder kode over til et andet sprog til fordel for øget bæredygtighed.

3.2 Grøn kodning

Grøn kodning handler om at indtænke klimamæssig bæredygtighed i generering og behandling af kode. Målet er at reducere energiforbruget fra software, bl.a. ved at integrere mere energieffektive kodningsteknikker og algoritmer til optimal udnyttelse af den tilgængelige hardware.

Minimer code bloat

Al software har en "vægt", som kan variere afhængigt af, hvordan koden er skrevet. "Tung" kode er overkompliceret og ineffektivt i sine funktioner, hvorimod "let" kode på den mest direkte og enkle vis løser sine opgaver. Inden for softwareudvikling anvendes det illustrative begreb code bloat til at beskrive situationer, hvor kilde- eller maskinkode bliver unødvendig lang og/eller langsom at eksekvere, hvilket typisk lægger et unødigt pres på de hardware- og softwarekomponenter, som driver den pågældende applikation.

Løsningen til at undgå denne form for ressourcespild er mere eller mindre simpel – om end

Eliminering af død kode

Eliminering af "død kode" (ubrugt kode) er en central tilgang inden for grøn kodning. Afhængig af programmeringssprog (se afsnit 3.1) er der forskellige muligheder for at optimere koden.

I **compiler-sprog** er det relativt let at identificere død kode. Her oversætter et softwarebaseret redskab – en såkaldt compiler – højtniveau-kode til binær maskinkode, som en computer direkte kan forstå og eksekvere. Forklaret lidt forenklet, læser og kvalitetssikrer compileren kildekoden inden dens eksekvering og er derfor i stand til at identificere funktioner, klasser og variabler i koden, der aldrig bliver kaldt.

I **dynamiske sprog**, som Python eller JavaScript, bliver koden oversat under eksekvering. Det betyder, at nødvendigheden af specifikke linjer kode først bliver be- eller afkræftet under programkørslen. Derfor kan man i dynamiske

sprog ikke lige så nemt og automatisk identificere, hvad der er død kode.

Der findes dog ikke desto mindre strategier og redskaber til at identificere og fjerne død kode ifm. anvendelsen af dynamiske sprog – med hver deres fordele og ulemper. Til JavaScript findes der fx en række konkrete metoder, såsom tree-shaking eller det nyudviklede Lacuna-redskab. Begge værktøjer analyserer kildekode, mens den eksekveres, hvorefter de identificerer afhængigheder mellem linjer af kode og dernæst fjerner isolerede kodestumper (dvs. dem der aldrig bliver kaldt).*

* Malavolta et al (2023). JavaScript Dead Code Identification, Elimination, and Empirical Assessment, tinyurl.com/2p95svn8

den kan tage tid. Det kræver nemlig, at udviklere effektiviserer deres kode ved at gennemgå og optimere den.[26] Tekstboksen på forrige side uddyber mulige tilgange hertil.

Brug open source-biblioteker ansvarligt

Det er en udbredt praksis at anvende open source-biblioteker, når en ny softwareapplikation skal udvikles.[27] Her drejer det sig kort fortalt om samlinger af allerede skrevet kode til fri afbenyttelse af andre brugere, der kan bygge egne applikationer oven på dem.

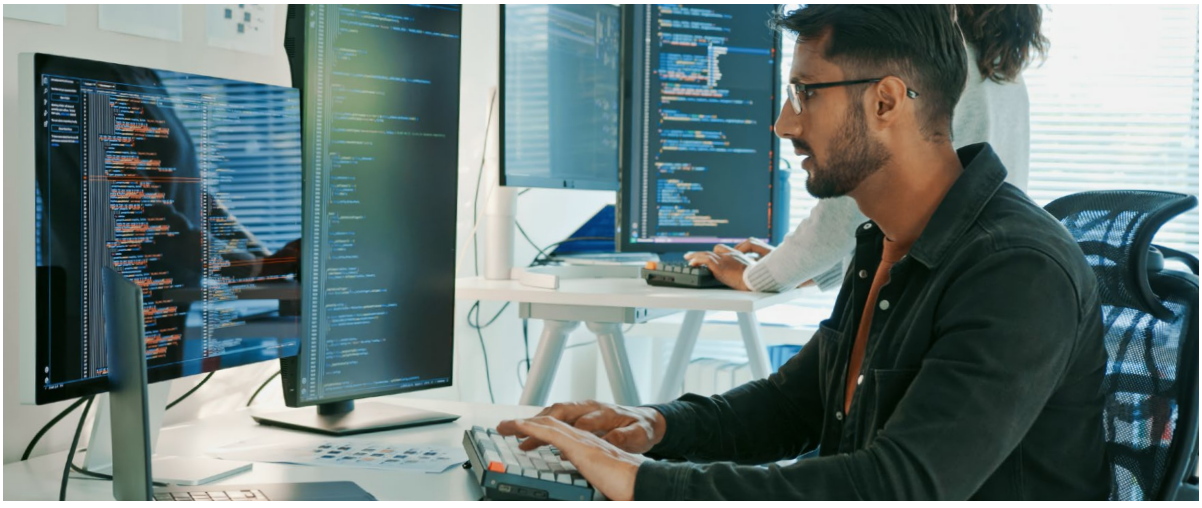
Open source-biblioteker kan medføre et energimæssigt ressourcespild, fordi de typisk indeholder en betydelig mængde kode for en bred vifte af funktioner. Det gør dem på den ene side til en slags multiværktøj, der kan anvendes i sammenhæng med mange forskellige applikationer. På den anden side kan brugen af open source-biblioteker medvirke til code bloat (se forrige underafsnit). Dette sker fx, når en udvikler ukritisk inkluderer et helt bibliotek i en applikation, selvom kun en lille del af dets funktionalitet er nødvendigt og faktisk bliver brugt. Den overflødige kode kan medføre ekstra energiforbrug og derfor unødvendig CO_2-udledning.[28]

Ud over at begrænse sig til relevante dele af open source-biblioteker er det også værd at overveje alternativer til de biblioteker, man er vant til at bruge. Ofte vil flere biblioteker tilbyde den samme funktionalitet, men være mere eller mindre energikrævende. Udfordringen er selvfølgelig, at man ikke på forhånd kan være sikker, hvilket bibliotek der er mest energieffektivt i et givent brugstilfælde. Her kan sammenligninger vha. målinger af energiforbruget anvendes for at teste sig frem til den bedste løsning.

Sammenlign forskellige versioner af kodebasen

Udover at energimåle på specifik kode kan en fremgangsmåde også være at sammenligne energiforbruget mellem forskellige versioner af en kodebase. Med begrebet beskrives den samling af kildekode, der omfatter alle filer og instruktioner, som er nødvendige for at kompilere og køre et givent program.

Softwareudviklere kan starte med at etablere en baseline, dvs. måle og identificere energiforbruget for den nuværende version af kodebasen, som grundlag for fremtidig sammenligning. Ved at teste energiforbruget af forskellige versioner af kodebasen, kan udviklere spore sig ind på, hvilke konkrete ændringer i et givent softwareprogram der har medført en stigning eller et fald i energiforbruget, og i hvilket omfang.[29]



Sørg for optimal udnyttelse af anvendt hardware

Det er fysiske computere og hardware, der skal udføre de instruktioner, der er defineret i udviklet kode. Denne proces er energikrævende, og valget af hardware, samt den måde koden eksekveres på, kan være af stor betydning for omfanget af energiforbruget.

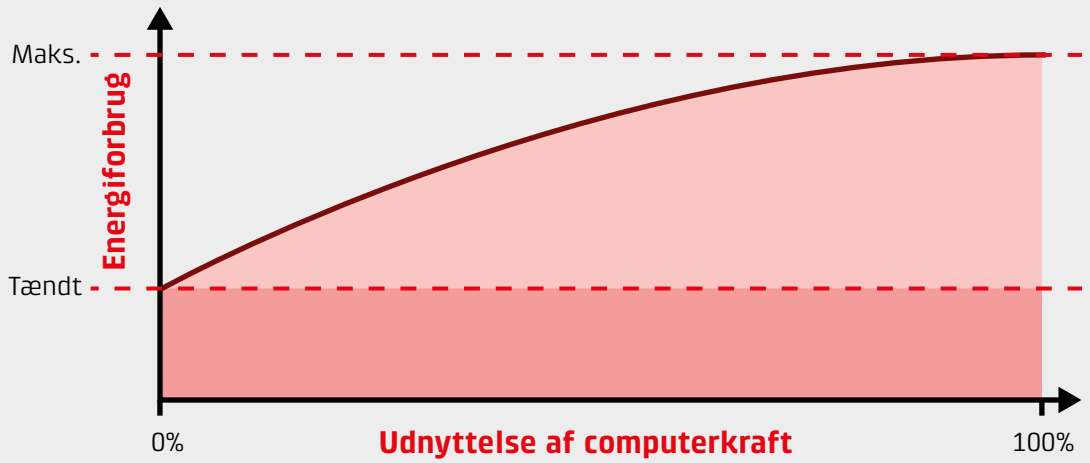
Et vigtigt princip i denne sammenhæng handler om energiproportionalitet. Princippet siger, at højere udnyttelsesgrader af computerressourcer fører til et lavere energiforbrug pr. andel anvendt computerkraft. Princippet er illustreret i figur 2 og går ud på, at en computer forbruger en vis mængde strøm, når den er inaktiv, mens computerens energiforbrug ikke stiger proportionelt med dens udnyttelsesgrad.

Derfor er det typisk en god idé at udnytte computerens regnekraft så vidt som muligt. Det kan fx være betydeligt mere energieffektivt at sætte computeren til at udføre flere opgaver samtidigt frem for ad flere omgange, eller at have én computer der udnyttes i høj grad, frem for to der udnyttes i mindre grad.

En mulig tilgang er multi-threading, hvor programmer gøres hurtigere ved at lade dem udføre flere opgaver parallelt. Metoden er særlig nyttig i programmer, der skal håndtere mange brugere samtidig, såsom web-servere. En anden metode er multi-processing. Her udnyttes den moderne arkitektur af processorer med flere kerner, som på samme tid kan løse forskellige opgaver. Mens multi-threading fungerer inden for en enkel proces, hvor flere sekvenser af instruktioner deles om de samme ressourcer, involverer multi-processing flere separate processer i forskellige dele af CPU'en.[30]

Endnu en mulighed er at udnytte processoren på dedikerede grafikkort (GPU). GPU'er kan behandle mange datatråde samtidigt, hvilket gør dem effektive til opgaver, der kan opdeles i mange små dele. At designe software, der skal trække på GPU'ers større regnekraft, er dog

Figur 2. Forhold mellem energiforbrug og udnyttelse af computerkraft



Kilde: Teknologisk Institut baseret på Suárez et al. (2021). Green Coding. tinyurl.com/ydt9dwzv

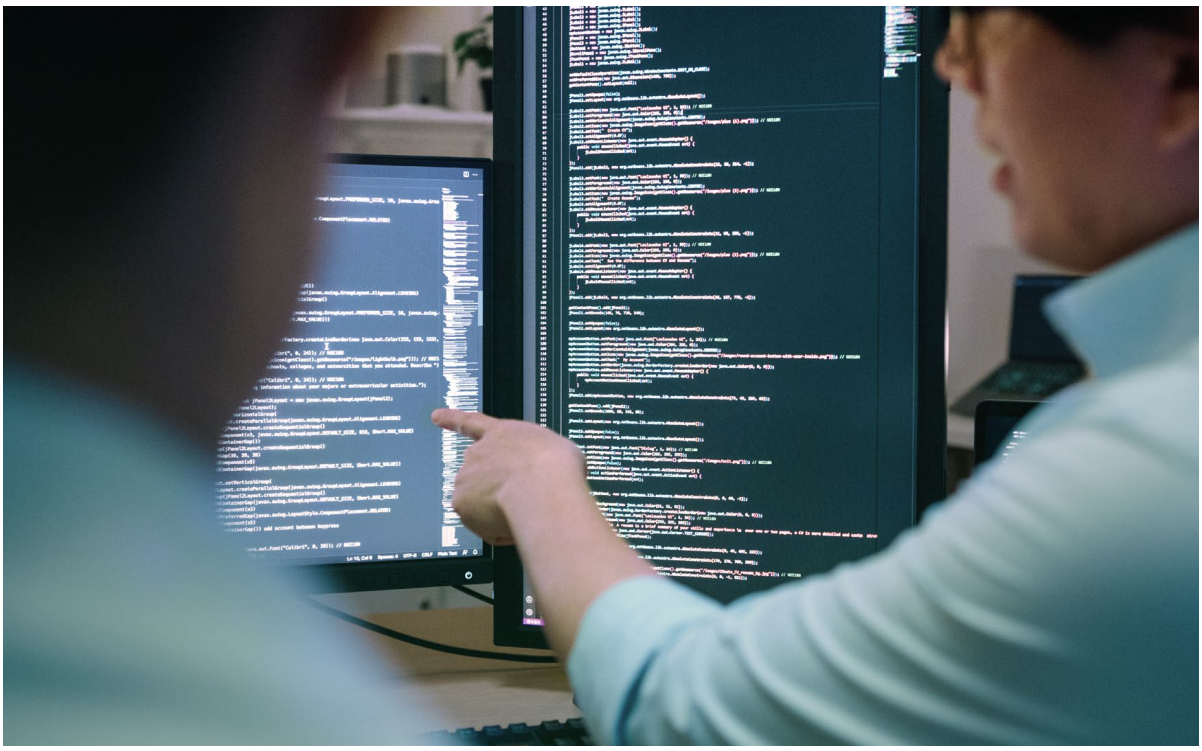
ikke automatisk et grønnere valg. For det første er disse enheder særligt energikrævende, og for det andet kan krav om deres brug stille større krav til den hardware, som slutbrugeren skal have til rådighed (og måske anskaffer sig som erstatning for eksisterende hardware til at kunne afvikle en given applikation). Fra et grønt perspektiv skal GPU'en altså bruges med omhu og udelukkende der, hvor opgaverne kræver dens styrke.[31]

Det er også vigtigt at fremhæve, at en maksimal belastning af hardware ikke er særlig energieffektiv. Overbelastning af processorer går ud over deres responstid og ydeevne, hvilket i sidste ende betyder større strømforbrug. For at kunne implementere krævende kode uden at overbelaste hardware (eller skabe behov for yderligere eller mere kraftfuld hardware), kan softwareudviklere bl.a. indskrive "pauser" i deres kode, der får processoren til at gå i en kortvarig dvaletilstand. Det forlænger eksekveringstiden, men reducerer samtidig processorens arbejdsbyrde og strømforbrug.[32] Her vil det formentlig være muligt at finde et "sweet spot" mellem udnyttelsesgraden af CPU'en og det samlede strømforbrug af tilgængelig hardware vha. målinger.

Overvej at genberegne

En anden relevant overvejelse at gøre sig under kodningsprocessen er, hvornår det er mest energieffektivt at udføre beregninger igen, og hvornår det er mest optimalt at indlæse data, der allerede er blevet beregnet.

Det er fx ikke altid mere energieffektivt at hente gemte værdier fra computerens harddisk, frem for at lave beregningen igen, da processoren kan gennemføre simple beregninger hurtigt og med mindre energiforbrug. Potentielle energibesparelser er dog bl.a. afhængige af, hvorvidt der gennemføres andre aktiviteter på systemet samtidig, og hvordan data er spredt på tværs af dets hukommelse.[33] Derfor findes der ikke en entydig regel om, hvad der er bedst. Ikke desto mindre er det en god idé at være opmærksom på, at simple genberegninger kan være mindre energikrævende på tidspunkter, hvor systemet ikke er særlig belastet.



> **Ligesom i andre brancher kræver den grønne omstilling på software- og dataområdet, at erfaringer og specialviden kombineres med grønne tankegange og innovationskraft.**

Danske IKT-virksomheder kan drage nytte af det store innovationspotentiale inden for grøn softwareudvikling og datahåndtering og aktivt bidrage til at reducere klimaaftrykket fra informations- og kommunikationsteknologi.

Kapitel 4

Grøn datahåndtering

Datahåndtering er lig med energiforbrug – både i lokale miljøer og netværksmiljøer. Når en computer behandler data lokalt, trækker den på hardwarekomponenter, såsom computerens processor, grafikkort, systemhukommelse og lagringsenhed. Jo mere intensiv datahåndteringen er (fx indlæsning af videomateriale i høj billedkvalitet eller beregninger i store regneark), jo mere strøm skal komponenterne bruge.

Ud over det lokale energiforbrug ved datahåndtering skal der også tages højde for forbruget, der sker i forbindelse med netværkstrafik. De fleste IT-systemer er i dag webbaserede og sender løbende data frem og tilbage mellem lokale enheder og eksterne servere. Denne proces involverer netværksudstyr som routere og switches, der bruger strøm for at sikre, at dataflytningen sker effektivt. Dertil kommer, at data sendes til og fra eksterne servere, der også bruger strøm. Her er der tale om specialiserede faciliteter med sammenkoblede computere og servere til behandling, lagring og distribuering af store mængder data. Datacentre er typisk store energiforbrugere, da deres systemer holdes konstant kørende og skal forsynes med tilstrækkelig køling for at undgå overophedning.[34]

Det konkrete energiforbrug fra netværkstrafik er svært at måle, bl.a. fordi data rejser gennem forskellige enheder og systemer, der befinder sig på forskellige lokationer. Som en proxy er man i længere tid gået ud fra, at der er et proportionelt forhold mellem den datavolumen, der sendes via et netværk, og netværkets energiforbrug – selvom det reelle energiforbrug er mere kompliceret at opgøre, da det også afhænger af konfigurationen, kapaciteten og effektiviteten af relevant netværksudstyret.[35]

Et af de områder, hvor energiforbruget af datahåndtering bliver mere håndgribeligt for slutbrugere, er cloudløsninger. Her er der tale om et netværk af eksterne servere, der tilgås via internettet, og som lagrer, administrerer og behandler data i stedet for at bruge lokale servere eller personlige computere. De store cloududbydere, såsom Amazon Web Services, Google Cloud, IBM Cloud og Microsoft Cloud kaldes også for hyperscalers, da deres løsninger bygger på enorme datacentre med dynamisk styring af ressourcer alt efter behov.

Kunder af cloududbydere kan betale for relevante tjenester på forskellige måder. Blandt de populære modeller er betaling for den regnekraft og lagerplads, som kunder faktisk bruger (pay-per-use). Her får kunder en regning fra deres cloududbyder baseret på ressourceforbruget.[36] Fordi kunder formentlig er interesse-

ret i at holde deres omkostninger nede, er der derfor et direkte incitament til at minimere, hvor meget deres software trækker på cloud-ressourcer.

Mange af de teknikker og principper, der blev gennemgået under grøn softwareudvikling, kan hjælpe i denne sammenhæng, idet energioptimeret software netop er designet til at begrænse unødvendig brug af computerkraft. I de følgende afsnit belyser vi flere måder, hvorpå IT-professionelle kan fremme grøn datahåndtering til fordel for et mere effektivt ressourceforbrug.

4.1 Energieffektiv datahåndtering

Hver proces inden for datahåndtering er forbundet med energiforbrug. Derfor giver det mening at stramme op på, hvor meget data der bliver overført og behandlet. Fra et grønt perspektiv er det derudover vigtigt, at den datahåndtering, der bliver foretaget, så vidt muligt er energioptimeret. Som det uddybes i det følgende, kan det indebære, at den i høj grad er drevet af vedvarende energikilder eller finder sted i relativ nærhed til, hvor data skal bruges (fx ifm. cloudløsninger).

Optimer installations- og netværkspakker

En mulig tilgang er at reducere størrelsen på installationspakker, der indeholder de nødvendige filer og biblioteker, som en modtagende enhed kræver for at kunne køre en given applikation. Ved at undgå at disse pakker indeholder kode, der er unødvendig for applikationens funktionalitet (i tråd med anbefalingen om at undgå code bloat), minimeres dataforbruget.

Effekten af små justeringer kan være forholdsvis stor, fx fordi webapplikationer i programmeringssproget JavaScript henter installationspakker pr. besøg på en relevant hjemmeside og ikke kun en gang. I forlængelse heraf, kan man også arbejde med at minimere størrelsen af netværkspakker, der groft sagt udgør opdelte segmenter af data, der sendes over internettet og andre computernetværk. Det kan bl.a. opnås ved at skære ned på overflødige informationer indeholdt i pakkerne. Afhængig af brugscasen og krav til funktionalitet, kan det fx være en mulighed at reducere informationer inkluderet i metadata.[37]

Brug mindre energitunge filformater

En anden mulighed er i højere grad at benytte sig af bæredygtige filformater, der indeholder mindre eller mere kompakt kode. Filformaterne JSON og XML bruges typisk til at overføre data fra en web-server til en hjemmeside. Lidt forenklet organiserer de begge komplekse datasæt på en måde, der gør dem forståelige og læsbare for forskellige programmeringssprog og grænseflader til at kommunikere mellem flere programmer (dvs. API'er). Der er markante forskelle mht. de to filformaters funktionalitet. Til simple dataoverførsler kan det fx være en god idé at anvende det mere kompakte JSON frem for det mere komplekst strukturerede XML-format, da JSON-filer typisk er mindre.[38]

Et andet eksempel er valget mellem forskellige billedformater, hvilket er særligt vigtigt inden for webdesign, hvor billeder spiller en central rolle. Til simple illustrationer kan det lette SVG-format være et bedre valg end det tungere PNG-format, der bl.a. kan håndtere gennemsigtige og halvtransparente baggrunde.[39] I forlængelse heraf er det også muligt at tilvælge filer med lidt lavere kvalitet til fordel for energibesparelser. Her er et eksempel brug af MP3-formatet til lydfiler, mens der mht. billeder og videoer kan vælges lavere opløsning for at reducere filstørrelser.

Overvej komprimering

Der kan selvfølgelig også arbejdes bevidst med at komprimere data for at reducere de datamængder, der skal lagres eller sendes over netværket. I forbindelse med sidstnævnte er det dog vigtigt at være opmærksom på, at det

kræver både energi at komprimere data på afsendersiden og at 'pakke ud' på modtagersiden. Som ved de fleste andre teknikker præsenteret i denne rapport gælder det om at prøve sig frem. Det kan være en start at måle på energiforbruget ifm. komprimering og udpakning af relevant data med henblik på at kunne sammenholde effekten med opnåede besparelser mht. netværkstrafik.

Hent data efter behov

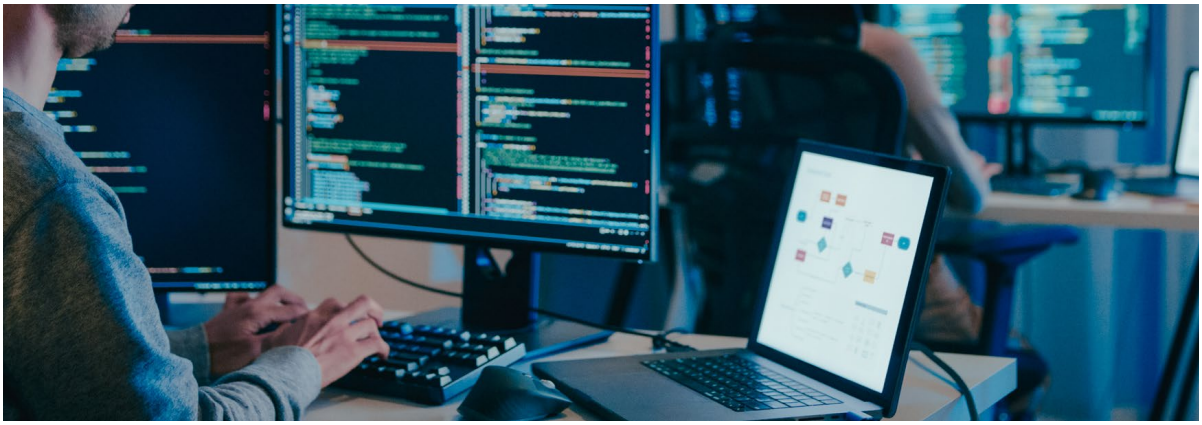
Softwareudviklere kan med fordel udskyde tidspunktet for indlæsning af data, indtil det faktisk er nødvendigt – dvs. skal tilgås af brugeren. Dette princip kaldes også for lazy loading.

En teknik er paginering hvor datasæt bliver opdelt og hentet i mindre og mere overskuelige dele frem for at blive indlæst i deres helhed.[40] Resultater af en Google-søgning bliver fx opdelt og vist ved hjælp af paginering. I stedet for at hente 10.000 elementer, kan der med paginering fx hentes 100 elementer ad gangen – mens der kun indlæses flere, når brugeren aktivt kræver det (hvilket på Google svarer til at klikke sig videre til næste resultatside).

En anden tilgang er det såkaldte proxy-designmønster, hvor et mindre objekt overføres som erstatning og pladsholder for et bagvedliggende og tungere hovedobjekt. Proxy-objektet indeholder alle de oplysninger, der er behov for i forhold til at hente hovedobjektet, men vil udskyde hentningen, indtil brugeren har behov for det. Det kan eksempelvis bruges ved opbygning af en større hjemmeside, der indeholder flere billeder og videoer. I stedet for at indlæse samtlige grafiske elementer på én gang, kan der længere nede på hjemmesiden – i den del, som brugere kan scrolle sig ned til – være proxy-objekter med metadata om de billeder og videoer, der kan hentes. Hvis brugerne ikke når længere ned på hjemmesiden, bliver hovedobjekterne aldrig indlæst.[41]

Planlæg dig til grønt energiforbrug

Når der skal udføres komplekse beregninger eller håndteres større mængder data, er det fra et grønt perspektiv ikke lige meget, hvornår det bliver gjort. For at gøre disse processer mere bæredygtige kan IT-professionelle bl.a. vælge at udføre dem på tidspunkter med lav kulstofintensitet. Det er primært tilfældet, når energimikset er forholdsvis grønt, og der i høj grad kan trækkes på vedvarende energi frem for kulstofudledende kraftværker. Er der tale om distribueret datahåndtering, dvs. på tværs af forskellige geografiske lokationer og datacentre, kan det også give mening at gennemføre relevant datahåndtering på det sted, hvor strømmen er grønnest. Ifm. disse tilgange er der også økonomiske gevinster at hente, fordi strømmen typisk er billigere, når kulstofintensiteten er forholdsvis lav.



Det kan selvfølgelig være en udfordring at planlægge tunge beregninger og processer mhp. at ramme tidspunkter eller steder med et særligt grønt energimiks, fx fordi det kan være svært at tilgå eller holde styr på data om energiforbrug og relateret klimaaftryk på tværs af datacentre og lokationer. Der eksisterer dog allerede i dag software, der kan hjælpe i denne proces. Her er der bl.a. tale om API'er til hentning af realtidsdata om internationale el-systemer og deres kulstofintensitet samt redskaber til orkestrering, der kan opdele og distribuere arbejdsopgaver på baggrund af disse data.

Disse løsninger kan med andre ord sørge for, at energikrævende processer udføres på det sted og tidspunkt, hvor strømforbruget er mest bæredygtigt.[42] Tekstboksen introducerer en række eksempler.

Det er klart, at en del processer og opdateringer – og især dem af kritisk natur – ikke kan vente på den bedst mulige tilgængelighed af vedvarende energi. Der kan være behov for regelmæssige eller cykliske opdateringer, som kun i mindre grad kan udskydes. Men det kan gøre en forskel at fokusere på kulstofbevidst

Redskaber til kulstofbevidst planlægning

Der findes forskellige API'er, der giver adgang til historiske data og realtidsdata om elnettet i forskellige lande og regioner. Disse løsninger kan brugere anvende til automatisk at hente informationer om bl.a., hvor strømmen i en given region kommer fra, og hvor meget CO_2 der bliver udledt ifm. dens produktion. Blandt de mest udbredte eksempler er den offentligt tilgængelige API fra nonprofitorganisationen WattTime* og den kommercielle API fra danske Electricity Maps,** der bl.a. benyttes af Google Cloud til at beregne klimaaftrykket fra deres cloudbrugere.

Kører man mikroservices, kan data fra de omtalte API'er hentes og integreres med redskaber til orkestrering af opgaver og processer. Specifikke eksempler er Kubernetes og Nomad.*** Begge er open-source værktøjer, der kan konfigureres til automatisk at planlægge og distribuere arbejdsopgaver i forhold til kulstofintensiteten i relevante el-systemer. Et automatisk setup til orkestrering af datahåndtering på tværs af tid og rum vil typisk give mening for brugere med adgang til flere datacentre – fx ifm. cloudløsninger.

*WattTime Data API, se mere her: tinyurl.com/4drww3zw
**Electricity Maps API, se mere her: tinyurl.com/4pu7xt2y
*** Flere informationer om kulstofbevidst planlægning vha. disse orkestreringsværktøjer kan findes her: tinyurl.com/3cyfs7xu & tinyurl.com/ed86h6nf

planlægning af ikke-kritiske opgaver – og det gælder også for brugere, der ikke har mulighed for at sprede "de tunge processer" over flere ressourcer eller lokationer.

Brug cachelagring for at spare på ressourcer

For at spare på ressourcer som CPU, hukommelse og netværksbåndbredde kan datahåndteringen gøres mere bæredygtig ved hjælp af cachelagring. Her gemmes kopier af hyppigt anvendte data i en cache, dvs. et midlertidigt lagringssystem, der giver hurtig adgang til disse data og forbedrer systemets eller applikationers ydeevne.[43]

Webbrowsere anvender typisk cachelagring, der gemmer indhold af besøgte hjemmesider på en lokal harddisk i et afgrænset tidsrum. Når brugeren besøger hjemmesiden for første gang, hentes indholdet fra webserveren til lagring i en lokal mappe. Ved genbesøg af siden indlæses gemt indhold fra den lokale cache i stedet for at blive hentet fra serveren igen.

Der findes mange forskellige former for caching. Fx rummer de fleste processorer (dvs. CPU'er) flere højhastigheds-caches, der sidder mellem processoren og computerens hukommelse (RAM). CPU-caching kan give processoren hurtigere adgang til instruktioner fra et givent program end fra computerens hovedhukommelse.

En anden form for caching sker i forbindelse med Content Delivery eller Content Distribution Network (CDN). Et CDN er et netværk af servere spredt ud over forskellige geografiske lokationer, herunder en central server med originalt indhold og flere edgeservere, som distribuerer kopier af dette indhold. Formålet er at levere hurtigere indhold til brugerne, uanset hvor de befinder sig i verden, ved at levere det fra den nærmeste CDN-server.[44]

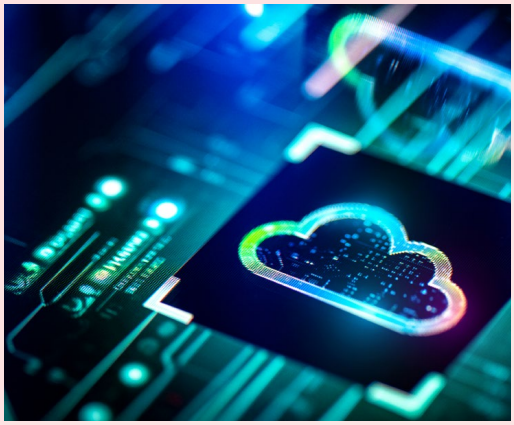
4.2 Bæredygtige cloudløsninger

Der er en lang række grunde til, at cloud computing (dvs. anvendelsen af cloudløsninger) er blevet udbredt i det digitale samfund. Cloud computing giver bl.a. virksomheder mulighed for nemt at skalere deres IT-ressourcer op eller ned efter behov. Ved brugen af cloudløsninger kan virksomheder omkostningseffektivt få adgang til teknisk avanceret hardware og infrastruktur, da de ofte kun betaler for de computerressourcer, de faktisk ender med at bruge.

Fra et klimaperspektiv byder udnyttelsen af cloudløsninger også på forskellige fordele, bl.a.

Energieffektivitet af cloudløsninger versus lokale datacentre

Til at beskrive hvor energieffektivt et datacenter er, anvendes måleenheden PUE. Det står for Power Usage Effectiveness og beregnes ved at dividere den samlede mængde strøm, der forbruges i et datacenter (inklusive energiforbrug til køling og belysning), med den mængde energi, der direkte bruges til at køre IT-udstyret i datacentret. Et perfekt PUE-resultat er 1,0, hvilket betyder, at al strøm går direkte til driften af hardwaren.



Ifølge en global survey af datacentre blandt industrivirksomheder ligger det gennemsnitlige PUE på 1,58 i industriens lokale ("on-premise") datacentre, mens især større datacentre er mere energieffektive og når ned til en PUE på 1,44.

Sammenlignet hermed opnår de største datacentre, de såkaldte hyperscalers, udbudt af Amazon Web Services, Microsoft Cloud og Google Cloud, PUE-værdier helt ned til 1,15 og lavere, takket være bl.a. avancerede design- og kølingsløsninger.

Gillis (2022). Definition: power usage effectiveness (PUE). tinyurl.com/bddapdbx
Almekinders (2024). AWS achieves near-perfect PUE score at European site. tinyurl.com/y44whb3w

fordi centraliserede datacentre kan drives mere effektivt end mange mindre, lokale serverrum. I denne sammenhæng handler de følgende principper om, hvordan bæredygtig brug af cloudløsninger kan bidrage til at reducere IKT's klimaaftryk.

Overvej flytning af arbejdsopgaver til skyen

Lokale datacentre er typisk mindre energieffektive end systemerne til datalagring og -håndtering hos store cloududbydere (se tekstboksen). Det skyldes bl.a., at disse udbydere opnår en højere udnyttelsesgrad af deres hardware og investerer massivt i grønne tiltag, herunder brugen af vedvarende energi og anvendelse af overskudsvarme fra kølingsprocesser i lokale fjernvarmesystemer.[45]

En anden forklaring på, at lokale datacentre ofte er mindre effektive, er, at der typisk sker en overestimering af serverkapaciteten. De ansvarlige udviklere kan fx forvente et for højt antal brugere generelt eller antage, at brugertallet vil stige i en længere periode, efter datacentret er taget i brug. Forkerte antagelser om udnyttelsen kan være ensbetydende med et overforbrug af computerressourcer. Netop her har offentlige cloudløsninger, som brugere kan købe sig adgang til, stort potentiale for at spare energi. De beror nemlig bl.a. på brugen af en stor pulje af forskellige datacentre og automatisk tildeling af ressourcer, hvilket muliggør en mere effektiv hardwareudnyttelse gennem hurtig op- og nedskalering efter behov.[46]

De store cloududbydere, som Google Cloud, Microsoft Azure og Amazon Web Services (AWS), tilbyder såkaldte carbon trackers, som hjælper brugerne med at måle energirelaterede CO_2-udledninger fra deres faktiske forbrug af cloudtjenester. Disse værktøjer samler og afrapporterer bl.a. data om brugen af specifikke datacentre på baggrund af lokale emissionsfaktorer.[47] Dermed kan virksomheder, der er kunder hos de omtalte cloududbydere, nøje holde øje med deres klimaaftryk fra brugen af cloudløsninger. Det skal dog fremhæves, at nøjagtigheden af kulstofmålerne endnu ikke er blevet afprøvet i uafhængige studier.

Det er klart, at anvendelsen af cloudløsninger som erstatning for lokale datacentre ikke altid vil være en mulighed fx pga. datasikkerhed, krav til hastighed af datahåndtering (latency) eller andre hensyn. Fra et grønt perspektiv kan det give god mening at flytte (dele af) sine arbejdsopgaver til skyen. Det er dog vigtigt, at de data og processer, som ønskes håndteret i skyen, optimeres til netop det.[48] Desuden skal der altid holdes øje med, om brugen af cloudløsninger fører til et større træk på ressourcer ifm. netværksinfrastruktur sammenlignet med mere lokale serverløsninger. Med andre ord skal det overvejes, hvornår stordriftsfordele ved brugen af offentlige cloudsystemer opvejes af betydeligt mere datatrafik over længere afstande.

Undersøg mulige fordele ved brug af edge computing

I de tilfælde, hvor det ikke er muligt at flytte al datahåndtering til skyen, kan det give mening at overveje anvendelsen af edge computing. I forbindelse med industrielle IoT-løsninger kan det fx være nødvendigt, at datalagring og beregninger delvis foregår tæt på de enheder og sensorer, der genererer og har brug for relevante data (fx pga. krav til hurtig datahåndtering og kommunikation mellem forbundne enheder). Ved edge computing foregår den lokale behandling af data typisk på mindre servere med begrænset regnekapacitet, som er forbundet med mere kraftfulde softwareapplikationer i en cloudløsning. Med andre ord udføres lagring og behandling af data delvis ved grænsen eller kanten af skyen (derfor "edge"), mens tungere dataprocesser håndteres i cloudsystemet.[49]

Ved at håndtere data delvist lokalt kan edge computing bl.a. bidrage til at reducere den datavolumen, der sendes til centraliserede datacentre. Fx er der mulighed for at filtrere og aggregere data lokalt, før det sendes til videre behandling i skyen. På den måde kan der spares på de energiomkostninger, som er forbundet med dataoverførsler over længere afstand.[50] (Her går de tekniske formål med edge computing – fx mindre netværksforsinkelser og hurtigere beregninger – hånd i hånd med konceptet grøn datahåndtering).

Større energieffektivitet kan også opnås ved lokalt at anvende specialiseret hardware, der er optimeret til at udføre specifikke arbejdsopgaver i edgemiljøet – hvilket hænger sammen med det grundlæggende princip om at udnytte tændte og aktive computerressourcer så meget som muligt. Løsninger til edge computing kan med fordel konfigureres til en dynamisk distribution af arbejdsopgaver og -belastning på tværs af netværket. Det kan bl.a. gøres med afsæt i tilgængelighed af ledige ressourcer og grønne energikilder[51] (se også kulstofbevidst planlægning i afsnit 4.1).

Selvom edge computing byder på mange (grønne) fordele, er det dog ikke en teknologi, der er fri for udfordringer med hensyn til bæredygtighed. Implementeringen af edgesystemer kræver hardware, der er forbundet med et klimaaftryk fra bl.a. produktion, strømforbrug under drift og eventuel bortskaffelse. Derudover viste en virksomhedssurvey blandt 307 brugere, at kapaciteten af edge computing-ressourcer sjældent udnyttes effektivt.[52] Det skal derfor undersøges, om de potentielle klimabesparelser fra anvendelsen af edge computing

kan opveje klimaaftrykket fra implementering og drift.

Vælg cloududbydere efter klimaprofil

Hvis datahåndtering skal foregå ved brug af cloudløsninger, kan brugere med fordel undersøge, hvilke udbydere der tilbyder de grønneste ydelser. Her kan man fx undersøge, hvorvidt udbydernes datacentre bruger vedvarende energikilder, og hvor effektivt deres el- og vandforbrug er ifm. køling, som er nødvendig for at undgå overophedning af hardware.

De største cloududbydere (de føromtalte hyperscalers) har sat sig ambitiøse klimamål, herunder opnåelsen af "net-zero" emissioner, og investerer derfor løbende i grønne tiltag. Men der kan være store forskelle udbyderne imellem, ift. hvor godt deres klimamål er afspejlet i deres praksis – dvs. måden hvorpå deres datacentre drives. Det kan bl.a. hænge sammen med tekniske udfordringer og den måde, de samarbejder med deres underleverandør på.[53]

Dertil kommer, at cloududbyderne ikke nødvendigvis er transparente, når det gælder afrapporteringen af deres klimaaftryk, og hvordan de helt konkret tilsigter at reducere det. Flere af de store udbydere har fx tidligere afrapporteret misvisende tal for klimaaftrykket fra deres interne datacentre.[54] Derfor er det relevant at vælge sin cloudløsning ved at undersøge og sammenligne tilgængelige data om klimabelastningen af den specifikke ydelse, der ønskes købt, på tværs af udbydere.

Kapitel 5

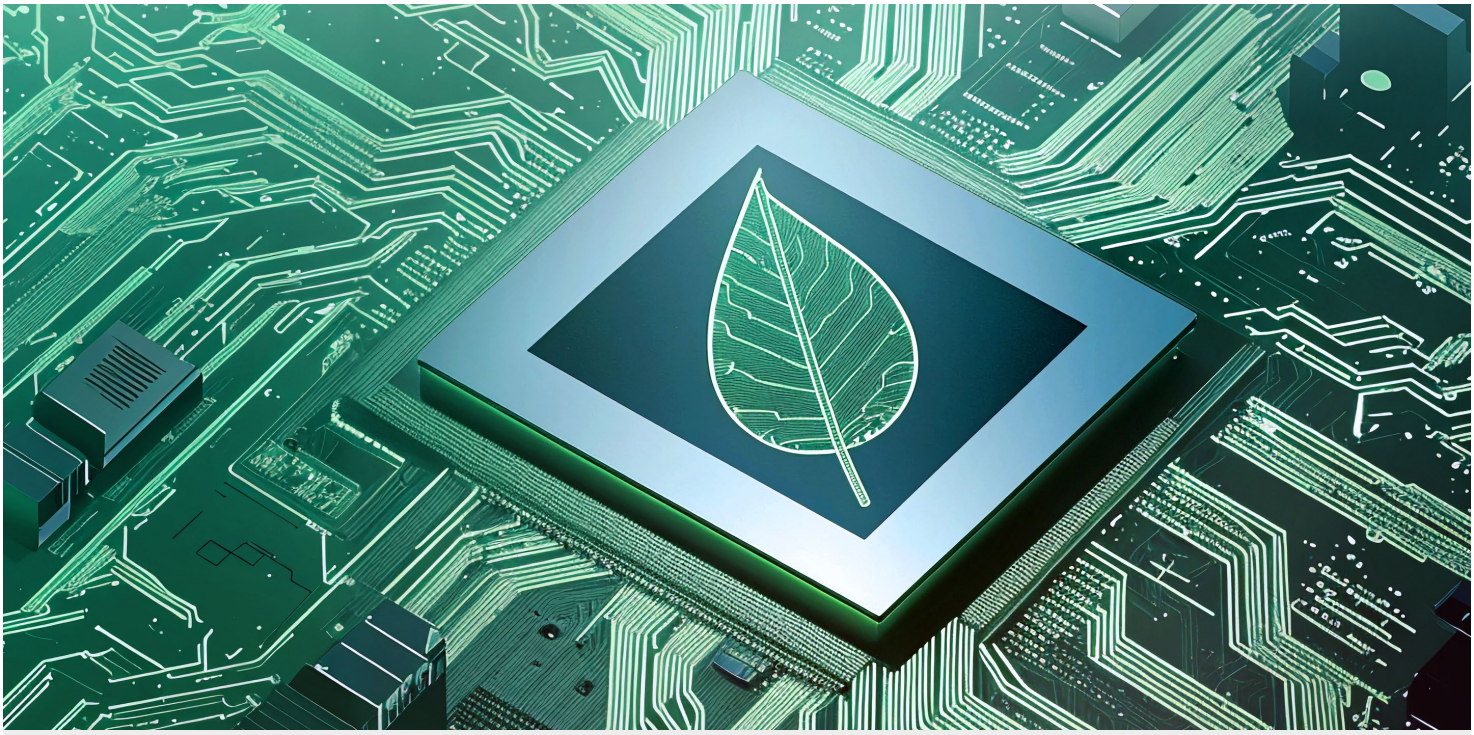
Afsluttende bemærkninger og anbefalinger

Det har været formålet med de foregående kapitler at give læseren inspiration til, hvordan der konkret kan arbejdes med grøn softwareudvikling og datahåndtering. Der er, som præsenteret, mange muligheder for grønne tiltag på software- og dataområdet, og det bliver kun vigtigere at beskæftige sig med dem fremadrettet. Blandt de væsentligste årsager hertil er en fortsat stærk digitalisering af samfundet på baggrund af bl.a. klimaudfordringer og et overforbrug af jordens ressourcer (herunder sjældne jordartsmetaller som fx anvendes i produktion af computere og mobiltelefoner) samt stigende krav til virksomheder om at adressere disse problemstillinger.

Der er flere energitunge områder inden for informations- og kommunikationsteknologi, som der ikke har været plads til at behandle særskilt i denne rapport. Det blev fx nævnt i indledningen, at antallet af IoT-enheder på verdensplan er vokset med næsten 190 procent i løbet af de sidste fem år (fra ca. 10 milliarder i 2019 til 18,8 milliarder i 2024).[55] Dermed vil det for fremadrettede initiativer bl.a. give mening at se nærmere på, hvordan man kan anvende principperne introduceret i denne rapport til at fremme mere energieffektive IoT-løsninger.[56]

Et andet voksende område, der allerede bliver diskuteret i forbindelse med energiforbrug, er kunstig intelligens (AI) – herunder især generativ AI i form af bl.a. chatbots som ChatGPT og Microsoft Copilot. Det estimeres, at et enkelt prompt i ChatGPT gennemsnitligt fører til samme forbrug som 40 mobilopladninger. Samtidig skaber komplekse sprogmodeller og de enorme mængder data, de trænes på, et støt stigende behov for datacentre, hvorfor AI-servere forventes at bruge lige så meget strøm som Sverige i 2027.[57] For slutbrugere er det derfor en god idé at begrænse forbruget af generativ AI vha. sund fornuft. Hvis det ikke er nødvendigt at bruge ChatGPT, fordi en almindelig Google-søgning er tilstrækkelig til at komme videre, er der her mulighed for at begrænse energiforbruget. En større forskel kan dog gøres af de udviklere, der arbejder med sprogmodeller og træning af sprogmodeller. Forskning har vist, at der kan opnås massive energibesparelser ved at indbygge bæredygtigt design og energieffektivitet i udviklingsfasen af sprogmodeller.[58]

Danske IKT-virksomheder kan drage nytte af det store innovationspotentiale inden for grøn softwareudvikling og datahåndtering og aktivt bidrage til at reducere klimaaftrykket fra informations- og kommunikationsteknologi. Infografikken i figur 3 indeholder en køreplan for, hvordan virksomheder, der udviklere, sælger og rådgiver om softwareløsninger, kan arbejde målrettet med netop det.



Figur 3. Vejen mod grøn softwareudvikling og datahåndtering

Klimabevidsthed

Indledningsvis er der behov for at øge klimabevidstheden i virksomheden. Både softwareudviklere og sælgere, der har direkte kundekontakt, bør tilegne sig viden om klimaaftrykket fra IKT, herunder hvorfor der er behov for at reducere det, og hvilke muligheder der er for at bidrage hertil – både i virksomheden og hos slutbrugerne. Et sted at starte, er at lade sine medarbejdere gennemføre online kurser inden for grøn IT og kodning (se fx tekstboksen s. 12).

Målsætning

Ud over at sprede viden i virksomheden, er der behov for at sætte klare målsætninger for arbejdet med grøn softwareudvikling og datahåndtering. En oplagt mulighed er at udvælge et konkret softwareprojekt og arbejde hen imod certificeringen af den givne software som energieffektiv. Her kan der fx arbejdes inden for rammerne af ISO-standarden til måling af kulstofintensiteten af software (se tekstboksen s. 17).

Implementering

Klædt på med baggrundsviden og klare mål for øje, er det næste skridt implementeringen. Her er der tale om at anvende grønne kodeteknikker og sikre bæredygtig datahåndtering. De mange principper og teknikker i denne rapport er et godt udgangspunkt, og der findes flere løsningsforslag (fx i rapportens mange kilder – se noterne). Vigtigt er at prioritere sin indsats der, hvor den har størst muligt effekt.

Opfølgning

Der mangler viden om effekten af grønne tiltag på området. Derfor er målinger af strømforbruget før og efter vigtige – for at blive klogere, tilpasse løbende og sætte nye mål. Opfølgning handler også om at vedligeholde og energioptimere eksisterende applikationer og løsninger til datahåndtering, bl.a. for at undgå, at de med tiden bliver mere energikrævende (fx pga. code bloat eller ineffektiv ressourceudnyttelse).

Anvisningerne i figur 3 er selvfølgelig lettere at beskrive end at udføre. Som tidligere fremhævet er der mange faktorer ud over bæredygtighed, som IT-professionelle må forholde sig til ifm. udvikling og levering af softwareløsninger. Der kan fx nævnes pris, funktionalitet, ydeevne, brugeroplevelse og sidst men ikke mindst sikkerhed (ifm. beskyttelse af data og infrastruktur). Dertil kommer, at arbejdet med grøn softwareudvikling og datahåndtering kan være tidskrævende, fordi der bl.a. er behov for at sætte sig ind i specifikke udfordringer og løsningsmuligheder. Mere overordnet er IKT-virksomheder nødt til at ændre deres praksis og fx lave om på den måde, programmeringen normalt foregår hos dem.

På trods af udfordringerne er det vigtigt for danske IKT-virksomheder at give sig i kast med udvikling og implementering af mere energieffektive løsninger inden for software og datahåndtering. Klimaaftrykket fra IKT forventes at blive større i de kommende år, mens klimamål og miljøreguleringer i stigende grad gør det nødvendigt for IKT-virksomheder og deres slutbrugere at arbejde målrettet med reduceringen af den negative klimapåvirkning.

Denne rapport har vist, at der er gode innovationsmuligheder inden for grøn softwareudvikling og datahåndtering. Danske IKT-virksomheder kan fremme deres egen og slutbrugernes miljøprofil ved at promovere og implementere grønne løsninger i samspil mellem udviklere, sælgere og kunder. Alt tyder på, at bæredygtighed også på software- og dataområdet bliver en konkurrenceparameter, hvilket understreger behovet for en aktiv indsats.

Noter

[1] Belkhir & Elmeligi (2018). Assessing ICT global emissions footprint: Trends to 2040 & recommendations. tinyurl.com/yjs6avyz

[2] World Economic Forum (2023). Charted: There are more mobile phones than people in the world. tinyurl.com/m3uyt3wp

[3] IoT Analytics (2024). State of IoT: Number of connected IoT devices growing to 18.8 billion globally. tinyurl.com/37eupzdc

[4] European Commission (2023). ICT Environmental Impact. tinyurl.com/5f88nnjw

[5] Det Internationale Energiagentur estimerer, at flytrafikken stod for 2 procent af de globale energi-relaterede CO_2-emissioner i 2022. IEA (2023). Tracking Aviation. tinyurl.com/4ke4dzsz

[6] Antal relevante firmaer med branchebetegnelse, seneste tal fra 2022: 13.430 It-konsulenter; 1.274 Informationstjenester; 928 Udgivelse af computerspil og anden software; 334 Telekommunikation. Kilde: Danmarks Statistik (DB07 10- 19- og 127-grp.)

[7] BJSS (2024). The BJSS Guide to Green Software Development. tinyurl.com/yeeucnan

[8] IBM (u.d.) What is software development? tinyurl.com/mryhx356

[9] IT Leksikon (2024). Vigtigheden af DBMS i den moderne teknologiske verden. tinyurl.com/crv8spj4 & IBM (2024). What is data management? tinyurl.com/ms8pbu4p

[10] Belkhir & Elmeligi (2018). Assessing ICT global emissions footprint: Trends to 2040 & recommendations. tinyurl.com/yjs6avyz

[11] Bærbak (2023). The Green Architecture Framework. tinyurl.com/bp7x9drr

[12] IBM (2022). Own your impact: Practical pathways to transformational sustainability. tinyurl.com/vcnbns4c

[13] Se fx Skatteministeriet (2024). En ny CO_2-afgift i industrien er en realitet. tinyurl.com/3b8wdntb

[14] McKinsey (2022). Making software and data architectures more sustainable. tinyurl.com/25e55eu7

[15] Cruz (2021). Tools to Measure Software Energy Consumption from your Computer. tinyurl.com/3z4d9vp4

[16] Bærbak (2023). The Green Architecture Framework. tinyurl.com/bp7x9drr

[17] Radarsma (2022): Green Coding: Reduce Your Carbon Footprint in Special theme: Ethical Software Engineering and Ethically Aligned Design. tinyurl.com/ybrpav5c

[18] Bærbak (2023). The Green Architecture Framework. tinyurl.com/bp7x9drr

[19] Green Software Foundation (2024). Software Carbon Intensity (SCI) Specification Achieves ISO Standard Status, Advancing Green Software Development. tinyurl.com/2pn2hwff

[20] Green Software Foundation (u.d.). Software Carbon Intensity (SCI) Specification. tinyurl.com/2wwbrc54

[21] Kushtagi, (2024). System Design Secrets: Monolith vs. Microservices Explained. tinyurl.com/52p4sc7u

[22] Hermann (2022). Green software engineering – Back to the roots! www.capgemini.com/dk-en/insights/expert-perspectives/green-software-engineering-back-to-the-roots

[23] Jain (2024). C++ vs Python (Statically Typed language vs Dynamic Language. tinyurl.com/bdf7sree

[24] Pereira et al. (2017). Energy efficiency across programming languages: how do energy, time, and memory relate? tinyurl.com/pkxdus2y

[25] Weber & Rauch (2023). Sustainable Software Design: Background and Best Practices. tinyurl.com/4j86xejv

[26] GFT (2021). Green Coding. tinyurl.com/yryk7mbw

[27] Goslin (2020). 96% of Organizations Use Open Source Libraries but Less Than 50% Manage Their Library Security Flaws. tinyurl.com/vjrpxx5k

[28] IBM (2023). Why green coding is a powerful catalyst for sustainability initiatives? tinyurl.com/bdcsmnct

[29] Meinhardt (2023). Software Energy-Efficiency: Code Optimization Tactics. tinyurl.com/2xdzvzrh

[30] Bærbak (2023). The Green Architecture Framework. tinyurl.com/bp7x9drr & Betsun (2024). The whats and whys of green software development tinyurl.com/2tj6762n

[31] Schoonhoven et al. (2022). Going green: optimizing GPUs for energy efficiency through model-steered auto-tuning. tinyurl.com/bdcuj79u

[32] SUSO (2023). Green Coding: The 5 most important basics for sustainable software development with code examples. tinyurl.com/5n7cunvw

[33] Radersma (2022). Green Coding: Reduce Your Carbon Footprint. tinyurl.com/2ptn62km

[34] Fogarty & Flucker (2023). Data Centre Essentials. tinyurl.com/3y83t2v4

[35] Mytton, Lundén & Malmodin (2024). Network energy use not directly proportional to data volume: The power model approach for more reliable network energy consumption calculations. tinyurl.com/4wbfvkz3

[36] Selvom denne regning kan være svær at gennemskue – se fx: Kuperman (2023). 5 Common Cloud Bill Issues & How To Deal With Them. tinyurl.com/bdf4yjte

[37] Bærbak (2023). The Green Architecture Framework. tinyurl.com/bp7x9drr

[38] Berga & Figueiredo (2023). JSON vs XML: which one is faster and more efficient? tinyurl.com/2cyeme7w

[39] Adobe uddyber forskellene mellem PNG & SVG her: tinyurl.com/3c8c3uxf

[40] Diallo (2024). Green API Design 4/5: Leveraging Pagination for Green API/Software. tinyurl.com/3z9emnub

[41] Bærbak (2023). The Green Architecture Framework. tinyurl.com/bp7x9drr

[42] IBM (2022). IT sustainability beyond the data center. tinyurl.com/55ervydr & BJSS (2024). The BJSS Guide to Green Software Development. tinyurl.com/yeeucnan

[43] SUSO (2023). Green Coding: The 5 most important basics for sustainable software development with code examples. tinyurl.com/5n7cunvw

[44] Sheldon (2023). What is caching? tinyurl.com/bd275upd

[45] GFT (2021): Green Coding. tinyurl.com/yryk7mbw

[46] Suárez et al. (2021). Green Coding. tinyurl.com/ydt9dwzv

[47] Se fx AWS (u.d.). Customer Carbon Footprint Tool. tinyurl.com/2thvsxz5; Google (u.d.). Carbon Footprint. tinyurl.com/4zrpaxz4 & Microsoft (u.d.). Emissions Impact Dashboard. tinyurl.com/zbmza4ps

[48] IBM (2022). IT sustainability beyond the data center. tinyurl.com/55ervydr

[49] Kjeldgaard (2021). Edge computing for begyndere. tinyurl.com/24mt6av9

[50] Efendi (2024). Edge Data Aggregation: Processing Data at the Edge. tinyurl.com/3nx5p9k9

[51] Gamage & Perera (2024). Optimizing Energy Efficient Cloud Architectures for Edge Computing. tinyurl.com/44ppnyux

[52] Nolle (2024). How edge can help improve sustainability and how it can't. tinyurl.com/mrvha3hm

[53] Law (2024). Sustainability is Central to Hyperscalers' Strategies. tinyurl.com/mps2whx3

[54] CUDO Ventures (2022). Greenwashing the cloud: hyperscale providers and the climate crisis. tinyurl.com/5e5kjcdw & O'Brien (2024). Data center emissions probably 662% higher than big tech claims. Can it keep up the ruse? tinyurl.com/34ax778u

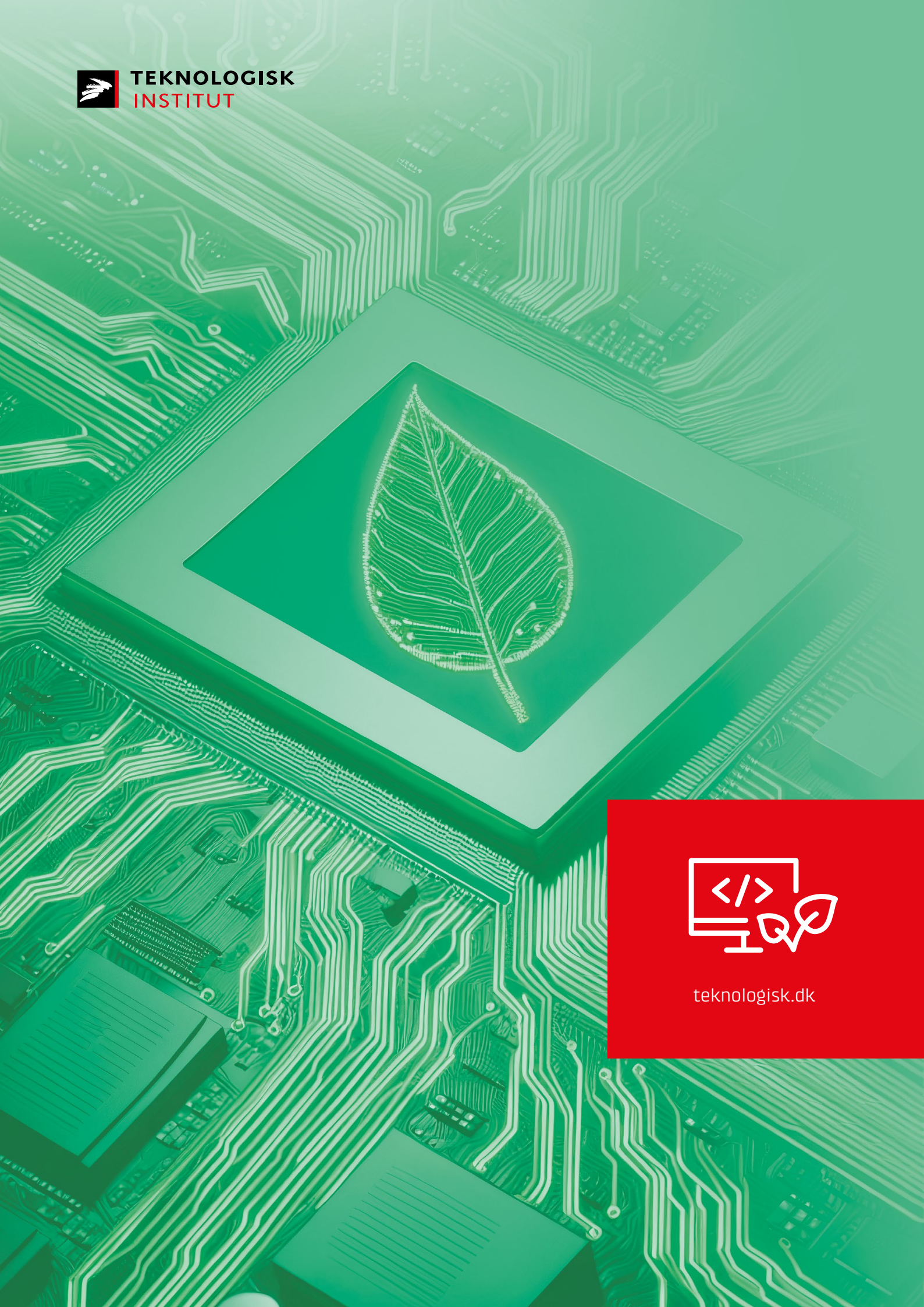
[55] IoT Analytics (2024). State of IoT: Number of connected IoT devices growing to 18.8 billion globally. tinyurl.com/37eupzdc

[56] Der findes allerede forskning på området, se fx Albreem & Jusoh (2021). Green Internet of Things (GIoT): Applications, Practices, Awareness, and Challenges. tinyurl.com/3p6ybz7j

[57] Københavns Universitet (2024). Computer scientists show the way: AI models need not be SO power hungry. tinyurl.com/nhe8rsp5

[58] Bakhtiarifard et al. (2024). EC-NAS: Energy Consumption Aware Tabular Benchmarks for Neural Architecture Search. tinyurl.com/ms37zy9c

TEKNOLOGISK
INSTITUT



teknologisk.dk