Università di Pisa

Dipartimento di Informatica
Corso di Laurea Triennale in Informatica

Corso a Libera Scelta - 6 CFU

Green Computing

Professore:
Prof. Stefano Forti

Autore:
Filippo Ghirardini

Anno Accademico 2023/2024

Contents

Green Computing

Realizzato da: Ghirardini Filippo

A.A. 2023-2024

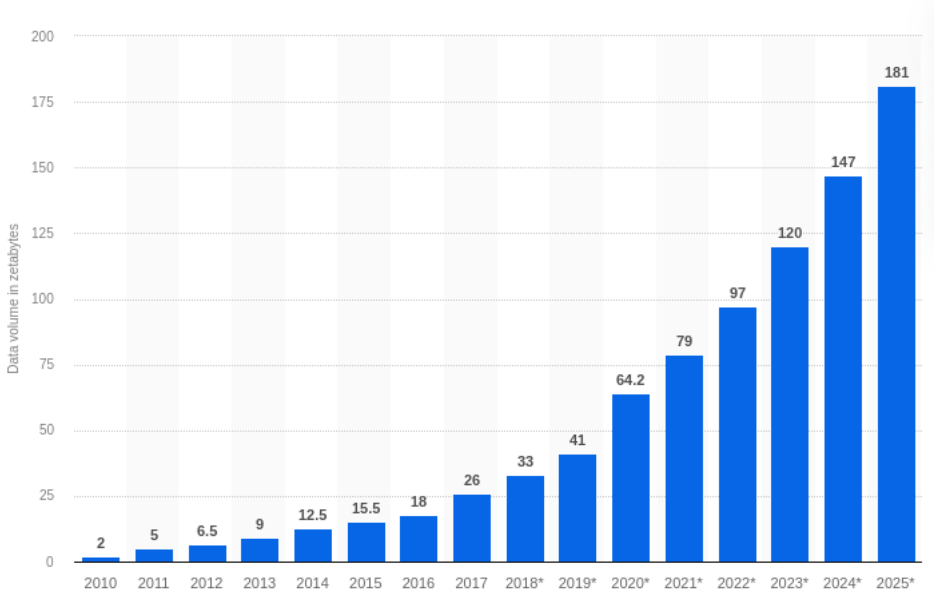
---

1 Introduzione

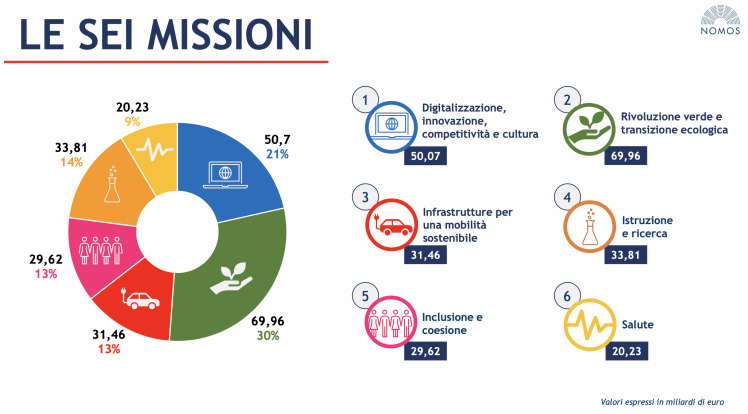
Il corso prevede di affrontare assieme due degli ambiti più importanti al giorno d'oggi: **trasformazione digitale** e **transizione verde**.

1.1 Trasformazione digitale

Con il tempo c'è stata un'evoluzione delle reti di comunicazioni esponenziale grazie alla diffusione di reti pervasive a banda ultra-larga e a basso costo. Inoltre, a causa della diffusione di servizi digitali per la condivisione di dati multimediali ad alta risoluzione, c'è una produzione, un trasferimento ed un consumo di una mole di dati sempre maggiore.

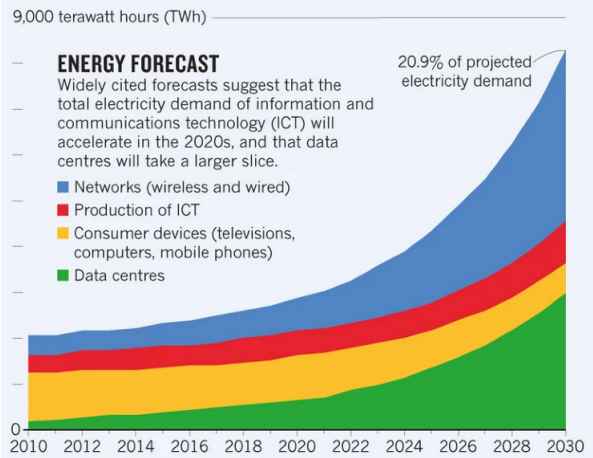


Numerosi sono i programmi di sviluppo nazionali ed europei mirati proprio alla trasformazione digitale, favoriti anche dalla pandemia di Covid-19. Un esempio classico è il PNRR:



1.2 Consumo energetico

Il consumo energetico da parte del settore ICT è ad oggi il 5% della domanda mondiale ed è previsto che superi il 20% nel 2030. La produzione di CO_2 del settore è pari al 2%, quanto quella degli aerei.



1.3 Dennard scaling

È una legge empirica che sostiene che, riducendo la dimensione dei transistor, il rapporto tra potenza e superficie ($watt/cm^2$) rimane costante. Detto altrimenti, dato che l'evoluzione tecnologica consente di impacchettare un certo numero di transistor su una superficie più piccola, allora, a parità di performance, un chip di prossima generazione consumerà minore energia elettrica.

1.4 E-waste

La continua emissione di dispositivi guasti o passati di moda contribuisce ad alimentare i rifiuti di apparecchiature elettriche ed elettroniche (RAEE).
Questi 50 milioni di rifiuti finiscono nei paesi in via di sviluppo dove non vengono smaltite correttamente (danno per l'ambiente e non-biodegradabilità).
Questo va combinato con l'aumento esponenziale dei dispositii elettronici: si prevede che dai 15 miliardi di dispositivi elettronici si arriverà a 18 miliardi entro il 2025, in aggiunta ai quali ci sono gli attuli 13 miliardi di dispositivi IoT (previsit 25 miliardi nel 2028).

1.5 Paris Agreement

Un accordo legalmente vincolante per mantenere il riscaldamento globale ben al di sotto dei $2C$ e idealmente sotto $1.5C$ basato sui seguenti principi:

- *Obiettivo a lungo termine*, con piani quinquennali
- *Contributi* dei vari paesi
- *Ambizione*
- *Trasparenza* sui dati
- *Solidarietà* dei paesi più sviluppati verso quelli in via di sviluppo

Per rispettare l'accordo ogni europeo dovrebbe ridurre le emissioni da 10 a 2 tonnellate di CO_2.

1.5.1 Aziende

Le aziende informatiche sono interessate al green computing per:

- Ridurre i costi di gestione ed aumentare gli utili
- Migliorare la reputazione aziendale verso il personale

- Greenwashing
- Realizzare una trasformazione energetica

2 Green Computing

In generale, il green computing può aiutare le organizzazioni a ridurre l'impatto ambientale e a risparmiare sui costi energetici e di gestione.

Definizione 2.0.1 (Green Computing)**.** *Il green computing tratta la* ***progettazione****, la* ***realizzazione*** *e l'****utilizzo*** *di sistemi ICT, computer e dispositivi elettronici*[1] *in modo responsabile e sostenibile dal punto di vista ambientale, considerando in particolare il* ***consumo energetico*** *e* ***impronta di carbonio****.*

Definizione 2.0.2 (CO_2-eq)**.** *L'anidride carbonica equivalente è una misura che esprime l'impatto di una certa quantità di gas serra rispetto alla stessa quantità di anidride carbonica.*

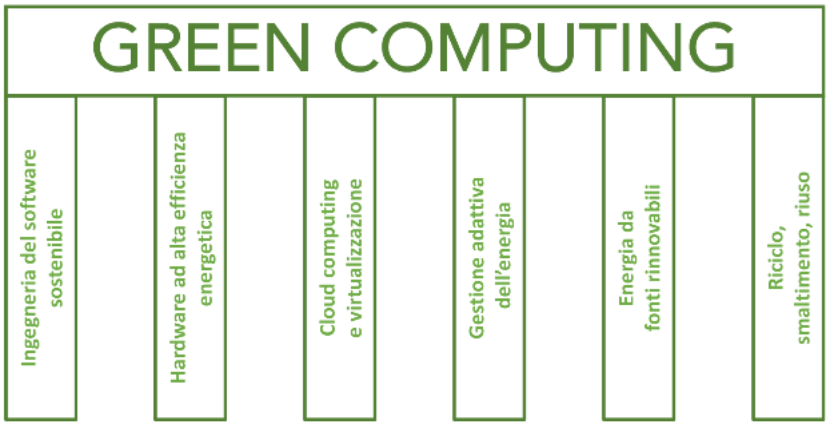
Definizione 2.0.3. *Energy Star Il progetto Energy Star nasce negli anni '90 ed è stata una delle prime iniziative relative al green computing per dare un indicatore dell'efficienza energetica. Il problema principale è che è* ***facoltativo****.*

2.1 Approccio olistico

Per funzionare segue un approccio **olistico**, analizzando tutto il **ciclo di vita** di un sistema, sia *vericalmente* che *orizzontalmente.*

- **Progetto**: progettare in modo sostenibile computer, server, sistemi di raffreddamento e software a basso consumo e alta efficienza.
- **Produzione**: attenzione a non sprecare risorse limitate, ridurre gli scarti di fabbricazione e utilizzare fonti rinnovabili per la produzione.
- **Trasporto**: cercare di ridurre e ammortizzare l'uso di carburanti fossili sostituendoli con veicoli elettrici o ibridi e facendo spedizioni accorpate.
- **Uso**: utilizzare i sistemi cercando di ridurre il consumo con politiche di risparmio (e.g. ibernazione)
- **Dismissione**: lo smaltimento di dispositivi elettronici attraverso il riciclo

2.2 Pilastri fondamentali



2.2.1 Ingegneria del software sostenibile

È possibile fare in modo che i programmi consumino meno energia e che il loro dispiegamento nelle varie fasi del ciclo di vita produca minori gas inquinanti. In particolare, programmare *sfruttando le peculiarità di linguaggi e hardware* che possano rendere il software più disponibile.

[1] Tutti quei dispositivi che si appoggiano all'informatica per funzionare, e.g. aspirapolvere

2.2.2 Hardware ad alta efficienza energetica

L'hardware ad alta **efficienza energetica** o le configurazioni Eco permettono di ridurre l'impatto ambientale e risparmiare sui costi energetici e di gestione. Inoltre, sistemi di raffreddamento più efficienti e processori di nuova generazione consumano meno. Alcune **periferiche** a basso consumo energetico sono:

- Schermi OLED: I pixel dello schermo si spengono su immagini nere risparmiando energia
- Stampanti, scanner, etc.

Va considerato un trade-off tra prestazioni e consumi, con l'obiettivo di avere un minore impatto ambientale

2.2.3 Cloud computing e virtualizzazione

Il Cloud Computing permette di **condividere** le risorse informatiche tra più utenti e organizzazioni, riducendo così gli sprechi di risorse. Inoltre, le grandi aziende che forniscono servizi cloud possono permettersi di investire in infrastrutture tecnologiche più **efficienti** dal punto di vista energetico, per esempio utilizzando fonti rinnovabili per alimentare i propri data center. Inoltre, il cloud computing permette una maggiore **flessibilità** nell'allocazione delle risorse, garantendo l'accesso (anche elastico) solo alle risorse necessarie per un'applicazione specifica. Ciò si traduce in una riduzione del consumo energetico. La virtualizzazione delle risorse consente di **ottimizzare** l'utilizzo hardware, riducendo così gli sprechi.

2.2.4 Gestione adattiva dell'energia

La gestione adattiva dell'energia permette di abbattere i consumi e i costi tramite metodi su più livelli, appiattendo la curva di consumo energetico, risparmiando soldi e risorse e riducendol'utilizzo di energia nelle ore di basso utilizzo. Alcuni esempi sono:

- **Adattività del raffreddamento**: ridurre/aumentare il raffreddamento in base all'utilizzo del processore, può portare fino ad una riduzione del 20% dei consumi
- **Batteria**: aumento della vita della batteria evitando di stressarla utilizzando l'energia della presa quando in carica
- **Sistemi operativi**: Adaptive job scheduling e timing di spegnimento e sospensione dello schermo e del sistema
- **Illuminazione adattiva dei dispositivi mobil**i: la maggior parte dei nuovi dispositivi sono in grado di adattare la luminosità del display in base alla luminosità ambientale

2.2.5 Energia da fonti rinnovabili

La transizione verde ha come pilastro fondamentale il passaggio da un sistema basato per la quasi totalità su fonti energetiche inquinanti a un modello virtuoso incentrato invece su fonti rinnovabili. Utilizzare **fonti rinnovabili** per alimentare i data center e i dispositivi informatici può quindi ridurre significativamente le emissioni di CO2 e contribuire a combattere il cambiamento climatico.

2.2.6 Riciclo, smaltimento, riuso

È possibile **sensibilizzare** l'utente sul giusto uso dei mezzi a sua disposizione e quindi della loro conseguente fine di utilizzo. Ottimizzare l'impiego dei dispositivi porta una determinante longevità, minimizzando quindi il rifiuto.

- Inoltre, per ridurre la produzione di rifiuti è fondamentale **riutilizzare** (ad esempio rivendendo) i dispositivi elettronici ancora validi. In molti casi è sufficiente sostituire componenti degradati (e.g. le batterie) e mantenere il resto. Oltretutto molti dispositivi possono essere considerati obsoleti per certi scopi ma ancora ottimi per altri (e.g. server).

- È fondamentale ingegnerizzare il processo di **smaltimento** in modo da permettere il **riciclo** di parte dei componenti. Ad esempio dalle schede stampate si possono recuperare metalli preziosi come l'oro. La legislazione italiana necessita il corretto trattamento dei rifiuti per ridurre l'inquinamento. Di conseguenza anche la scelta di macchinari e strumenti mirati allo smaltimento è fondamentale per fare in modo che un'azienda possa essere ritenuta green.
- La tecnologia stessa può essere uno strumento potente per **sensibilizzare** il consumatore su queste tematiche e per fargli conoscere le aziende green.

2.3 Applicazioni green

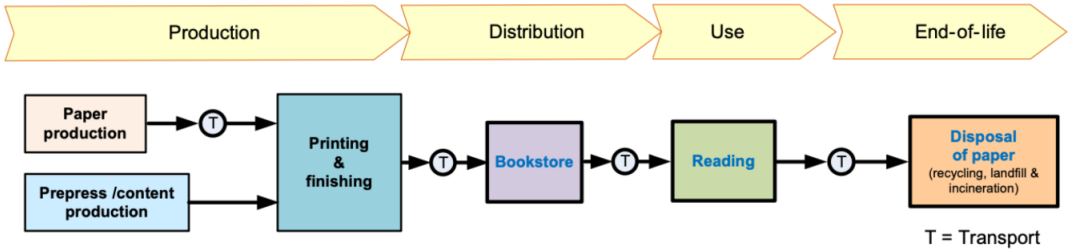
Sfruttare i sistemi ICT per l'**ottimizzazione** di processi che sfruttano risorse limitate (e.g. combustibili fossili nel trasporto, energia elettrica nel riscaldamento, acqua potabile nell'irrigazione) è un aspetto importante del green computing.

2.4 Ebook reader

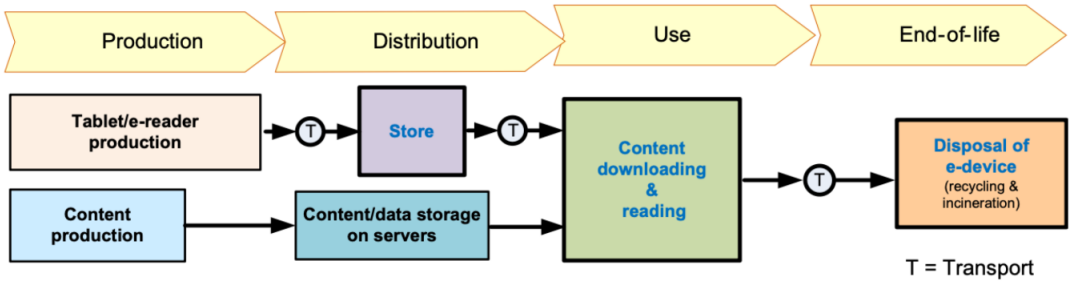
Entro il 2025 si prevede che gli e-reader rappresenteranno circa il 75% del mercato totale, anche se allo stesso tempo il numero di libri cartacei prodotti e venduti è in continuo aumento.

2.4.1 Ciclo di produzione

Vediamo il ciclo di vita di un libro tradizionale cartaceo



Le materie prime necessarie sono, per un libro a copertina morbida, $150-300$g di carta e 7.5lt di acqua. Sono necessari 2KWh e la loro distribuzione (assumendo che non si usi la macchina per comprarlo) produce circa 10 volte quella della produzione. L'utilizzo è trascurabile dal punto di vista energetico in quanto al massimo serve una luce per leggere.



Per quanto riguarda invece gli e-book reader, sono necessari circa 15Kg di materie prime (metalli rari, sabbia, etc...) e 300lt di acqua (batterie, chip, oro dei circuiti). Sono necessari 100KWh per la produzione e assumiamo i costi di distribuzione di un volo Milano-Roma.

2.4.2 Confronto

Considerando i dati precedenti:

1. Quanti libri si producono con le materie prime necessarie per produrre un e-book reader?

$$\frac{15Kg}{0.150Kg} = 100 \quad \frac{15Kg}{0.300Kg} = 50$$

2. Quanti libri si producono con l'acqua necessaria per produrre un e-book reader?

$$\frac{300lt}{7.5lt} = 40$$

3. Quanti libri si producono con l'energia necessaria per produrre un e-book reader?

$$\frac{100KWh}{2KWh} = 50$$

4. Quanti libri serve produrre e trasportare per inquinare quanto per la produzione e il trasporto di un e-book reader?

Produzione e-book reader $= 0.319\frac{g}{Kw/h} \cdot 100Kw/h = 31.9Kg$ Distribuzione e-book reader $= 41.8Kg$

Totale e-book reader $= 31.9Kg + 41.8Kg = 73.7Kg$

Produzione libro $= 0.319\frac{g}{Kw/h} \cdot 2Kw/h = 0.638Kg$ Distribuzione libro $= 0.638Kg \cdot 10 = 6.380Kg$

Totale libro $= 6,380Kg + 0.638Kg = 7.018Kg$

Libri per e-book reader $= \frac{73.7Kg}{7.018Kg} = 10.5$

5. Qual'è la media dei valori delle risposte precedenti (quanti libri vale un e-book reader)?

$$\frac{\frac{100+50}{2} + 40 + 50 + 10.5}{5} = 43.9$$

6. Quanti libri bisogna leggere all'anno per ammortizzare un e-book reader su 5 anni di vita media?

$$\frac{43.9}{5} = 8.8$$

2.4.3 Salute

La produzione di libri ed e-book reader produce ossidi di azoto e zolfo che entrano in profondità nei polmoni, peggiorando l'asma, causando la tosse cronica e aumentando il rischio di morte prematura. Un e-book reader produce 70 volte questi prodotti rispetto che ad un libro cartaceo.

2.4.4 Dismissione

Libro	E-book reader
La **decomposizione** può generare il doppio delle emissioni e degli impatti tossici sulle falde acquifere rispetto alla sua intera produzione	In caso di **smaltimento illegale** in uno dei paesi in via di sviluppo, i lavoratori (spesso bambini) saranno esposti all'impatto tossico di alcune sostanze smantellate.
Può essere prestato, regalato, donato ad una biblioteca oppure correttamente riciclato.	Se correttamente riciclato, molti materiali si potranno recuperare o smaltire correttamente.

2.5 Blockchain

Bitcoin nasce nel 2008 come prima tecnologia basata sulla blockchain.

Definizione 2.5.1 (Blockchain)**.** *Blockchain è un libro mastro distribuito in grado di registrare e validare transazioni in assenza di un'entità centrale (e.g. banca).*

In particolare una blockchain ha le seguenti caratteristiche:

- **Distribuita**: tutti i nodi partecipanti ne conservano una copia per trasparenza
- **Immutabile**: i record nella catena non possono essere né modificati né cancellati
- **Marcata temporalmente**: ogni transazione ha un timestamp
- **Unanime**: tutti i nodi partecipanti devono riconoscere la validità delle transazioni
- **Anonima**: l'identità dei partecipanti non è rivelata
- **Sicura**: tutti i record vengono criptati individualmente
- **Programmabile** per mezzo di SmartContracts

2.5.1 Proof of Work

La blockchain si basa sul concetto per cui ogni blocco, composto dalla transazione e dal riferimento a quella precedente, possa essere aggiunto solo quando viene fornita una **proof of work** da parte dei minatori, che risolvono problemi difficili (e.g. scomposizione in fattori primi).
Quando viene richiesta una transazione si crea un blocco che viene distribuito a tutti i partecipanti.
La difficoltà della proof of work aumenta con l'aumentare delle capacità computazionali dei nodi che scrivono nella blockchain, in modo tale da equilibrare:

- **Sicurezza**: ad esempio evitando attacchi di doppia-spesa, o aggiunta di blocchi falsi
- **Velocità di esecuzione** delle transazioni (stabilita attorno ai 10 minuti)

2.5.2 Hardaware

La potenza hardware per Bitcoin si misura in **GigaHash** al secondo (un hash è un calcolo da risolvere).
L'hardware necessario si è evoluto con il tempo:

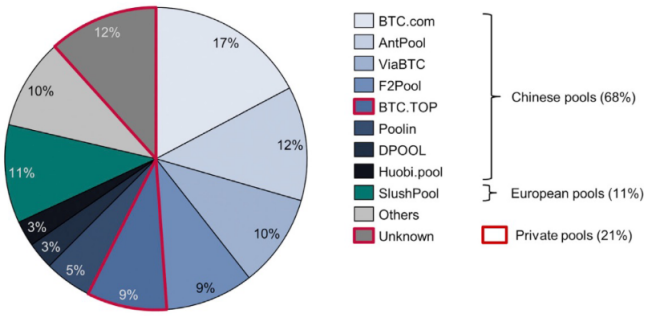
- 2008 - **CPU**, $0.01 GH/s$ con un consumo di $2.5 Wh/GH$
- 2009 - **GPU**, $0.2 - 2 GH/s$

2.5.3 Minatori

Possiamo suddividere le categorie dei minatori in:

- **Piccoli**, il 15% del totale, con un consumo fino a $0.1 MW$ per $0.9 PH/s$
- **Medi**, il 19% del totale, con un consumo tra $0.1 MW$ e $1 MW$ per $9 PH/s$
- **Grandi**, il 66% del totale, con un consumo maggiore di $1 MW$ per oltre $9 PH/s$

I minatori si dividono in **pool** dove condividono il potere di calcolo:



2.5.4 Analisi

Consideriamo che al 2019 il consumo dell'hardware più efficiente era di $1.4 \cdot 10^{-5} Wh/GH$ e che per raffreddarlo veniva utilizzato il 5% del consumo. Il numero di hash eseguiti in un'ora a novembre del 2019 era di $3.56 \cdot 10^{11} TH$. La localizzazione geografica dei minatori era:

- **Cina** con il 68% ad un costo di $0.55 \frac{kgCO_2 - eq}{kWH}$
- **EU** con l'11% ad un costo di $0.28 \frac{kgCO_2 - eq}{kWH}$
- **Privati** con il 21% ad un costo di $0.475 \frac{kgCO_2 - eq}{kWH}$

Considerando queste informazioni

1. Qual è un limite inferiore al consumo energetico annuo di Bitcoin?

   $$\text{Consumo per hash} = 1.4 \cdot 10^{-5}\frac{Wh}{GH} + 5\% = 1.47 \cdot 10^{-5}\frac{Wh}{GH}$$

   $$\text{Consumo per ora} = 1.47 \cdot 10^{-5}\frac{Wh}{GH} \cdot 3.56 \cdot 10^{14}GH = 5.2332 \cdot 10^{9}W = 5.2332 \cdot 10^{6}kW$$

   $$\textbf{Consumo annuo} = 5.2332 \cdot 10^{6}kWh * 8760h = 4.5842832 \cdot 10^{10}kWh$$

2. Quante emissioni di carbonio vengono prodotte all'anno se si utilizza quel limite inferiore come stima?

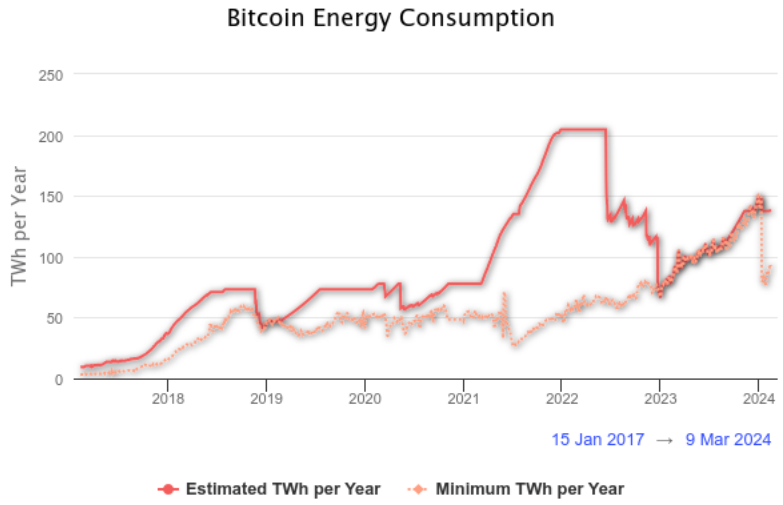
   $$\text{Consumo } \textbf{Cina} = 4.5842832 \cdot 10^{10}kWh \cdot 0.68 \cdot 0.55\frac{kgCO_2 - eq}{kWH} = 1.7145219168 \cdot 10^{10}kgCO_2 - eq$$

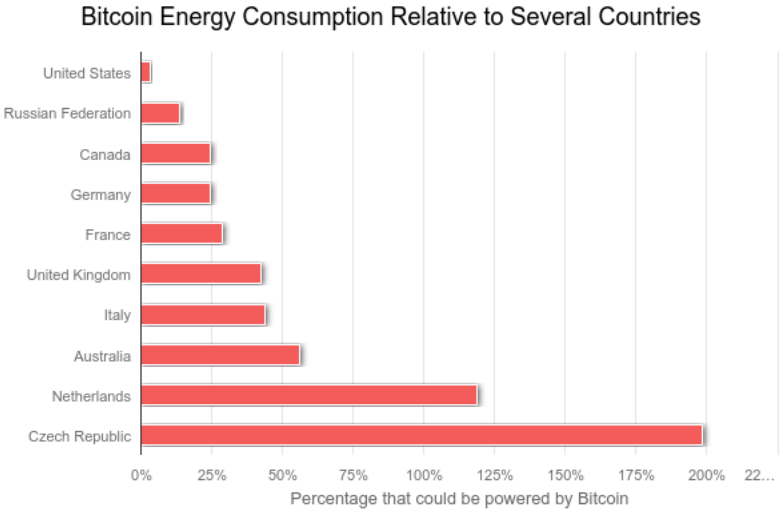
   $$\text{Consumo } \textbf{EU} = 4.5842832 \cdot 10^{10}kWh \cdot 0.11 \cdot 0.28\frac{kgCO_2 - eq}{kWH} = 1.4119592256 \cdot 10^{9}kgCO_2 - eq$$

   $$\text{Consumo } \textbf{privati} = 4.5842832 \cdot 10^{10}kWh \cdot 0.21 \cdot 0.475\frac{kgCO_2 - eq}{kWH} = 4.572822492 \cdot 10^{9}kgCO_2 - eq$$

2.5.5 Oggi

Nella primavera del 2021 alcuni stati come la Cina proibiscono il mining di Bitcoin e questo ha aumentato l'intensità del mining del 43% rispetto al 2019. Si noti che al momento la Cabon Footprint del Bitcoin è di 77.42 Mega Tonnellate di CO_2 ogni anno. In pratica una transazione con Bitcoin equivale a $1,000,000$ transazioni VISA.





Ad oggi esiste il **Crypto Climate Accord** che ha come obiettivo quello di contribuire a raggiungere gli Accordi di Parigi tramite l'utilizzo di energie rinnovabili entro il 2030.

2.5.6 Conclusione

La blockchain è una tecnologia all'avanguardia che potrebbe avere un impatto molto grande su molti settori. È importante eseguire un'analisi di costi e benefici per valutare se conviene o meno:

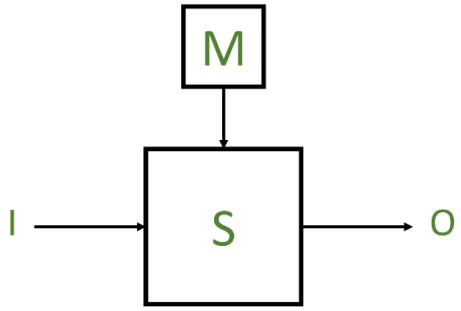
1. **Emissioni** di carbonio
2. Rischi di **centralizzazione**: se qualcuno ottenesse il 51% della computing power avrebbe il controllo della blockchain
3. Possibilità di **controllo** per evitare traffici illegali

2.5.7 Proof of Stake

Per affrontare le problematiche indicate ai punti 1 e 2 si vorrebbe introdurre la **proof of stake**, dove l'abilità di minare è determinata in base alla quantità di moneta che un utente possiede. Il minatore non viene premiato con la moneta al completamento del calcolo ma con degli interessi. In questo modo si evita anche l'attacco del 51% poiché si rende necessario avere il 51% della moneta (più difficile).

3 Performance

Un modello per misurare le performance di un sistema ICT è il seguente:



in cui abbiamo degli **input** elaborati dal sistema che produce **output** è che viene monitorato per la **QoS** (Quality of Service) da un **monitor**.
Il **monitoraggio** di metriche di performance in un sistema ICT è fondamentale per garantirne il corretto funzionamento e necessita quindi di un **modello**. Queste metriche devono essere messe in corrispondenza con una **Quality of Service** da garantire, e viene fatto nel **Service Level Agreement** (SLA).

3.1 Metriche

Definizione 3.1.1 (Metrica)**.** *Si tratta della misurazione di una caratteristica (o di un insieme di esse) di un sistema che fornisce un'informazione utile.*

In sistemi complessi le metriche possono essere combinate mediante:

- **Somma** (e.g. latenza)
- **Moltiplicazione** (e.g. disponibilità)
- **Max/Min** (e.g banda)

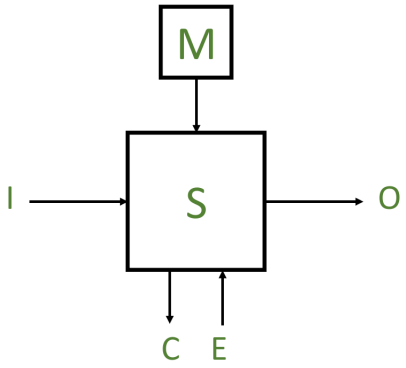
3.1.1 Categorie

Le metriche di performance di un sistema ICT si dividono in 4 categorie:

- **Elaborazione**
  - *Capacità di calcolo*: il massimo numero di richieste per unità di tempo che il sistema riesce a processare senza perdite
  - *Qualità*: la precisione ottenuta dai risultati
- **Trasmissione** dati
  - *Banda*: il massimo throughput di dati supportato da un collegamento all'altro o lungo un cammino nella rete. La *banda disponibile* si ottiene come differenza tra la *banda nominale* e quella attualmente in uso
  - *Latenza*: l'intervallo di tempo in cui si completa il trasferimento di dati lungo un collegamento o un cammino di rete
  - *Jitter*: la variazione di latenza tra richieste successive
  - *Packet loss*: la percentuale di pacchetti non ricevuti sul totale di pacchetti inviati
- **Immagazzinamento** dati
  - *Capacità*: spazio a disposizione per salvare i dati
  - *Throughput*: massima quantità di dati che si riescono a scrivere/leggere per unità di tempo
- **Qualità dell'esperienza** dell'utente. Spesso riportata su una scala da 1 a 5 e può includere *disponibilità*, *framerate*, *qualità video*, etc.

3.2 Efficienza energetica

Dato il modello in precedenza, aggiungiamo l'**energia elettrica** utilizzata e l'**anidride carbonica** prodotta:



Un primo modo per valutare un sistema ICT dal punto di vista *energetico* è tramite l'**efficienza**:

$$\epsilon = \frac{\#calcoli}{E} = \left[\frac{FLOPS}{J}\right] \simeq \left[\frac{FLOPS}{W}\right] \tag{1}$$

Definizione 3.2.1 (Legge di Koomey)**.** *La quantità di calcoli per Joule di energia raddoppia all'incirca ogni* 1.5 *anni.*

La legge si è dimostrata vera fino al 2010 circa. Oggi raddoppia ogni 2.5 anni si fermerà come quella di Moore e quella di Dennard.
Allo stesso modo la quantità di batteria necessaria per svolgere una certa quantità di calcoli diminuisce nel tempo (oggi di 16 volte ogni 10 anni).

3.2.1 Power Usage Effectiveness

L'energia consumata nel mondo ICT si divide in:

- Per i **sistemi IT**
  - Energia per alimentare server e dispositivi di rete
- Per i **sistemi non IT**
  - Sistemi di raffreddamento
  - Sistemi di allarme e UPS
  - Sistemi di illuminazione

Per valutare l'efficienza di un sistema ICT si usa il Power Usage Effectiveness (PUE):

$$PUE = \frac{P_{IT} + P_{non-IT}}{P_{IT}} = 1 + \frac{P_{non-IT}}{P_{IT}} \tag{2}$$

3.2.2 Datacentre Infrastructure Efficiency

L'inverso del PUE è detto Datacentre Infrastructure Efficiency (DCiE):

$$DCiE = \frac{P_{IT}}{P_{IT} + P_{non-IT}} = \frac{1}{PUE} \tag{3}$$

Sia PUE che DCiE sono influenzati da:

- **Utilizzo** del sistema nel *tempo* e nello *spazio*
- **Età** e **progettazione** del sistema
- **Efficienza** complessiva del sistema

Alcuni valori tipici sono:

PUE	**DCiE**	**Level of Efficiency**
3.0	33%	Very inefficient
2.5	40%	Inefficient
2.0	50%	Average
1.5	67%	Efficient
1.2	83%	Very efficient

4 Energia in un sistema ICT

Prendiamo come riferimento il modello **olistico** di *Drouant et al* proposto nel 2014.

4.1 Energia

L'energia si compone di:

$$E = E_i + E_u + Ef \tag{4}$$

dove:

- E_i (iniziale) rappresenta l'energia necessaria per la *progettazione* (E_p), *produzione* (E_m) e *trasporto al venditore* (E_{ti})
- E_u (utilizzo) rappresenta l'energia necessaria per l'*uso* e la *gestione*
- E_f (finale) rappresenta l'energia necessaria al *trasporto al centro di smaltimento* (E_{tf}) e il *riciclo* (E_r)

Ottenendo quindi:

$$E = (E_p + E_m + E_{ti}) + \int_{t=0}^{t=\text{fine vita}} P_u(t)dt + (E_{tf} + E_r) \tag{5}$$

In particolare, l'integrale rappresenta la somma di quanto consuma in ogni istante di vita in un prodotto:

$$P\tilde{U}E \cdot \int_{t=0}^{t=1_{year}} P_u(t)dt \simeq P\tilde{U}E \cdot \sum_{i \in S} \epsilon_i h_i \tag{6}$$

ovvero il *PUE* moltiplicato per la sommatoria della **potenza media oraria** moltiplicato per le **ore** di funzionamento annue di ogni elemento dell'insieme dei componenti hardware del sistema (S).

4.2 Carbonio

Per quanto riguarda invece l'emissione di CO_2 equivalente, la formula prevede, con terminologia analoga:

$$C = C_i + C_u + C_f = (\frac{\alpha_p}{\tau_p} \cdot E_p + \frac{\alpha_m}{\tau_m} \cdot E_m + \frac{\alpha_{ti}}{\tau_{ti}} \cdot E_{ti}) + \frac{\alpha_u}{\tau_u} \cdot \int_{t=0}^{t=\text{fine vita}} P_u(t)dt + (\frac{\alpha_{tf}}{\tau_{tf}} \cdot E_{tf} + \frac{\alpha_r}{\tau_r} \cdot E_r) \tag{7}$$

dove α_j rappresenta l'**intensità** di carbonio per la fase j, misurata in $\frac{gCO_2-eq}{KWh}$ mentre τ_j rappresenta l'**efficienza** dell'infrastruttura elettrica (e.g. in Europa 0.95). Possiamo anche qui trasformare la formula come segue:

$$C = C_i + C_u + C_f = \sum_{S \in \{i,u,f\}} \frac{\alpha_S}{\tau_S} \cdot E_s = \frac{\alpha_i}{\tau_i} \cdot E_i + \frac{\alpha_u}{\tau_u} \cdot E_u + \frac{\alpha_f}{\tau_f} \cdot E_f \tag{8}$$

4.3 Componenti

Il nostro sistema S si compone di due parti:

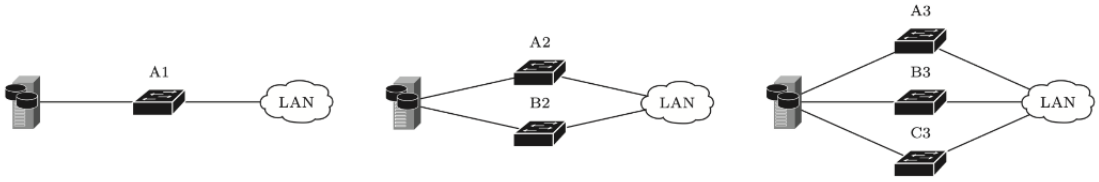
$$S = S_r \cup \tilde{S_r} \tag{9}$$

ovvero componenti **riusabili** e non. Di quelli non riusabili, una percentuale $p_i \in [0, 1]$ rappresenta la parte **riciclabile**.

4.4 Caso di studio

Prendiamo come caso di studio una rete LAN che si connette ad un datacentre privato. Si vuole verificare quale delle tre architetture sotto fornisce:

- **Riciclabilità** sopra al 70%
- **Emissioni** sono i $700Kg$ lungo l'intero ciclo di vita
- **Performance**, ovvero probabilità di fallimento oraria, inferiore a 10^{-6}



I dati del caso sono i seguenti:

- Indici di **riciclabilità** p_i:
  - *Switch*: 0.7
  - *Cavo*: 0.9

  dove durante il ciclo di vita un cavo e uno switch verranno sostituiti e alla fine i cavi saranno riusati e gli switch dismessi.
- **Potenza** assorbita da uno switch in un ciclo di vita di 3 anni (con $\alpha = 0.389 \frac{kgCO_2-eq}{kWh}$):
  - *Costruzione* di uno switch $750kWh$ mentre di un cavo $1kWh$
  - *Smaltimento* di uno switch $400kWh$ mentre di un cavo $1kWh$
  - *Utilizzo* di uno switch $0.05kW$ da sommare a:
    - $0.015kW$ per ogni porta usata al 100% (porte 100 BaseT con uso medio di 10Mbps)
    - $0.006kW$ per ogni porta idle
- Probabilità di **fallimento** di uno switch o cavo ogni ora 10^{-5}
- La **ridondanza** viene utilizzata solo quando necessario ed assumiamo quindi che sia sempre in idle

Di seguito lo svolgimento dei vari casi:

- **Riciclabilità**

$$R_1 = \frac{0.7 + 0.7 + 0.9}{3} = 0.767 = 76.7\%$$
$$R_2 = \frac{0.7 + 0.9 + 0.7 + 0.7}{4} = 0.75 = 75\%$$
$$R_3 = \frac{0.7 + 0.9 + 0.7 + 0.7 + 0.7}{5} = 0.74 = 74\%$$

- **Emissioni**
  - Caso 1:

$$C_{switch_m} = 750kWh \cdot 2 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 614.211kgCO_w - eq$$
$$C_{switch_r} = 400kWh \cdot 2 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 327.579kgCO_2 - eq$$
$$C_{cavi_m} = 1kWh \cdot 3 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 1.228kgCO_2 - eq$$
$$C_{cavi_r} = 1kWh \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 0.409kgCO_2 - eq$$
$$C_{switch_u} = 0.015kW \cdot 2 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} \cdot 8760\frac{h}{y} \cdot 3y +$$
$$+ 0.05kW \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} \cdot 8760\frac{h}{y} \cdot 3y = 860.877kgCO_2 - eq$$
$$C_{tot} = (614.211 + 327.579 + 1.228 + 0.409 + 860.877)kgCO_2 - eq =$$
$$= 1804.304kgCO_2 - eq$$

  - Caso 2:

$$C_{switch_m} = 750kWh \cdot 3 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 921.316kgCO_w - eq$$
$$C_{switch_r} = 400kWh \cdot 3 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 491.368kgCO_2 - eq$$
$$C_{cavi_m} = 1kWh \cdot 5 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 2.047kgCO_2 - eq$$
$$C_{cavi_r} = 1kWh \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 0.409kgCO_2 - eq$$
$$C_{switch_u} = 0.015kW \cdot 2 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} \cdot 8760\frac{h}{y} \cdot 3y +$$
$$+ 0.006kW \cdot 2 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} \cdot 8760\frac{h}{y} \cdot 3y +$$
$$+ 0.05kW \cdot 2 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} \cdot 8760\frac{h}{y} \cdot 3y = 1528.058kgCO_2 - eq$$
$$C_{tot} = (921.316 + 491.368 + 2.047 + 0.409 + 1528.058)kgCO_2 - eq =$$
$$= 2943.198kgCO_2 - eq$$

– Caso 3:

$$C_{switch_m} = 750kWh \cdot 4 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 1228.421kgCO_w - eq$$
$$C_{switch_r} = 400kWh \cdot 4 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 655.158kgCO_2 - eq$$
$$C_{cavi_m} = 1kWh \cdot 7 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 2.866kgCO_2 - eq$$
$$C_{cavi_r} = 1kWh \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} = 0.409kgCO_2 - eq$$
$$C_{switch_u} = 0.015kW \cdot 2 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} \cdot 8760\frac{h}{y} \cdot 3y +$$
$$+ 0.006kW \cdot 4 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} \cdot 8760\frac{h}{y} \cdot 3y +$$
$$+ 0.05kW \cdot 3 \cdot \frac{0.389\frac{kgCO_2-eq}{kWh}}{0.95} \cdot 8760\frac{h}{y} \cdot 3y = 2195.238kgCO_2 - eq$$
$$C_{tot} = (1228.421 + 655.158 + 2.866 + 0.409 + 2195.238)kgCO_2 - eq =$$
$$= 4082.092kgCO_2 - eq$$

- **Performance**: la probabilità che il sistema funzioni è

$$(1 - 10^{-5})^3 = 0.99997$$

e quindi:

$$P_1 = (1 - 0.99997)^1 = 3 \cdot 10^{-5}$$
$$P_2 = (1 - 0.99997)^2 = 9 \cdot 10^{-10}$$
$$P_3 = (1 - 0.99997)^3 = 2.7 \cdot 10^{-14}$$

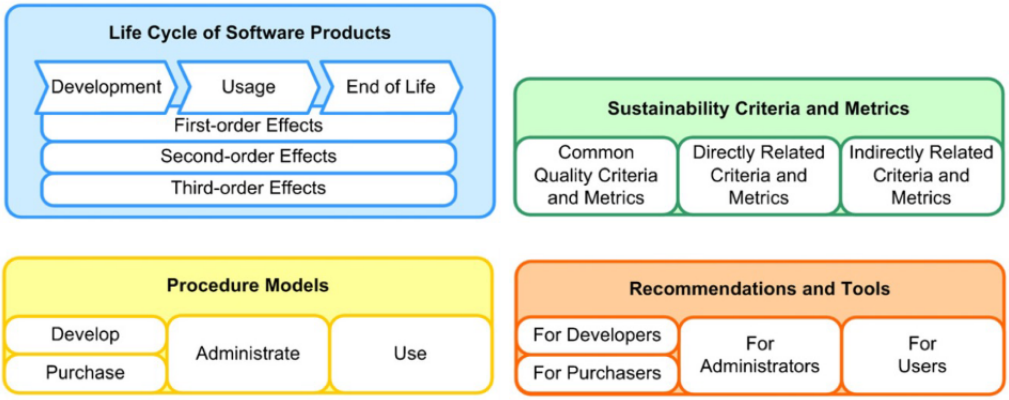
Concludendo, tutte le configurazioni rispettano il punto sulla *riciclabilità*, nessuna quello sulle *emissioni* e solo le ultime due quello sull'*affidabilità*.

5 Ingegneria del software sostenibile

Il digitale aumenta l'impronta ecologica ma può anche contribuire a ridurla rendendo più efficiente l'uso delle risorse.
L'**ingegnere del software** progetta, sviluppa, manutene, testa e valuta prodotti software.
L'obiettivo è quello del **triangolo di ferro**, ovvero coniugare **tempi**, **costi** e **qualità**.

5.1 Modello GREENSOFT

Definizione 5.1.1 (GREENSOFT)**.** *Lo sviluppo software secondo il modello GREENSOFT prevede di utilizzare tecniche che permettano di* ***monitorare*** *gli impatti positivi e negativi sull'ambiente durante tutto il ciclo di vita in modo da garantirne l'ottimizzazione.*

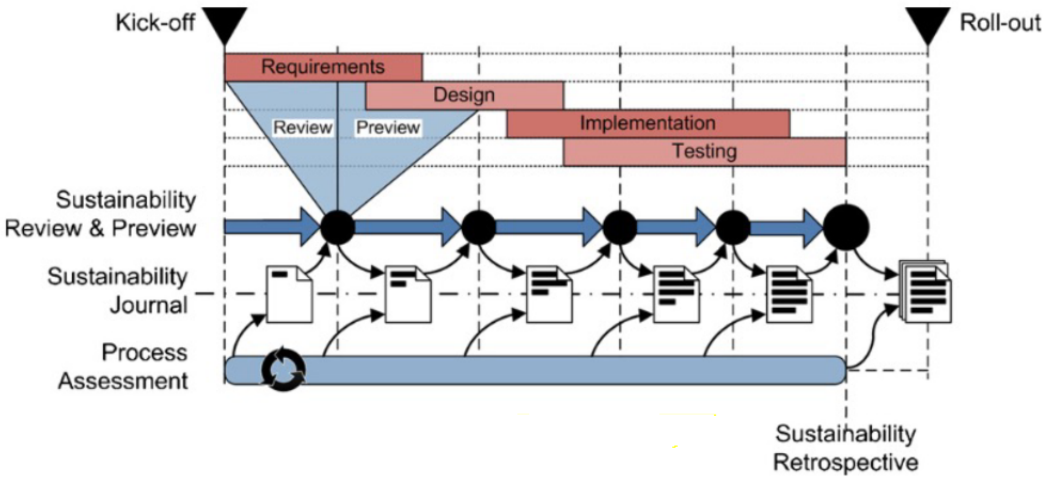


	Development	Usage	End of Life
Third-order Effects	- ... - Changes in software development methods - Changes in corporate organizations - Changes in life style	- ... - Rebound effects - Changes of business processes	- ... - Demand for new software products
Second-order Effects	- ... - Globally distributed development - Telework - Higher motivation of team members	- ... - Smart grids - Smart metering - Smart buildings - Smart logistics - Dematerialization	- ... - Media disruptions
First-order Effects	- ... - Daily way to work - Working conditions - Business trips - Energy for ICT - Office HVAC - Office lighting - ... - Manuals - Transportation - Packaging - Data medium - Download size	- ... - Accessibility - Hardware requirements - Software induced resource consumption - Software induced energy consumption	- ... - Backup size - Long term storage of data (due to legal issues) - Data conversion (for future use) - ... - Manuals - Data medium - Packaging
	Development, Distribution	Usage	Deactivation, Disposal

Questa tabella rappresenta le tre categorie principali degli effetti dello sviluppo software sull'ambiente:

- *First order*: effetti dovuti alla **fornitura** di sistemi ICT (Green IT)
- *Second order*: effetti dovuti all'**utilizzo** di sistemi ICT (Green by IT)
- *Third order*: effetti **sistemici** dell'IT (effetto *rebound*)

che avverrà in maniera **agile**:



5.2 Principi

I principi dell'ingegneria del software sostenibile sono i seguenti:

1. **Carbonio**: costruire applicazioni efficienti dal punto di vista del carbonio
2. **Elettricità**: costruire applicazioni efficienti dal punto di vista energetico
3. **Intensità di carbonio**: consumare elettricità con la minore intensità di carbonio
4. **Embodied carbon**: costruire applicazioni efficienti dal punto di vista dell'hardware
5. **Proporzionalità energetica**: massimizzare l'efficienza energetica dell'hardware
6. **Networking**: ridurre la quantità di dati e la distanza da percorrere nella rete
7. **Modellamento della domanda**: costruire applicazioni consapevoli delle emissioni di carbonio
8. **Misurazione e ottimizzazione**: ottimizzazioni graduali che aumentino l'efficienza complessiva delle emissioni di carbonio

5.3 Caso di studio

Prendiamo come caso di studio gli aggiornamenti di un sistema operativo:

e analizziamo la situazione sia dal punto di vista dell'*utente* che dello *sviluppatore*.

5.3.1 Utente

Supponiamo un aggiornamento di 10 minuti con un consumo medio di $50W$. Abbiamo quindi $50W \cdot \frac{1}{6}h = 8.3Wh$. Se consideriamo un aggiornamento al mese in un anno abbiamo circa $100W$ per ogni utente, che aumenta molto velocemente se consideriamo ad esempio Windows con un miliardo di utenti.

$$0.1kWh \cdot 10^9 = 10^8 kWh = 100GWh$$

che corrispondono a 43259 tonnellate di $CO_2 - eq$, ovvero circa ciò che serve per alimentare 5449 case in un anno.
Dal punto di vista dei costi, l'utente non è particolarmente impattato: se assumiamo un costo medio dell'elettricità di $0.3\frac{}{kWh}$, un utente ogni anno spenderà 0.03 e circa $2h$ di tempo.
Cosa può fare?

- Essere **consapevoli** del proprio impatto ambientali
- Richiedere software di **qualità migliore** per evitare patch
- Richiedere che la **sostenibilità** non alteri i requisiti esistenti

5.3.2 Architetto

Gli architetti sono coloro che determinano i *requisiti funzionali* e non del sistema e che progettano le *interazioni* tra i componenti. Ne l caso di aggiornamenti di un sistema operativo:

- *Funzionali*:
  - Consegna corretta dell'aggiornamento
- *Non funzionali*:
  - 99.9% di disponibilità del servizio
  - Scalabilità
  - Durata di 10 minuti al massimo tra download e installazione

Per rendere gli aggiornamenti più sostenibili gli architetti devono lavorare sui requisiti non funzionali:

- **Scomponendo** i servizi in unità scalabili:
  - **CDN**: mantenere i dati dell'aggiornamento *vicini*, riducendo la *banda* necessaria tramite del *caching*
  - **Microservizi**
- **Bin packing**: è una tecnica che cerca di minimizzare il numero di server per far girare i servizi che compongono il sistema
- **Carbon awareness**: ad esempio Windows Update schedula gli aggiornamenti in base alle emissioni di carbonio di dove si trova la macchina e dall'orario

5.3.3 Sviluppatore

Gli sviluppatori implementano i requisiti specificati dall'architetto ponendo attenzione ad alcune scelte tecniche:

- **Linguaggio** di programmazione: trovare un bilanciamento tra *energia* consumata, *tempo* di esecuzione e *memoria* utilizzata
- **Librerie** con algoritmi efficienti
- **Pattern** efficienti (e.g. *event driven* invece di *polling*)
- **Implementative**: favorire *bin packing* e disaccoppiare software da struttura per facilitare la migrazione

Il programmatore può fornire **metriche** con *KPI* che siano rapporti tra quantità, e.g. $\frac{\text{CPU Usage}}{\text{Updates delivered}}$

5.3.4 Operatore

L'operatore è colui che **gestisce** e **garantisce** la disponibilità dell'applicazione. Deve:

- Offrire **risorse** infrastrutturali adeguate: favorire il *bin packing*
- Utilizzare al meglio le risorse per contenere i **costi** e favorire la **sostenibilità**, ad esempio con auto-scaling
- Garantire **backup e ripristino** nella giusta quantità
- Implementare **logs** e **monitoraggio** (KPI infrastrutturali vs KPI aziendali)

5.4 Quadro normativo

Le leggi e i regolamenti possono favorire una transizione sostenibile tassando le **esternalità negative** (emissioni nel lifecycle) e obbligando alla trasparenza.
L'Unione Europea ha definito all'interno del Green Deal una direttiva sul *Corporate sustainability reporting* che obbliga le grandi aziende a relazionare sulla loro sostenibilità.
A livello globale purtroppo è improbabile che avvenga.

6 Green coding

6.1 Java collections

Le strutture dati forniscono un modo per organizzare e processare le informazioni. Una struttura **CRUD** consente di fare quattro operazioni:

- *Create*: aggiunta di un elemento
- *Read*: lettura di un elemento
- *Update*: aggiornamento di un valore
- *Delete*: eliminazione di un dato

In particolare Java fornisce strutture dati come *List*, *Set* e *Map*. Noi utilizzeremo le prime due. Queste hanno diverse implementazioni con diverse complessità:

Struttura	*Create*	*Read*	*Update*	*Delete*
ArrayList	$O(1)$	$O(1)$	$O(N)$	$O(N)$
LinkedList	$O(N)$	$O(N)$	$O(N)$	$O(N)$
HashSet	$O(1)$	$O(1)$	$O(1)$	$O(1)$
TreeSet	$O(\log N)$	$O(\log N)$	$O(\log N)$	$O(\log N)$

Note 6.1.1. La differenza tra *lista* e *dizionario* è che la prima può contenere duplicati mentre il secondo no.

6.1.1 Analisi

Le strutture di tipo *List* sono meno efficienti di quelle *Set*, però possono contenere duplicati. Se si prevede un mix bilanciato di operazioni *ArrayList* è migliore di *LinkedList* tranne per la modifica e l'eliminazione. Alla fine la struttura più efficiente è teoricamente *HashSet*, ma ad esempio se è necessario tenere gli elementi ordinati conviene *TreeSet*.
È quindi fondamentale scegliere una struttura dati coerente con il caso di utilizzo.

6.2 Calcolo approssimato

Data l'importanza del problema del risparmio energetico, è interessante anche bilanciare all'interno dei linguaggi **energia** e **accuratezza**.

Definizione 6.2.1 (Calcolo approssimato)**.** *Il calcolo approssimato è una qualunque forma di calcolo il cui risultato non è garantito essere corretto.*

EnerJ estende Java con **qualificatori di tipo** per dichiarare dati che possono essere approssimati. Il compilatore poi mappa le variabili approssimate su **memorie a basso voltaggio**, usa **operazioni a basso consumo** e usa **algoritmi efficienti**.
È dimostrata una riduzione del consumo del 15% in presenza di HW specifico.
Il modello di EnerJ è:

- **Sicuro**: il programmatore fa la distinzione tra dati precisi e non, eventualmente forzandolo tramite *endorsement* espliciti
- **Generale**: unifica memoria, calcolo ed operazioni approssimate

6.2.1 Annotazioni

Di default un tipo è **preciso** e si può quindi omettere

```
@Precise
```

È illegale assegnare una variabile approssimata ad una precisa. Ciò previene un flusso diretto da dati che possono essere approssimati a dati che devono essere precisi.

6.2.2 Sintassi

La sintassi di EnerJ è la seguente:

$$
\begin{aligned}
Prg &::= \overline{Cls}, C, e \\
Cls &::= \text{class } Cid \text{ extends } C\{\overline{fd}\,\overline{md}\} \\
C &::= Cid | Object \\
P &::= int | float \\
q &::= precide | approx | top | context | lost \\
T &::= qC | qP \\
fd &::= Tf; \\
md &::= T\ m(\overline{t\ pid})\ q\{e\} \\
x &::= pid | this \\
e &::= null | L | x | new\ q\ C() | e.f | e_0.f := e_1 | e_0.m(\overline{e}) | (q\ C)\ e | e_0 \oplus e_1 | if(e_0)\{e_1\} else\{e_2\}
\end{aligned}
$$

Dove abbiamo che:

- f: field identifier
- m: method identifier
- *pid*: parameter identifier
- *Cid*: class identifier

Note 6.2.1. Si noti che l'overline di un campo ne indica una lista.

6.2.3 Sub-typing

Il sub-typing è una relazione di ordinamento tra quantificatori di precisione.

$$q <_{:q} q' \tag{10}$$

Abbiamo le seguenti regole, dati *precise*, *approx*, *top*, *ctx* e *lost*:

- tutti gli identificatori diversi da *top* precedono *lost*

$$\frac{q \neq top}{q <_{:q} lost} \tag{11}$$

- tutti i qualificatori stanno sotto *top*

$$\frac{\bullet}{q <_{:q} top} \tag{12}$$

- è **riflessiva**

$$\frac{\bullet}{q <_{:q} q} \tag{13}$$

6.2.4 Adattamento al contesto

Nelle **classi** possiamo fare la **combinazione** di qualificatori con il **contesto**:

$$q \triangleright q' = q'' \tag{14}$$

Combinando un qualificatore *context* con *approx* o *precise*, prendo quello scelto, altrimenti se è *context* devo risalire più in alto.

$$\frac{q' = ctx \wedge q \in \{approx, precise, ctx\}}{q \triangleright q' = q} \tag{15}$$

Combinando un *context* da decidere con qualcosa che è troppo in alto per poter decidere (*top* o *lost*): il programma non può essere tipato.

$$\frac{q' = ctx \land q \in \{top, lost\}}{q \triangleright q' = lost} \tag{16}$$

Combinando due qualificatori già definiti, quindi non *context* irrisolti, prendiamo quello più interno nella classe.

$$\frac{q' \neq ctx}{q \triangleright q' = q'} \tag{17}$$

6.2.5 Regole di tipo

Definiamo le seguenti funzioni:

- **FType**: determina il tipo di un campo che stiamo cercando di accedere
- **MSig**: determina la firma di un metodo invocato

Partiamo da un **ambiente statico** ${}^S\Gamma$ che associa variabili locali al tipo dichiarato:

$${}^S\Gamma \vdash e : T \tag{18}$$

e definiamo le seguenti regole:

- **Accesso ad un campo**

$$\frac{{}^S\Gamma \vdash e_0 : q\,C \quad FType(q\,C, f) = T}{{}^S\Gamma \vdash e_0.f = T} \tag{19}$$

- **Assegnamento**

$$\frac{{}^S\Gamma \vdash e_0 : q\,C \quad FType(q\,C, f) = T \quad lost \notin T \quad {}^S\Gamma \vdash e_1 : T}{{}^S\Gamma \vdash e_0.f := e_1 : T} \tag{20}$$

- **If**

$$\frac{{}^S\Gamma \vdash e_0 : precise\,P \quad {}^S\Gamma \vdash e_1 : T \quad {}^S\Gamma \vdash e_2 : T}{{}^S\Gamma \vdash if(e_0)\{e_1\}else\{e_2\} : T} \tag{21}$$

  questa regola ci garantisce:

  1. La guardia sia di un tipo *primitivo preciso*
  2. Sia e_1 che e_2 hanno un tipo comune

6.2.6 Semantica operazionale big-step

Definiamo:

- Una **heap** h che mappa indirizzi i su oggetti, che sono coppie composte da *tipo a tempo di esecuzione* e *valori dei campi*.
- L'**ambiente dinamico** ${}^r\Gamma$ che mappa variabili locali x su valori v

e le seguenti regole:

- **Accesso ad un campo**

$$\frac{{}^r\Gamma \vdash h, e_0 \rightsquigarrow h', i_0 \quad h'(i_0.f) = v}{{}^r\Gamma \vdash h, e_0.f \rightsquigarrow h', v} \tag{22}$$

- **Assegnamento**

$$\frac{{}^r\Gamma \vdash h, e_0 \rightsquigarrow h_0, i_0 \quad {}^r\Gamma \vdash h_0, e_1 \rightsquigarrow h_1, v \quad h_1[i_0.f := v] = h'}{{}^r\Gamma \vdash h, e_0.f = e_1 \rightsquigarrow h', v} \tag{23}$$

- **If**

$$\frac{{}^{r}\Gamma \vdash h, e_0 \rightsquigarrow h_0, (q, {}^{r}\mathcal{L}) \quad \mathcal{L} \neq 0 \quad {}^{r}\Gamma \vdash h_0, e_1 \rightsquigarrow h', v}{{}^{r}\Gamma \vdash h, if(e_0)\{e_1\}else\{e_2\} \rightsquigarrow h', v} \tag{24}$$

$$\frac{{}^{r}\Gamma \vdash h, e_0 \rightsquigarrow h_0, (q, 0) \quad {}^{r}\Gamma \vdash h_0, e_2 \rightsquigarrow h', v}{{}^{r}\Gamma \vdash h, if(e_0)\{e_1\}else\{e_2\} \rightsquigarrow h', v} \tag{25}$$

$$\frac{{}^{r}\Gamma \vdash h, e \rightsquigarrow h', v \quad \tilde{h} \cong h' \quad \tilde{v} \cong v}{{}^{r}\Gamma \vdash h, e \rightsquigarrow \tilde{h}', \tilde{v}} \tag{26}$$

6.2.7 Modello hardware

Si rendono quindi necessarie tecniche per la riduzione del consumo hardware:

- **Voltaggio adattivo**: da *high-power* a *low-power* a seconda del tipo, ottenendo fino al 30% di riduzione del consumo con l'1% di errore in più