GreenICTComp

Kestävät Ohjelmistot
Antti Sipilä, TIEKE ry
Jari Porras ja Laura Partanen, LUT yliopisto

OPETUSHALLITUS RAHOITTAA HANKETTA

LUT University
UNIVERSITY OF EASTERN FINLAND
TURKU AMK
TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

Luennon sisältö

1. Ennakkotehtävän läpikäynti
2. Kestävien ohjelmistojen määrittely
3. Ohjelmistotuotannon prosessit kestävien ohjelmistojen perustana
   a. Kuinka kestävyys ymmärretään organisaatioissa
   b. Kuinka kestävyyttä lähestytään prosesseissa
   c. Kestävyys ohjelmistoprosessin eri osissa
4. Manifestit/julistukset liittyen kestäviin ohjelmistoihin

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

1. Ennakkotehtävän läpikäynti

Ennakkotehtävä: Määrittele mielessäsi kestävä ohjelmisto omien mielikuviesi perusteella, tai tutustumalla verkosta löytyvään materiaaliin (esim. Green Software - Best practices, 10 recommendations for green software development, Building a sustainable future with green software engineering). Tärkeintä on luoda selkeä käsitys millaisia asioita kestäviin ohjelmistoihin pitäisi mielestäsi sisällyttää.

Nyt: Kirjaa Miro-alustalle kestäviin ohjelmistoihin liittyviä osa-alueita/yksityiskohtia, mitä lukemasi materiaali toi mieleen.

Keskustelu: Reflektoidaan omaa näkemystä verraten muiden esille nostamiin osa-alueisiin. Asioita, joita ei itse mieltänyt osaksi kestävää ohjelmistoa

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

GreenICTComp

2. Kestävien ohjelmistojen määrittely - iso kuva

OPETUSHALLITUS RAHOITTAA HANKETTA

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

*“Kestävässä ohjelmistokehityksessä ei tule ottaa huomioon vain **teknisiä vaatimuksia** ja **kestäviä teknologiavalintoja**, vaan myös se, että testaajat ja kehittäjät ottavat omistajuuden **pitkäaikaisesta asiakastyytyväisyydestä**. Kestävyyden kannalta yksi haastavimpia asioita onkin jatkuvasti muuttuviin asiakastarpeisiin mukautuminen, esimerkiksi miten priorisoida ominaisuuksien kehittämistä asiakaspalautteen tai muuttuvien liiketoiminnan tarpeiden mukaan.”*
(Compile.fi)

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Kuvat. Compile

Kertaus: Kestävyyden näkökulmat (ohjelmistonäkökulma)

- **Ekologinen kestävyys** - Ohjelmistoratkaisujen energiankulutus, esim. AI:n vaatima laskenta
- **Taloudellinen kestävyys** - Ohjelmistoratkaisujen kustannustehokkuus asiakkaille ja tuottajille, esim. Apotti
- **Sosiaalinen kestävyys** - Ohjelmistoratkaisujen vaikutukset yhteiskuntaan, esim. AirBnB
- **Inhimillinen kestävyys** - Yksilön hyvinvointi, esim. terveyssovellukset
- **Tekninen kestävyys** - Järjestelmien kyky kehittyä muuttuvien olosuhteiden ja vaatimusten mukaisesti, esim. käyttöjärjestelmät

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

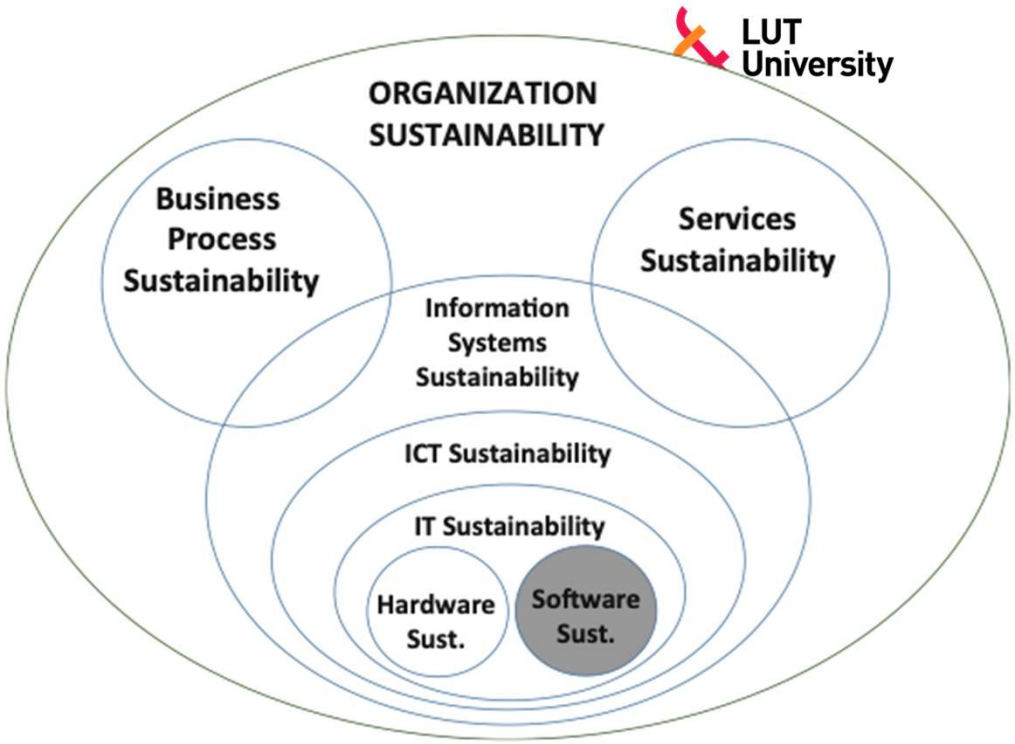
UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Ohjelmistot osana kokonaisuutta

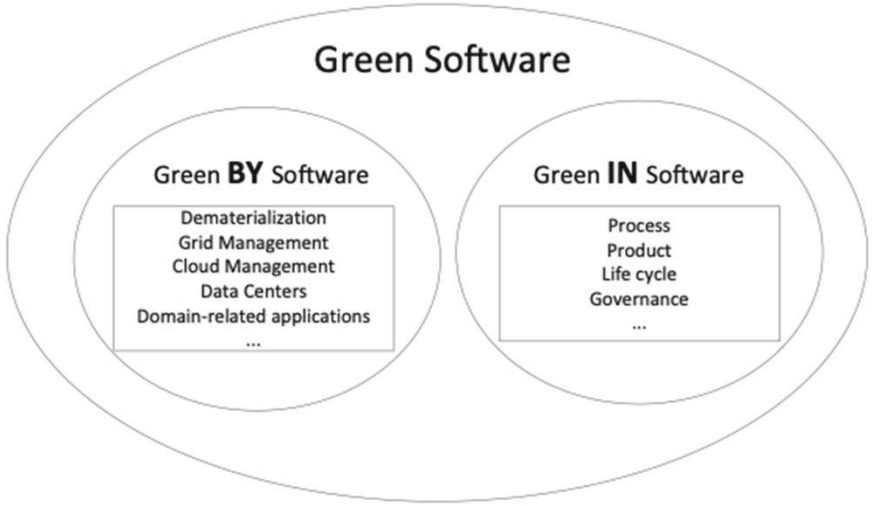
- Huomattava, että ohjelmistot ovat
  - osa isompaa kokonaisuutta
  - linkittyneitä laitteistoon
  - osa yrityksen laajempaan liiketoimintakokonaisuutta
  - tuotteita (tuottaja) tai työkaluja (hankkija)



Calero & Piattini: Introduction to Green in Software Engineering, Green in Software Engineering, Springer, 2015

Ohjelmistojen kestävyys

- Kestävyys, tai “vihreys” jos katsotaan vain ympäristönäkökulmia, jakautuu
  - Jalanjälkeen (IN) hyvinkin eri tasoilla
  - Kädenjälkeen (BY) monilla eri tavoilla



Calero & Piattini: Introduction to Green in Software Engineering, Green in Software Engineering, Springer, 2015

OPETUSHALLITUS RAHOITTAA HANKETTA

Lyhyt reflektio - esimerkkejä (pos./neg.) ohjelmistojen kestävyydestä

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

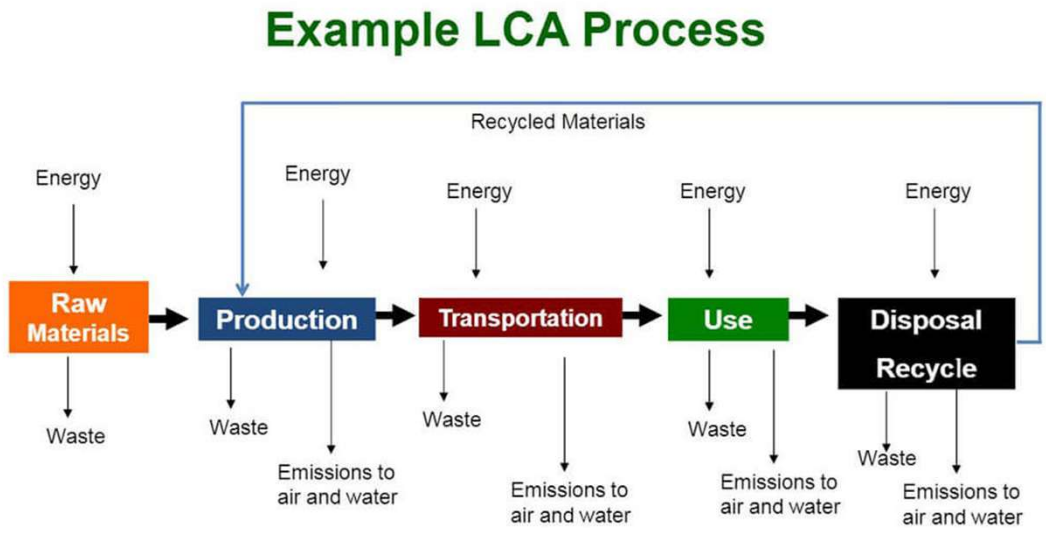
UNIVERSITY OF EASTERN FINLAND

TURKU AMK

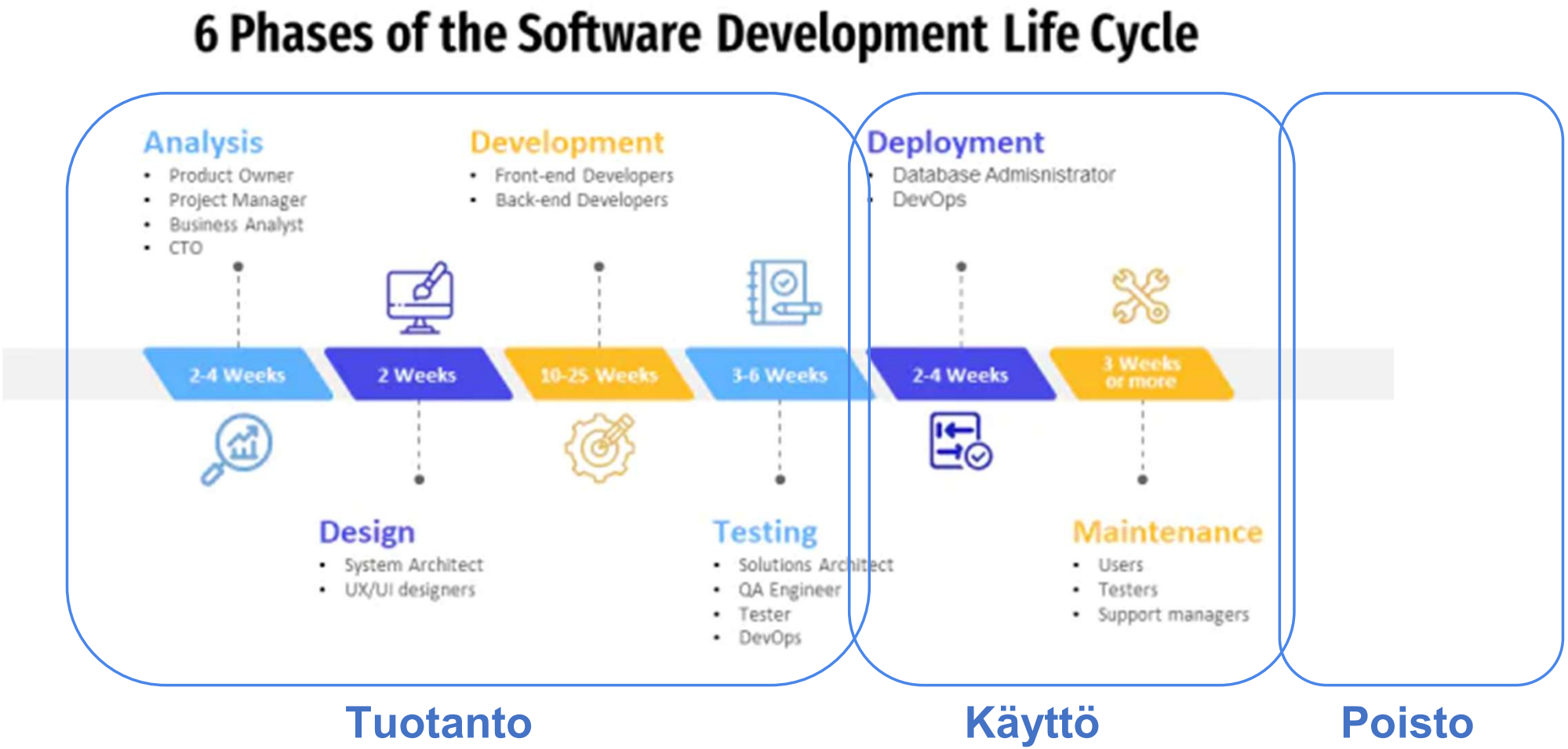
OPETUSHALLITUS RAHOITTAA HANKETTA

Elinkaarianalyysi kestävyyden arvioinnissa

- **Elinkaarianalyysi** (LCA) tarkastelee nimensä mukaisesti tuotteen tai palvelun koko elinkaaren aikana syntyviä ympäristövaikutuksia. (Ecobio.fi)
- Tuotanto - käyttö - hävitys



Kuinka ohjelmistot asettuvat aikajanalle



GreenICTComp

3. Ohjelmistotuotannon prosessit kestävien ohjelmistojen perustana

OPETUSHALLITUS RAHOITTAA HANKETTA

LUT University
UNIVERSITY OF EASTERN FINLAND
TURKU AMK
TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

3.1 Kuinka kestävyys ymmärretään ohjelmistotuotannossa

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

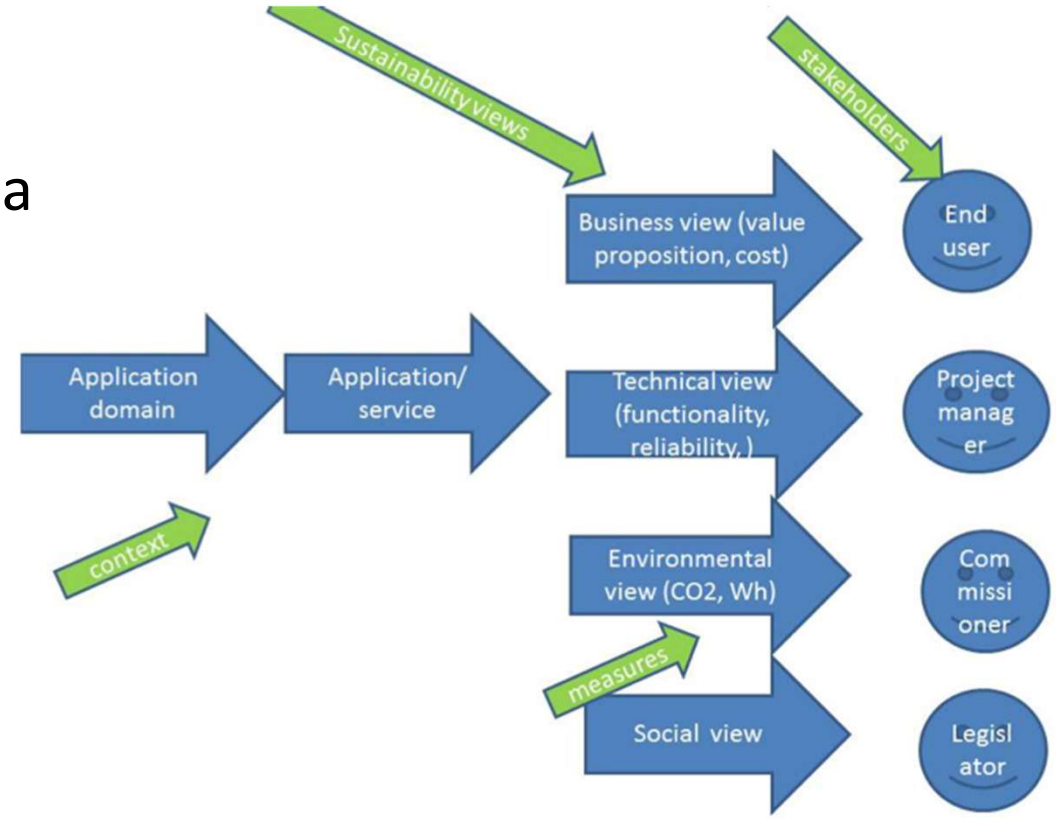
UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Kestävyys osana isompaa kokonaisuutta

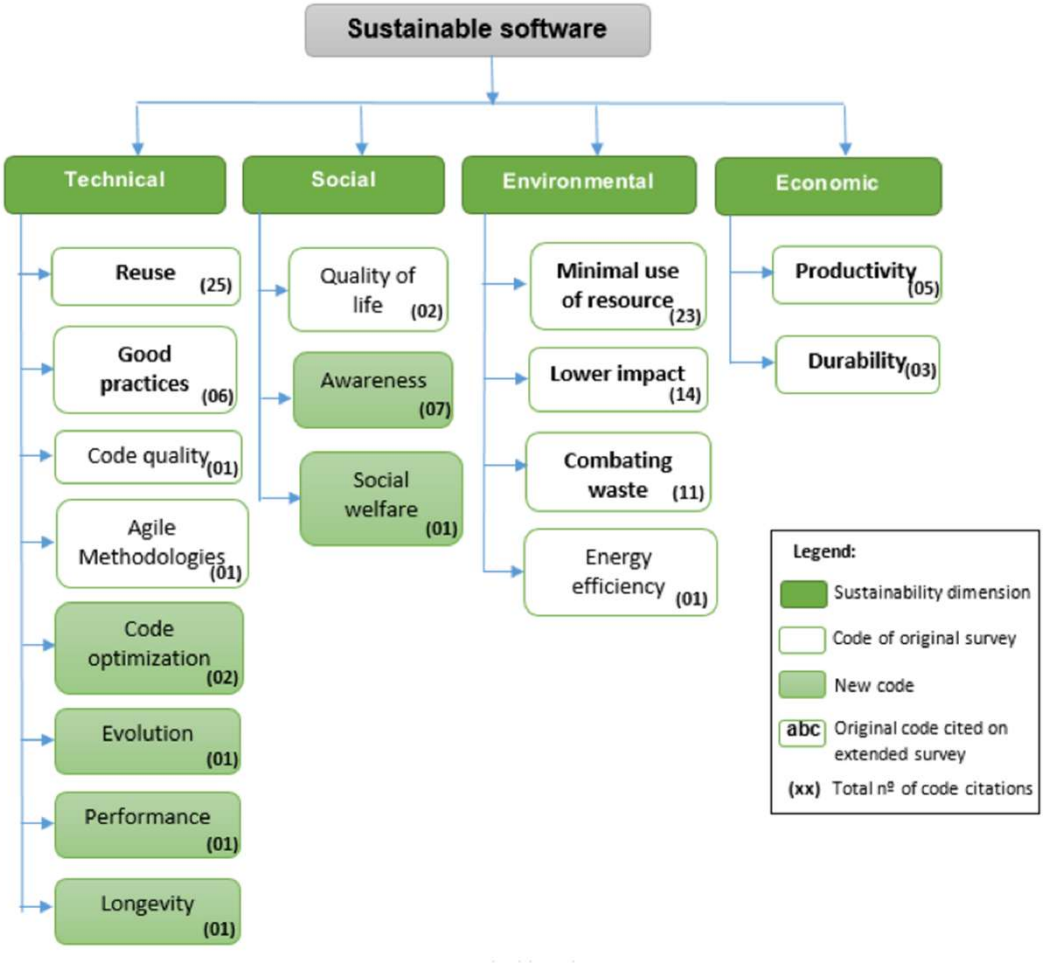
- Moni seikka vaikuttaa kestävyyden määrittelyyn ja toteutukseen yrityksissä
  - Sidosryhmien tarpeet
  - Konteksti
  - Painotettu näkökulma
  - Kriteerit (KPI) ja niiden mittaaminen



Lago et al.: Leveraging “Energy Efficiency to Software Users”, GREENS/ICSE, 2013

Kestävyyden näkökulmat

- Eri kestävyysdimensioilla voi olla hyvinkin erilaisia tavoitteita ohjelmiston toteutukselle ja lopputuotteen vaikutuksille
  - Jalanjälki (tyypillisesti)
  - Kädenjälki (yritykset heräämässä)



Karita et al.: Software industry awareness on green and sustainable software engineering: a state-of-the-practice survey, SBES, 2019

Näkökulmien painotus

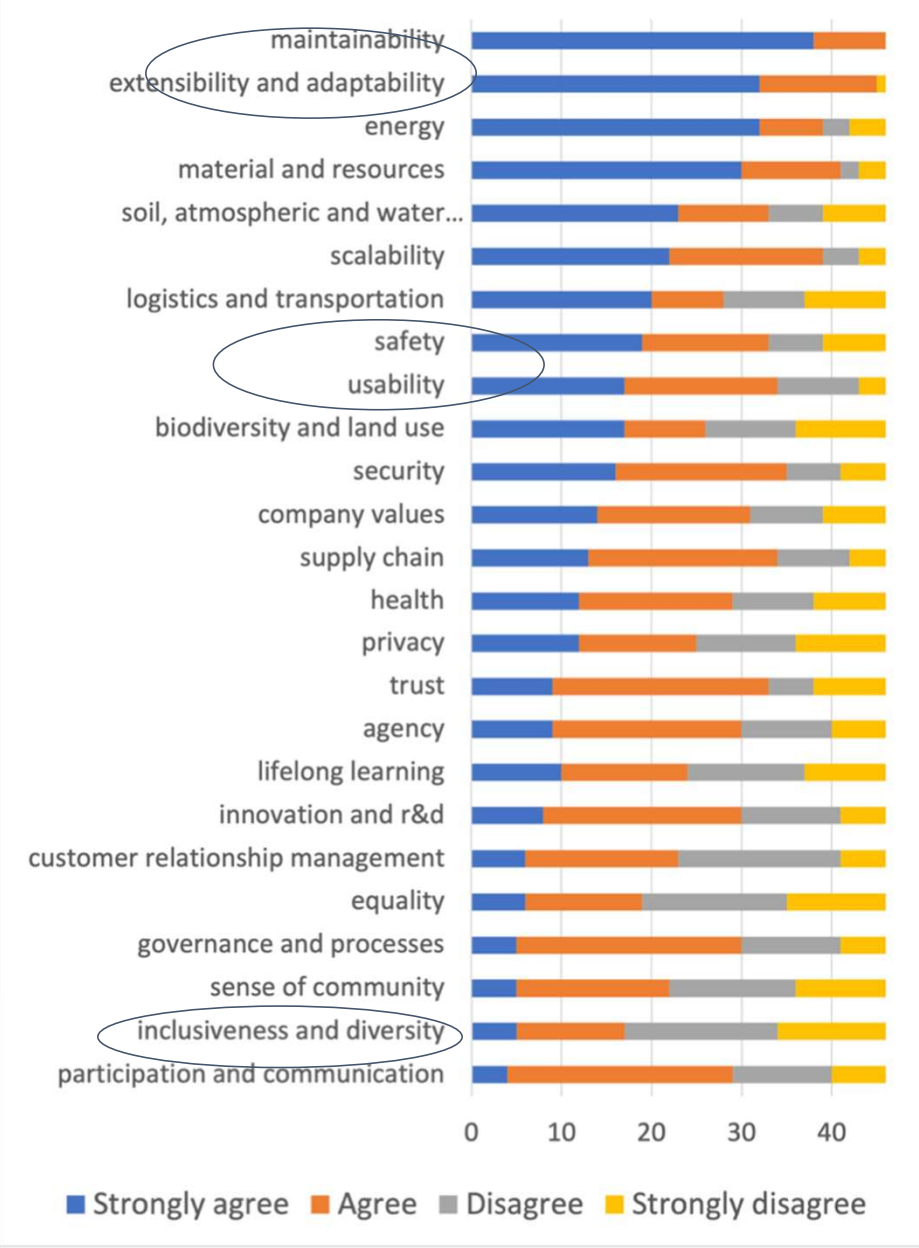
- Yrityksillä voi olla myös näkemyksiä, mikä näkökulma on yrityksen itsensä kannalta paras

Dimension	Target audience
Technical	Software architects, Requirements engineers
Economic	Software architects, Project managers
Social	Requirements engineers, Researchers in ICT & Sustainability ICT practitioners supporting sustainable development
Environmental	ICT practitioners supporting sustainable development Researchers in ICT & Sustainability

Condori-Fernandez & Lago: Characterizing the contribution of quality requirements to software sustainability, JSS, 2018

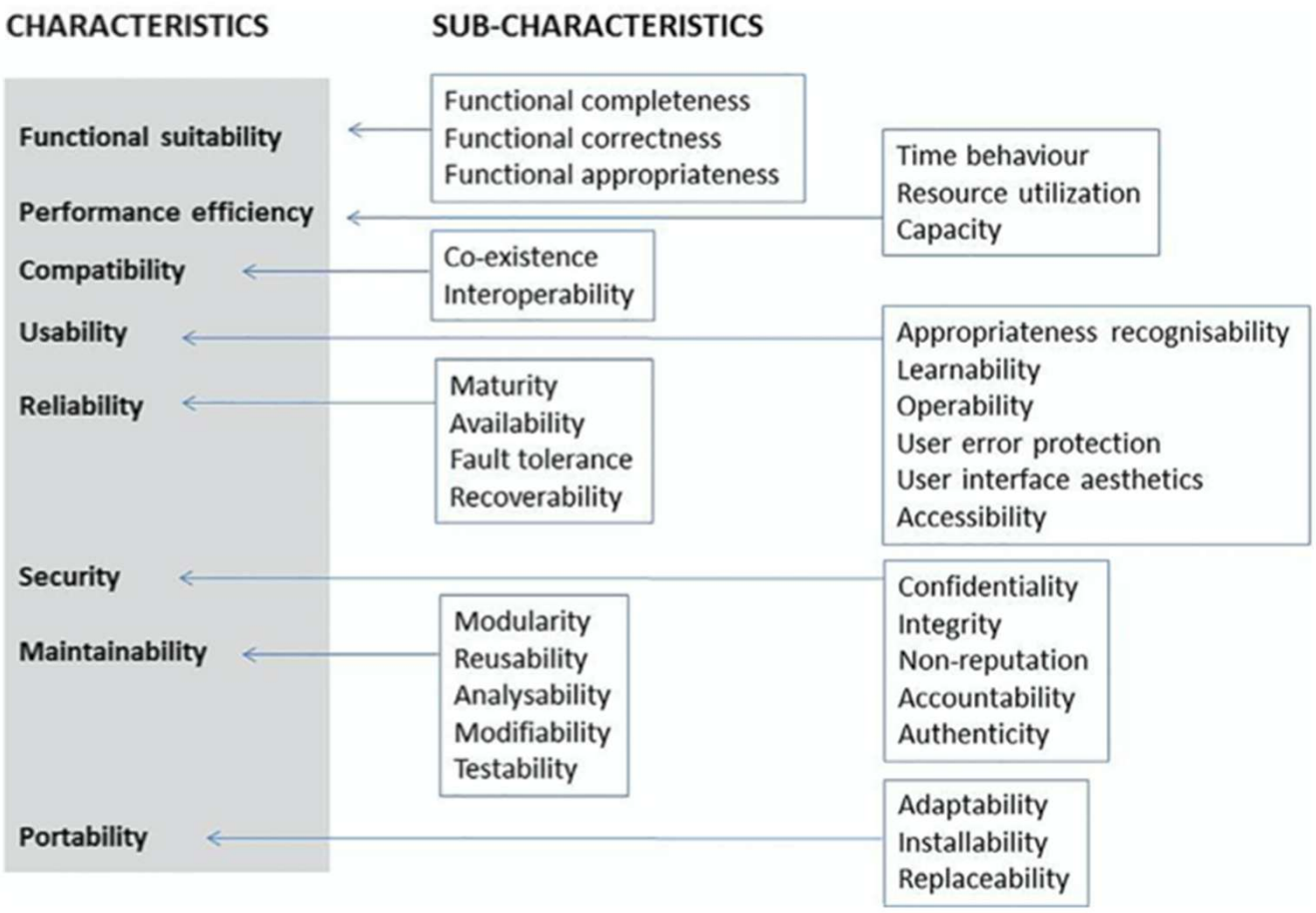
Ohjelmistoyritysten näkemys kestävyydestä

- Ohjelmistoyritysten näkemys eri kestävyysdimensioihin ohjelmistojen vaikutusten näkökulmasta
  - Kertoo, mitä yrityksissä ymmärretään tai arvostetaan ohjelmistojen kestävyydessä



Sustainability in Agile Software Development: A Survey Study among Practitioners (ICT4S 2022)

Ohjelmistojen ominaisuuksia avattuna (ISO/IEC 25010)



TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Condori-Fernandez & Lago: Characterizing the contribution of quality requirements to software sustainability, JSS, 2018

Dimensioiden sisältö ohjelmistotuotannon näkökulmasta

Dimensions	Sustainability concern	Irrelevant (1)	Less important (2)	Neutral (3)	Important (4)	Very important (5)
Technical	Longevity		1		**12**	**12**
Technical	Resilience to uncertainty				11	**14**
Technical	Performance			1	9	**15**
Technical	Software Evolution		1	1	**13**	10
Technical	Reusability				7	**18**
Technical	System Quality				5	**20**
Social	Product Roadmap		1	3	**13**	8
Social	Awareness		1	1	**13**	10
Social	Ethics			3	**11**	**11**
Environmental	Energy consumption	1	1	5	**10**	8
Environmental	Environmental concern		1	6	**13**	5
Economic	Time to Market	1		3	**11**	10
Economic	Development effort			3	**12**	10

Karita et al.: Software industry awareness on green and sustainable software engineering: a state-of-the-practice survey, SBES, 2019

Condori-Fernandez & Lago: Characterizing the contribution of quality requirements to software sustainability, JSS, 2018

	ISO/IEC 25010:2011 Quality model Qualities	Technical	Economic	Social	Environmental
Product quality model	**Compatibility**				
	Co-existence	x	x	x	x
	Interoperability	x	x	x	x
	Functional suitability				
	Functional appropriateness	x	x		x
	Functional corrected	x	x		x
	Functional completeness	x	x		x
	Maintainability				
	Analysability	x	x		
	Modifiability	x	x		x
	Modularity	x	x		
	Reusability	x	x		x
	Testability	x	x		
	Performance efficiency				
	Capacity	x	x		x
	Resource utilization	x	x		x
	Time behaviour	x	x		x
	Portability				
	Adaptability	x	x		
	Installability	x	x		
	Replaceability	x	x		
	Reliability				
	Availability	x	x		x
	Fault tolerance	x	x		x
	Maturity	x	x		x
	Recoverability	x	x		x
	Security				
	Accountability			x	
	Authenticity			x	
	Confidentiality			x	
	Integrity			x	
	Non-repudiation			x	
	Usability				
	Accessibility		x	x	
	Appropriateness recognizability		x	x	
	Learnability		x		
	Operability		x		
	User error protection		x		
	User interface aesthetics		x		
	Context coverage				
	Context completeness	x	x		x
	Flexibility	x	x		x
Quality in use model	**Effectiveness**	x	x	x	x
	Efficiency	x	x	x	x
	Freedom from risk				
	Economic risk mitigation		x		
	Environmental risk mitigation			x	x
	Health and safety risk mitigation			x	
	Satisfaction				
	Comfort	x	x	x	
	Pleasure	x	x	x	
	Trust	x	x	x	
	Usefulness	x	x	x	

"LIKE PERFORMANCE, RELIABILITY, SECURITY,

SUSTAINABILITY DOES NOT JUST HAPPEN

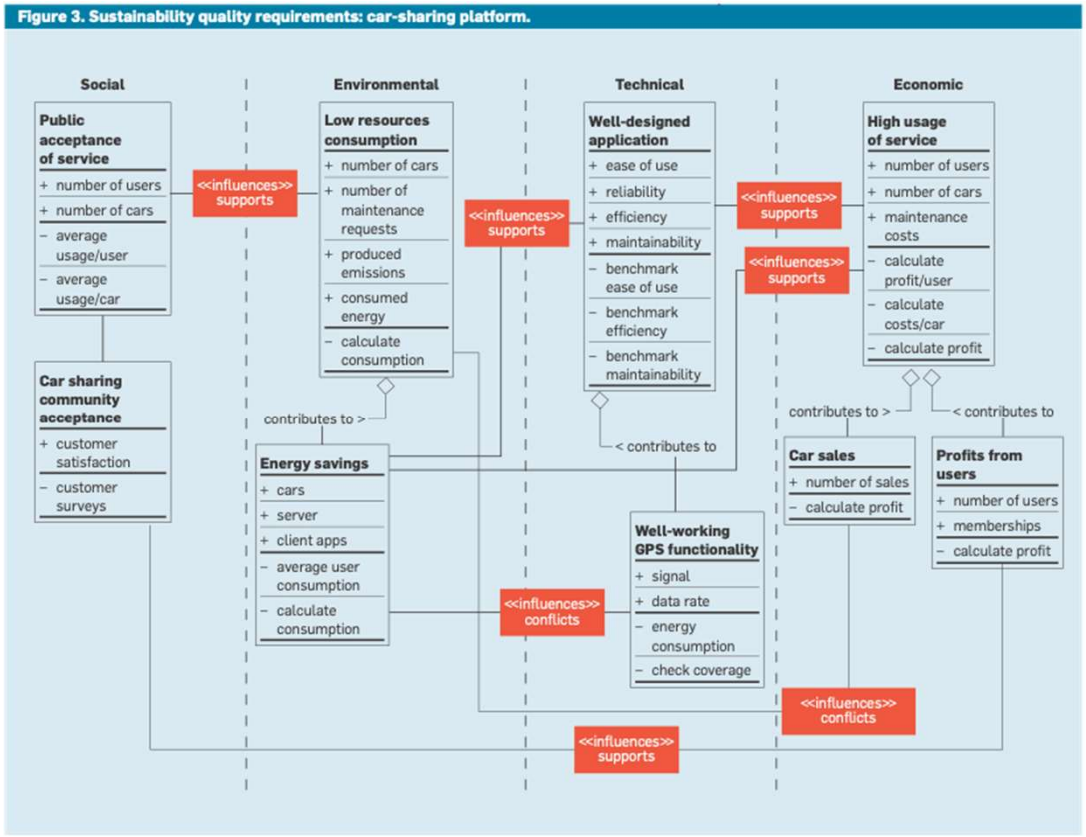
UNLESS WE PLAN FOR IT."

Patricia Lago @ 2016 Inaugural speech

VU
Vrije Universiteit
Amsterdam

Esimerkki yhteiskäyttöautot

- Kestävyys on huomioitava jo ohjelmiston vaatimuksia pohdittaessa, ei valmiin tuotteen kohdalla
  - Fokus
- Kun tunnetaan / tiedostetaan ohjelmiston mahdolliset vaikutukset, voidaan niiden välille luoda riippuvuuksia.
- Esimerkissä luotu myös mittareita eri vaikutuksille



Lago et al.: Framing sustainability as a property of software quality, CACM, 2015

Keskustelu: Kuinka mahdollisesti löydetään tällaisia vaikutuksia?

Millaisia vaikutuksia näette esimerkiksi Uberillä?

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

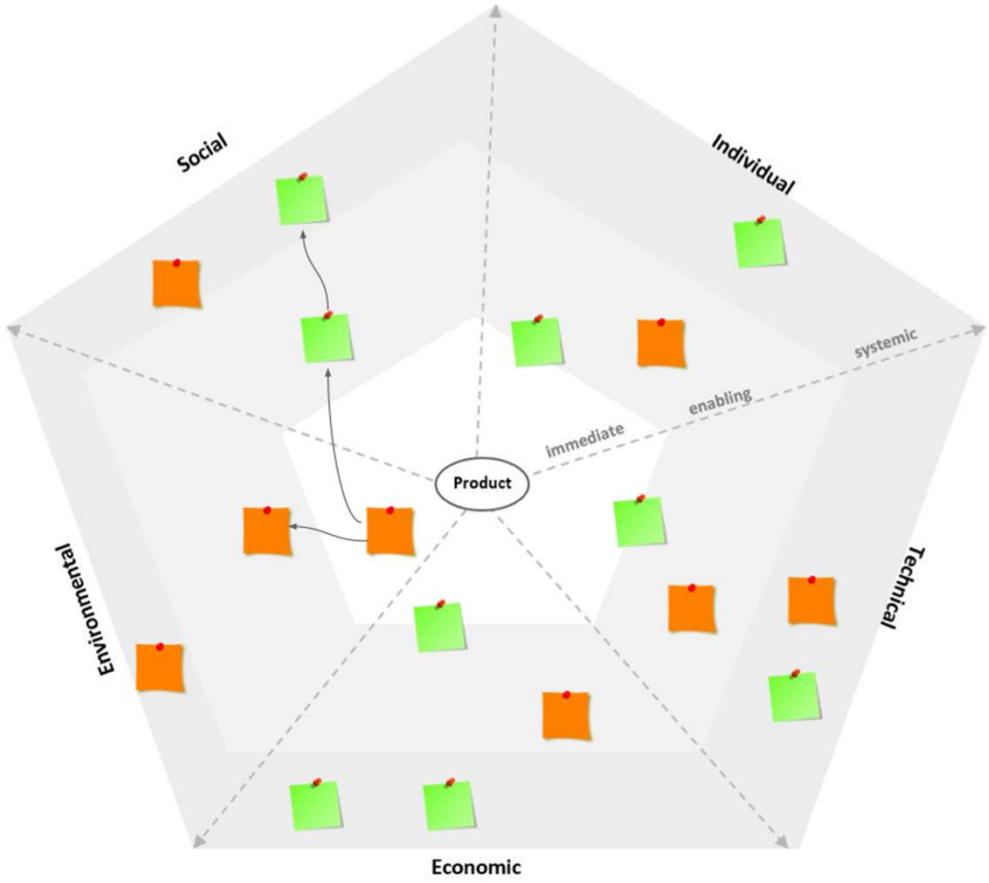
UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

SUSAF – Sustainability awareness framework

- Kysymys-/työpajapohjainen lähestymistapa mahdollisten kestävyyshaasteiden löytämiseen
  - Positiiviset ja negatiiviset
  - Suora, Epäsuora, Rakenteinen
- Vaikutusketjut



Duboc et al.: Do we really know what we are building? Raising awareness of potential Sustainability Effects of Software Systems in Requirements Engineering, RE, 2019

SUSAF esimerkki kysymyksistä

Self-awareness and Free will means the capacity of an individual to act or make decisions on their own.

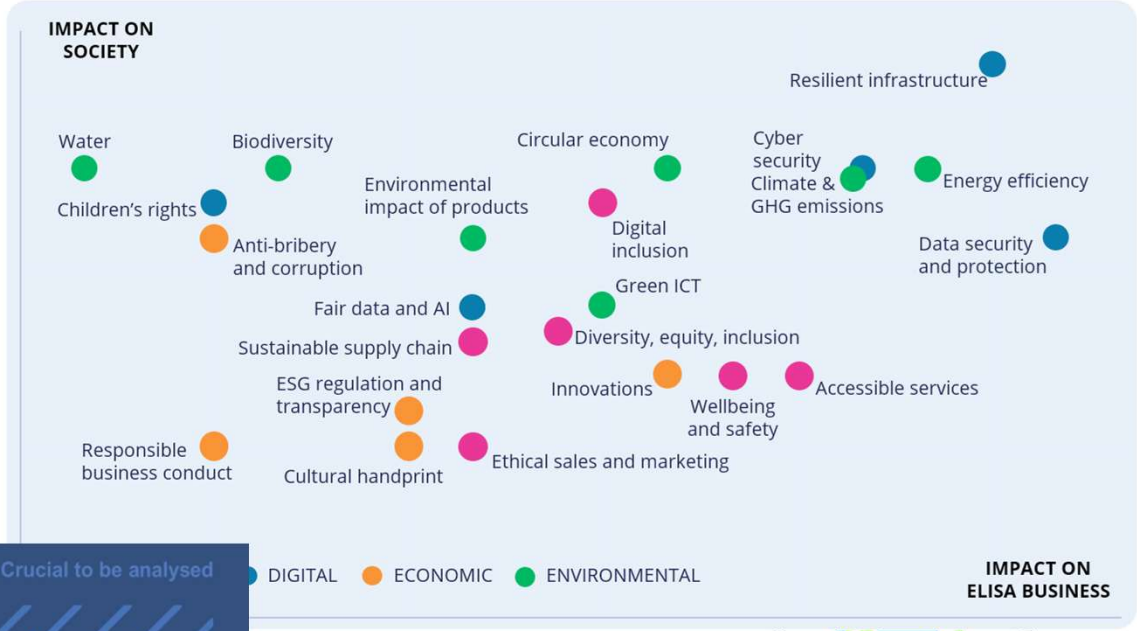
- How can the product or service empower (or prevent) a person from taking an action / decision when necessary?
- Can those affected by the product or service understand its implications, express concerns or be represented by someone?

Inclusiveness and diversity refers to the inclusion of people who might otherwise be excluded or marginalized.

- How can the product or service impact on how people perceive others?
- What effects can it have on users with different backgrounds, age groups, education levels, or other differences?

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY
LUT University
UNIVERSITY OF EASTERN FINLAND
TURKU AMK
OPETUSHALLITUS RAHOITTAA HANKETTA

SUSAF esimerkki



Sustainability Assessment Framework (SAF)

- Kestävyys-laatumalli
- Ohjelmisto-näkökulma
- Tarkoitettu ohjelmisto-yritysten edustajile

Characteristics	Attributes	Definition according to [9]	TECH	SOC	ENV	ECON
Compatibility	Co-existence	product can perform its functions efficiently while sharing environment and resources with other products.				
	Interoperability	a system can exchange information with other systems and use the information that has been exchanged.				
Context coverage	Context completeness	system can be used in all the specified contexts of use				
	Flexibility	system can be used in contexts beyond those initially specified in the requirements.				
Effectiveness	Effectiveness	accuracy and completeness with which users achieve specified goals.				
Efficiency	Efficiency	resources expended in relation to the accuracy and completeness with which users achieve goals.				
Freedom from risk	Economic risk mitigation	system mitigates the potential risk to financial status in the intended contexts of use.				
	Environmental risk mitigation	system mitigates the potential risk to property or the environment in the intended contexts of use.				
	Health and safety risk mitigation	system mitigates the potential risk to people in the intended contexts of use.				
Functional suitability	Functional appropriateness	the functions facilitate the accomplishment of specified tasks and objectives.				
	Functional correctness	system provides the correct results with the needed degree of precision.				
	Functional completeness	degree to which the set of functions covers all the specified tasks and user objectives.				
Maintainability	Modifiability	system can be effectively and efficiently modified without introducing defects or degrading existing product quality				+
	Modularity	system is composed of components such that a change to one component has minimal impact on other components.				
	Reusability	an asset can be used in more than one system, or in building other assets				+
	Testability	effectiveness and efficiency with which test criteria can be established for a system.				
Performance efficiency	Capacity	the maximum limits of a product or system parameter meet requirements.				
	Resource utilization	the amounts and types of resources used by a system, when performing its functions, meet requirements.				
	Time behaviour	response, processing times and throughput rates of a system, when performing its functions, meet requirements				

Condori-Fernandez et al.: Using Participatory Technical-action-research to validate a Software Sustainability Model, ICT4S, 2019

3.2 Kuinka kestävyyttä lähestytään ohjelmistoprosesseissa

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

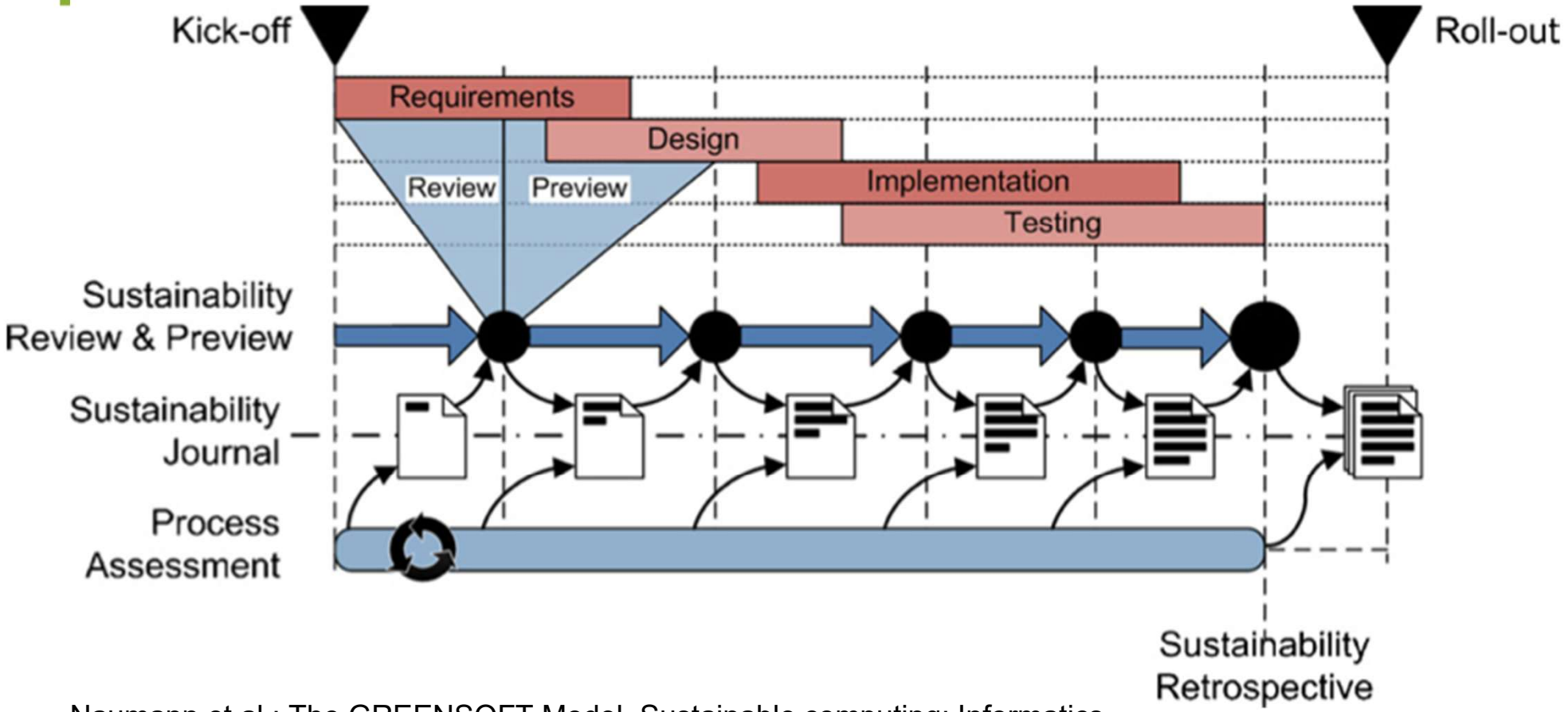
LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

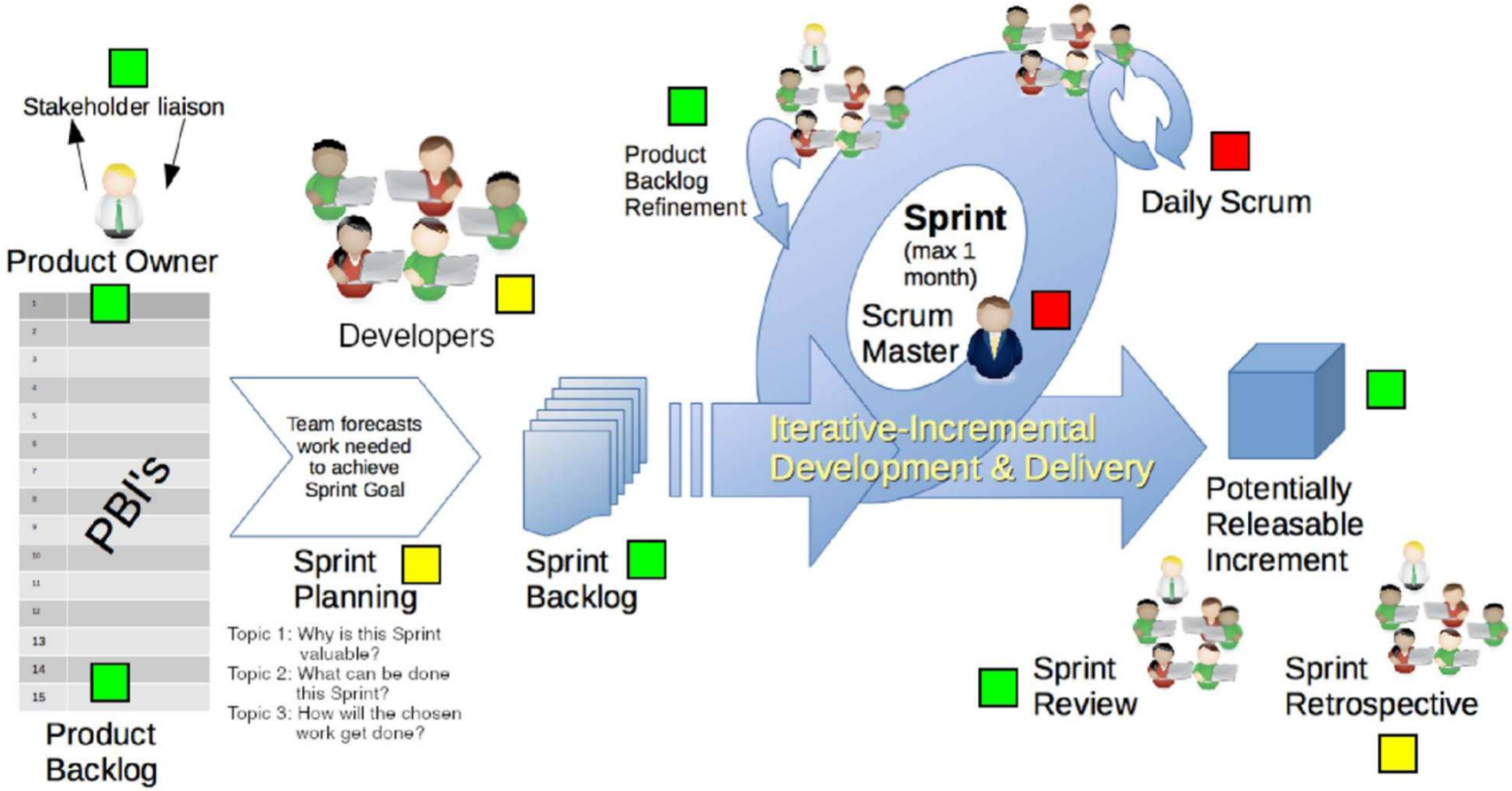
Kestävyys on tuotava osaksi koko prosessia



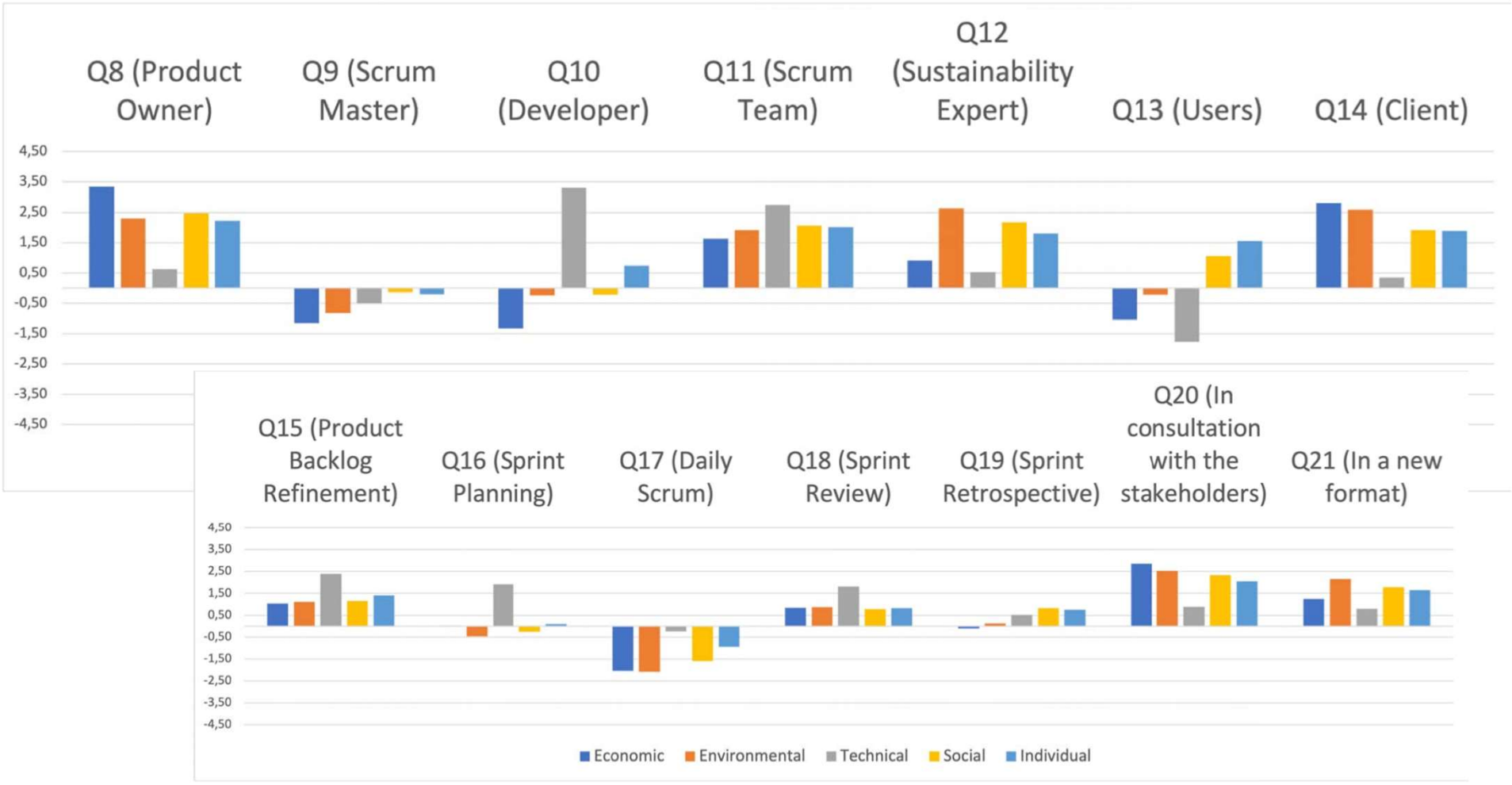
Naumann et al.: The GREENSOFT Model, Sustainable computing: Informatics and Systems, 2011

Kestävyys Agile prosessissa

Sustainability in Agile Software Development: A Survey Study among Practitioners (ICT4S 2022)



Eri dimensioiden vaikutus/näkyvyys agile prosessissa



Esteitä kestävien ohjelmistojen tiellä

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

Level	*Obstacle*	*Mitigation Strategy*
Individual	Lack of Knowledge	Education
	Lack of Experience	Training
	Lack of Methodology and Tool Support	Demonstrators of current methodology and tool applicability; New tool and methodology development
	Resistance to Change	Education on need for Change; Motivation for change adoption
	Fear of Unknown due to Change	Clear evaluation and assessment timelines, criteria, and support provision
Professional Environment	Lack of Higher Management Support	Education, Demonstrators of benefits of Sustainability Design
	Reliance on Customer for Sustainability Requests	Demonstrators of benefits from Software Engineering leadership
	Tradeoffs: Sustainability vs. NFRs	Demonstrators of win-win solutions
	Risk due to change	Stepwise transition support for risk reduction; A roadman with strategies, methodologies, sample case studies
	Fear of client and income loss	Demonstrators of win-win solutions, Experience of past success
	Unavailable Time and Resource	sustainable design into current practice within the available resources; Stepwise transition plans
	Short-termism and income focus	Education, Demonstration of past success
	Poor Communication of Sustainability Values	Embedding sustainability into key values throughout organization, rewarding sustainability inductive practice and innovation
Norms in Professional Practice	Lack of responsibility for long term consequence of software, Sustainability as fundamental ground for software acceptance, Integration of sustainability requirements into SE guidance and practice standards	Review of and integration of sustainability principles within the professional standards, guidance, and accreditation criteria

Sustainability Design in Requirements Engineering: State of Practice, ICSE 2016

3.3. Kestävyys ohjelmistoprosessin eri osissa

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

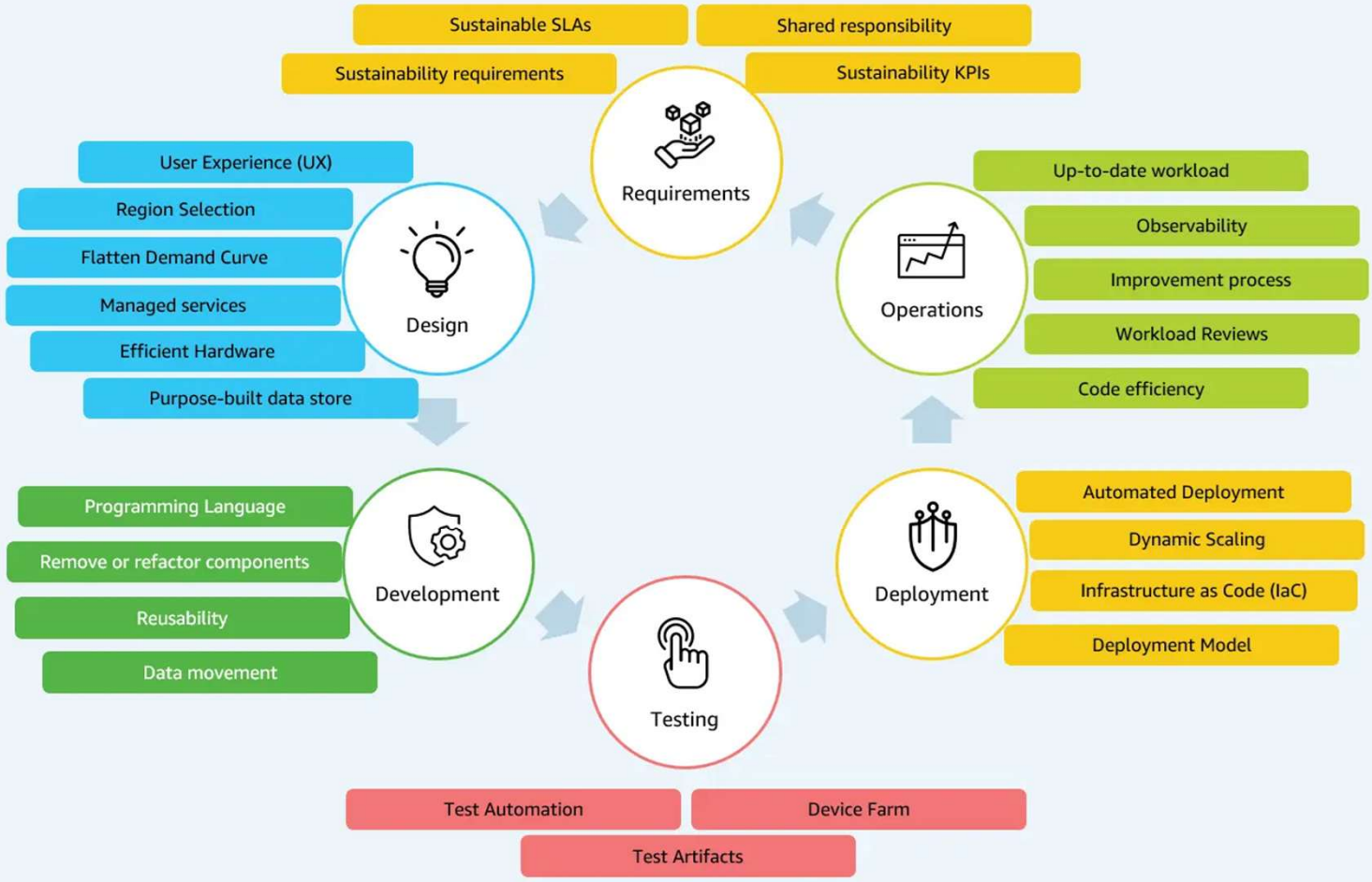
LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Sustainable- Software Development Lifecycle (S-SDLC) Phases



Vaatimusmäärittely

- Kestävyysvaatimukset ohjelmistotuotannossa ja vaatimusmäärittelyssä?
- Määritelty näyttämään erilaiselta kuin vaatimukset yleensä

Area	Key concepts	Motivation	Main actors	Sustainability requirement context
IS	Cost effectiveness Process improvement Process structuring	Improve cost effectiveness of process, aiming for cost reduction.	Business, Regulators, Customers	Metrics and controls context, "such as operating and capital cost, safety, energy cons., waste gen., efficiency"
ICT	Optimisation of IT infrastructure, Green computing, Environmental sustainability, Sustainability of IT services, Longevity of energy systems	Improved resource and energy efficiency of ICT	Customers, employees, business partners, NGOs	Environmental sustainability related to energy consumption and performance
SW Eng	Software development process models	Environmental impacts of ICT	Software developers, administrators, users	Implicit non-functional qualities
Sys Eng	Optimize systems considering sustainability issues	Economic expectations and environmental consciousness	All stakeholders in context, noting they have varying background	Sustainability requirements have to be communicated
Ergonomics	Multi-dimensional understanding with economic, social, and environmental	Economic and business-strategic aspects, human factors	Wide range of stakeholders, including all designers	Environmental context and long life cycles
RE	Multi-dimensionality of sustainability, Interdependence of dimensions, Trade-offs, General models of sustainability	Make sustainability more tangible, Make related goals explicit, Assess sustainability	Decision making households and/or software professionals, regulators	Multiple dimensions and trade-offs: 'Achieve acceptable level of service (...), have min. impact on natural env., be socially and economically acceptable'

Venters at al.: Characterising Sustainability Requirements, SEIS/ICSE, 2017

Palvelusopimukset – SLA

- SLA on palvelun tarjoajan lupaus palvelun käyttäjille palvelun laadusta
- Erilaisilla palveluilla erilaisia kriteereitä / lupauksia
- Tyypillisesti kriteereille määritetty mittarit

Sl.No.	Performance Indicator Name		Unit
1.	Network Availability	Connectivity (IPPM)	% (Percentage)
		Functionality	
2.	Delay	One way delay	Time in Milliseconds
		RTT delay (Round Trip Time)	
3.	Latency		Time in Milliseconds
4.	Packet Delivery Ratio(PDR) or Packet Loss Ratio(PLR)		% (Percentage)
5.	Jitter		Time in Milliseconds
6.	Congestion		% (Percentage)
7.	Flow Completion Time (FCT)		Time in Milliseconds/ Seconds
8.	Response Time		Time in Milliseconds
9.	Bandwidth		Hertz (Hz)
10.	Utilization		% (Percentage)
11.	LAN/WAN period of Operation		Time in Milliseconds/ Seconds
12.	LAN/WAN Service Time		Time in Milliseconds
13.	MTBF (Mean Time between Failure)		Time in Milliseconds
14.	MTRS (Mean Time to Restore Services)		Time in Milliseconds
Sl.No.	**Performance Indicator Name**		**Unit**
15.	Solution Times		Time in Seconds/Minutes/ Hours
16.	Internet access across Firewall		YES/NO
17.	RAS (Remote Access Service)		YES/NO
18.	Resolution Time (TTR)		Time

Esimerkki pilvipalveluiden palvelusopimuksesta

- Pilvipalvelut (verkosta hankittavia resursseja) erittäin hyvä kohde palvelusopimuksille
- Kuitenkin, yleensä vain palvelun saatavuus kriteerinä
  - Amazon - Uptime
  - Google - Uptime

Analysis on Existing Basic SLAs and Green SLAs to Define New Sustainable Green SLA, International journal of advanced computer science and application, 2015

Sl.No.	Performance Indicator Name		Unit
1.	Broad Network Accessibility		% (Percentage) Or YES/NO
2.	Multi-tenancy		YES/NO
3.	Rapid Elasticity		% (Percentage)
4.	Scalability		% (Percentage)
5.	Resource Pooling Time		Time in Milliseconds Or Seconds
6.	Solution Time		Time in Seconds/Minutes/ Hours
7.	Response Time		Time in Milliseconds Or Microseconds
8.	Availability	MTBF MTTR	Time in Milliseconds Or Seconds
9.	Capacity		Number Or Request per Minutes
10.	Virtualization		% (Percentage)
11.	Delay		Time in Milliseconds
12.	Service Time		Time
13.	Logging & Monitoring		YES/NO
14.	Resolution Time (TTR)		Time

Vihreitä SLA kriteereitä/mittareita

Green Computing Domain	Performance Indicator Name	Unit
Energy/ Power	Total Power Consumption [26, 41]	kW-h (Kilowatt-hour)
	PUE (Power Usages Effectiveness) [24, 35, 37, 42]	Number (1.0 to ∞) Or Dimensionless
	DCiE (Data Center Infrastructure Efficiency) [24, 38, 42]	% (Percentage)
	CPE (Compute Power Efficiency) [35]	Watts
	SPECPower [24, 35]	Watt
	JouleSort [26]	kW/J
	WUE (Water Usages Effectiveness) [35]	Liter/kW-h
	TDP (Thermal Design Power) [42]	Watts
	ERF (Energy Reuse Factor) [35]	Number [0 to 1.0]
	ERE (Energy Reuse Effectiveness) [35]	Number [0 to ∞]
	GEC (Green Energy Co-efficient) [35]	Number [0 to 1.0]
	ITEE (IT Equipment Energy Efficiency) [43]	% (Percentage)
	ITEU (IT Equipment Utilization) [43]	Number
	HVAC (Heating, Ventilation, Air-conditioning) Effectiveness [42]	Dimensionless
	Cooling System Efficiency [42]	kW/ton
Carbon footprint	CUE(Carbon Usages Effectiveness) [35]	KgCO2 per kW-h
	DPPE (Data Center Performance Per Energy) [43]	Number [0 to 1]
Recycling	e-Wastage Or IT Wastage [42]	Gm (Gram)
	Recycling [37,44]	% (Percentage)
Productivity	DCP (Data Center Productivity) [35]	Not Available
	DCeP (Data Center Energy Productivity) [24,35]	Not Available
	Analysis Tool [26]	Not Known
	EnergyBench [26]	Numeral Rating
	ScE (Server Compute Efficiency) [35]	% (Percentage)
Costing Information	Energy/Power Cost [41]	Currency [according to country]
Others	SWaP (Space, Wattage and Performance) [24, 35]	Not Available
	User Satisfaction [11, 24]	Number [0 to 5]
	Mean Opinion Score (MOS) [11, 24, 45]	Number [1 to 5]
	Reliability [24]	Number [0.0 to 1.0]
	Air Management Metric [42]	F (Fahrenheit)
	UPS System Efficiency [42]	% (Percentage)
	Risk Assessment [11, 24]	% (Percentage)

Mitä on design?

“Design is about making choices.”

“*Sustainable* design is about making *sustainable* choices.”

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

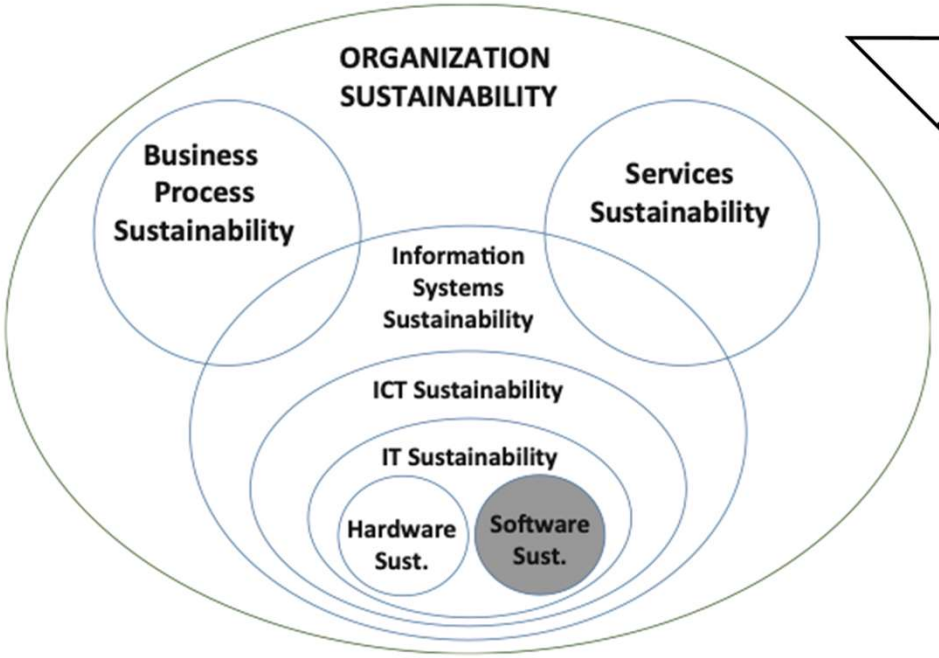
LUT University

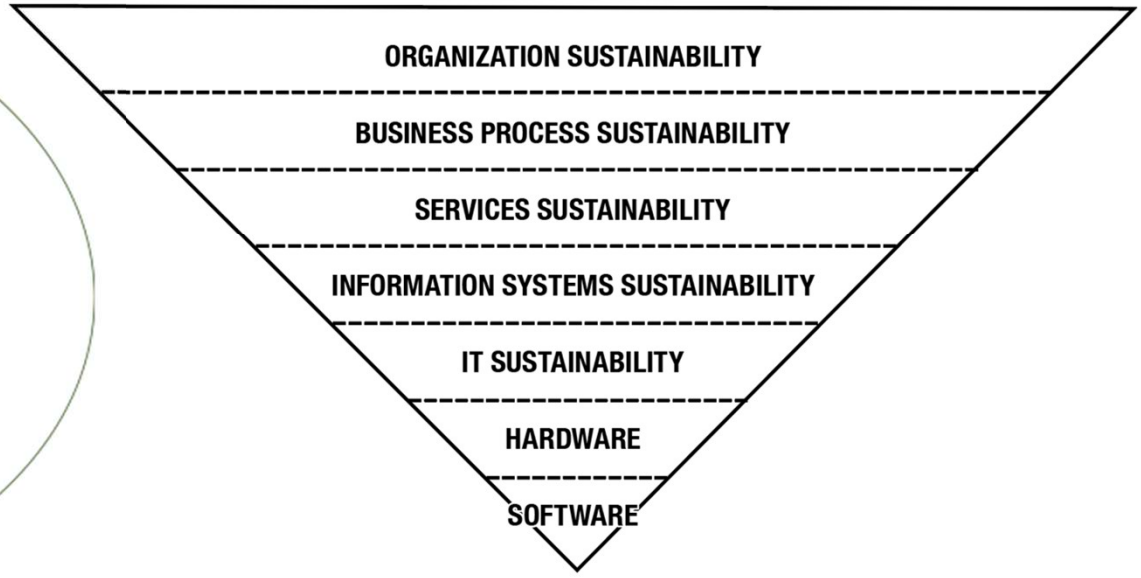
UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Kestäviä valintoja





Kestävän suunnittelun periaatteet

- Ohjelmistot ja laitteisto linkittyy vahvasti toisiinsa vastavuoroisen vanhenemisen periaatteen (mutual obsolescence) kautta
- “Jos päätämme, että perustavanlaatuinen muutos tarvitaan ja se saattaa olla muutos, jota käyttäjät eivät halua, kuka päättää, minkä muutoksen pitäisi tapahtua ja miten?”
  - Innovaatio <-> hävitys
  - Uudistaminen vs. uudelleenkäyttö
  - Laatu + tasa-arvo
  - Omistajuus / identiteetti
  - Luonnollisten mallit + reflektio

Blevis: Sustainable Interaction Design: Invention & Disposal, Renewal & Reuse, CHI, 2007

1. **disposal**—does the design cause the disposal of physical material, directly or indirectly and even if the primary material of the design is digital material?
2. **salvage**—does the design enable the recovery of previously discarded physical material, directly or indirectly and even if the primary material of the design is digital material?
3. **recycling**—does the design make use of recycled physical materials or provide for the future recycling of physical materials, directly or indirectly and even if the primary material of the design is digital material?
4. **remanufacturing for reuse**—does the design provide for the renewal of physical material for reuse or updated use, directly or indirectly and even if the primary material of the design is digital material?
5. **reuse as is**—does the design provide for transfer of ownership, directly or indirectly and even if the primary material of the design is digital material?
6. **achieving longevity of use**—does the design allow for long term use of physical materials by a single owner without transfer of ownership, directly or indirectly and even if the primary material of the design is digital material?
7. **sharing for maximal use**—does the design allow for use of physical materials by many people as a construct of dynamic ownership, directly or indirectly and even if the primary material of the design is digital material?
8. **achieving heirloom status**—does the design create artifice of long-lived appeal that motivates preservation such that transfer of ownership preserves quality of experience, directly or indirectly and even if the primary material of the design is digital material? This notion of heirloom status is similar to Nelson & Stolterman’s [30] description of “ensoulment”.
9. **finding wholesome alternatives to use**—does the design eliminate the need for the use of physical resources, while still preserving or even ameliorating qualities of life in a manner that is sensitive to and scaffolds human motivations and desires?
10. **active repair of misuse**—is the design specifically targeted at repairing the harmful effects of unsustainable use, substituting sustainable use in its place?

Design – Suunnittelu vai Muotoilu?

- Suunnittelu terminä viittaa ajatteluun, johonkin, jota tullaan (mahdollisesti) tekemään
- Muotoilu on konkretiaa, muotoillaan jokin asia tietynlaiseksi, muokataan todellisuutta
- Design = suunnittelu + muotoilu?

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

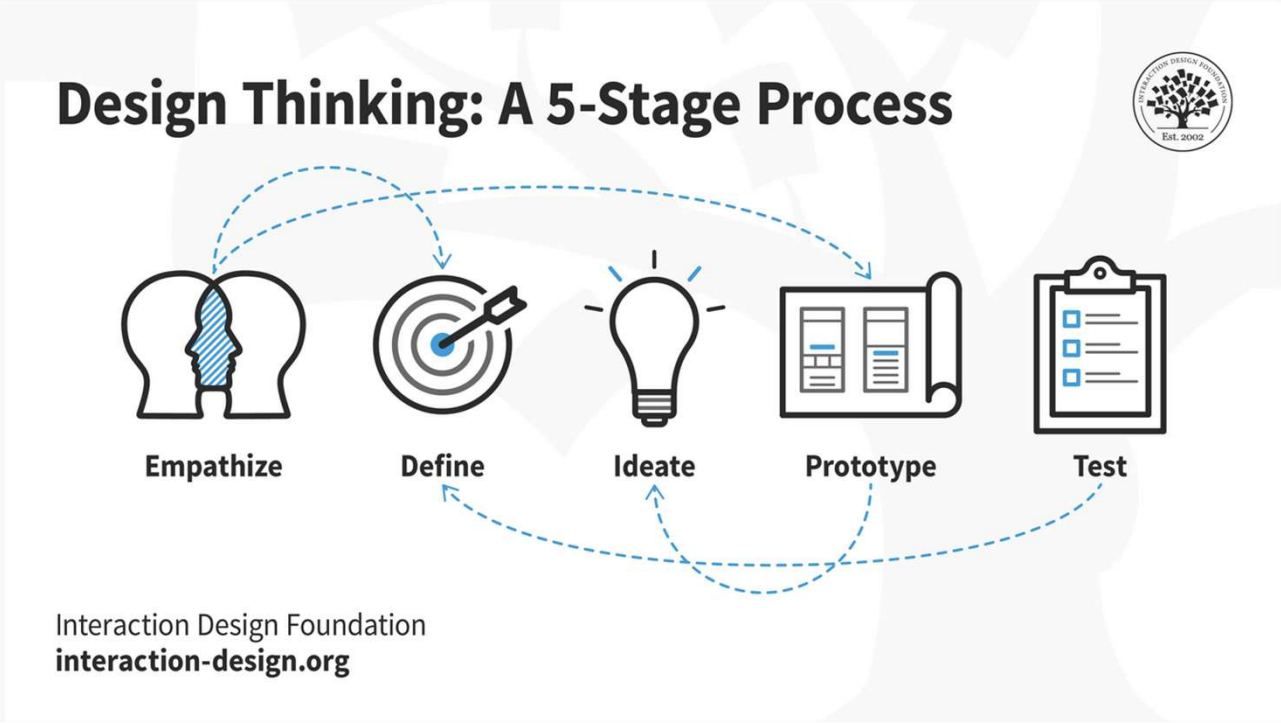
Sustainability design

SUUNNITTELU

MUOTOILU

Muotoiluajattelu

- ei-lineaarinen iteratiivinen prosessi
- käyttäjän ymmärrys
- oletusten haastaminen
- ratkaisujen löytäminen innovatiivisesti



Elinkaariajattelu ja muotoilu – tärkeimmät seikat

Käyttäjän osallistaminen – osallistetaan käyttäjiä suunnitteluvaiheessa, saadaan tietoa ongelmista, ajatuksia vaihtoehtoisista ratkaisuista

Käytettävyys – vahvistetaan, että suunnittelu on tehokasta ja toimivaa, helppo oppia ja muistaa, käytännöllinen, käyttäjän vaatimuksiin vastaamisen arviointi

Iterointi – testataan ja kehitetään kunnes täyttää käyttäjän vaatimukset

Todellinen vuorovaikutus – seurataan käyttäjien käyttäytymistä

T. Issa and P. Isaias, *Sustainable Design*, 2022

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

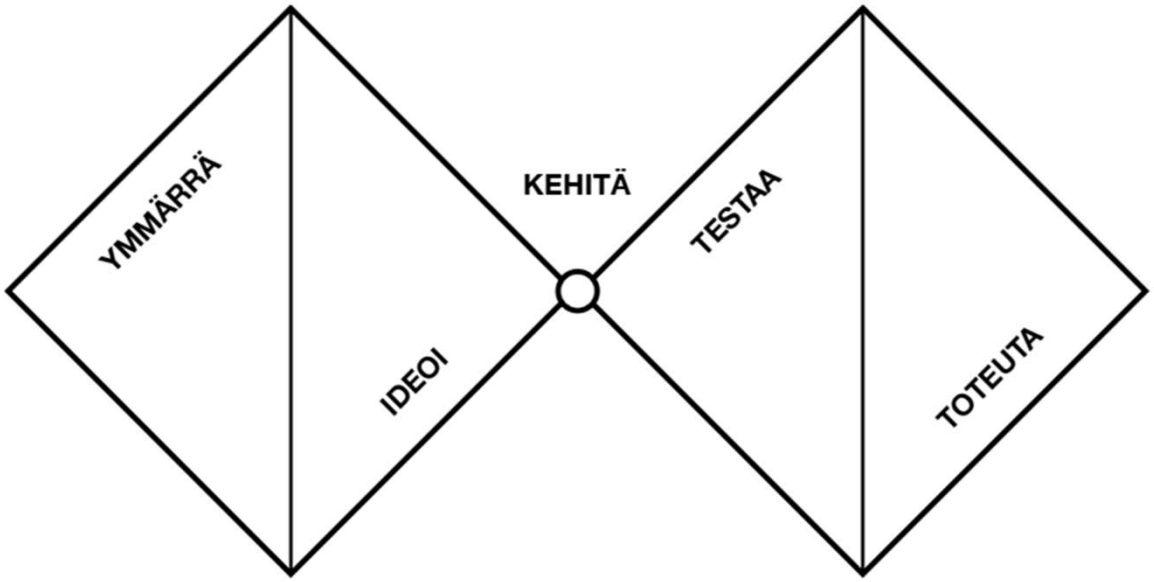
UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Green ICT -työkalun muotoilu

- osallistettiin 5 yritystä
- määriteltiin todellinen tarve – ymmärrys
- ideoimalla raamit työkalulle
- idean testaus yrityksillä
- iterointi
- prototyypin testaus yrityksillä
- iterointi
- julkaisu



“Suomessa tehdään koodia nopeasti, ei nopeaksi” Janne Kalliola, Exove

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

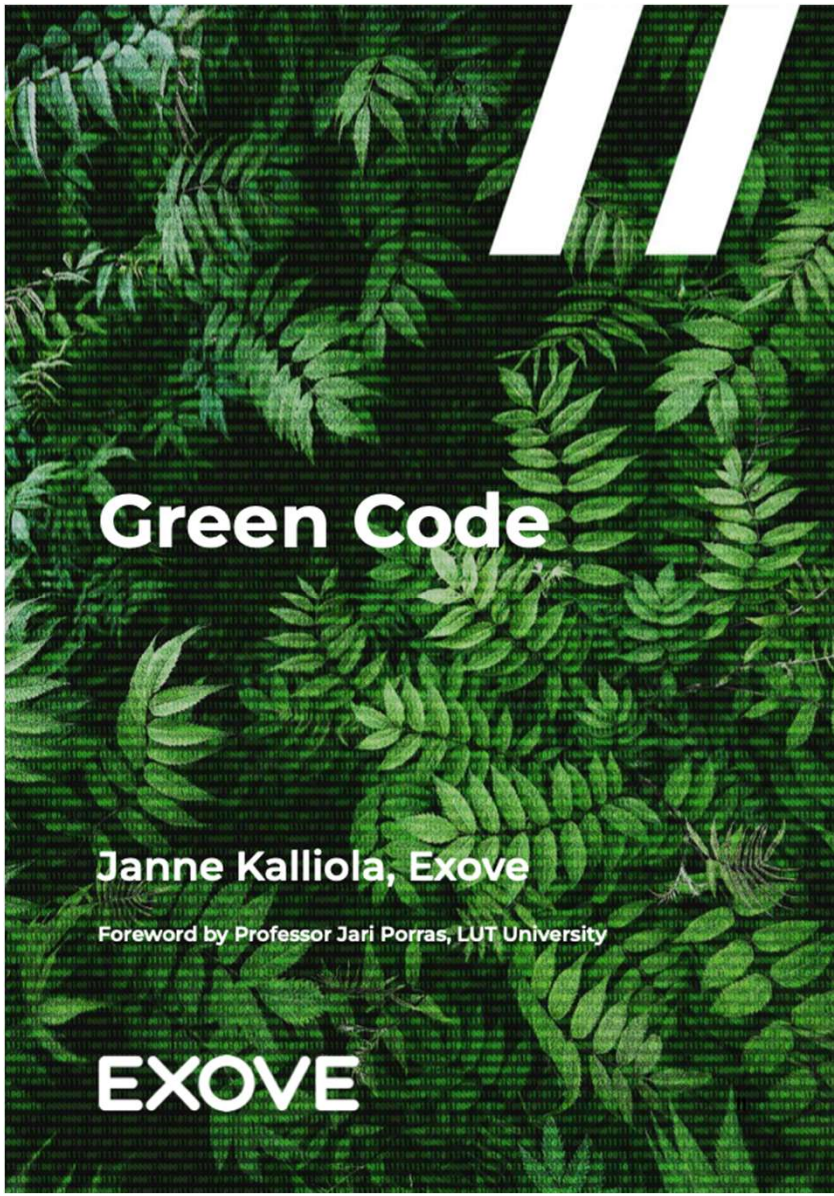
UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Vihreä koodi

- Janne Kalliolan, exove kirja “Green Code” hyvä yleiskatsaus
- Kadotetun tehokkuuden metsästys - Hukka
  - Tarpeettomat ohjelmistot
  - Väärä käyttö (tapa)
  - Käyttäjän virheet
  - Väärä arkkitehtuuri
  - Väärät datamallit
  - Ei optimoitu datan käyttö
  - Tarpeeton tiedonsiirto
  - Epäsopivat algoritmien / tehottomuus
  - Väärin ohjeistetut käyttäjät
  - Turhaa koodia
  - Tehoton koodaus
  - Ohjelmiston käynnistyksen kankeus
  - Turha toisto



Ratkaisuja vihreän koodin haasteisiin

- **Tallennetun datan minimointi**
  - Minimoi staattine data ja tiedostot, Käyttäjän syöttämän datan minimointi, Cold storage (nykypäivän nauha-asemat), Analyyseihin käytetyn datan minimointi / tuhoaminen
- **Siirretyn datan minimointi**
  - Siirtotaajuus, datan kompressointi, viestiformaatti/-protokolla, esitysdatan poisto, vain muuttuneen datan siirto, lähetettävän datan oikeellisuuden tarkistus, datan yhdistäminen siirtoa varten
- **Koodin minimointi**
  - Kuollut koodi, kirjastojen minimointi,sovelluksen piirteiden minimointi, natiivi koodi
- **Sovelluksen tehokkuuden maksimointi**
  - Algoritmit, datarakenteet, syöppöjen kohtien optimointi, refaktorointi, ajoympäristö, ohjelmointikieli, taustaprosessit
- **Ulkopuolisten resurssien käyttö**
  - CDN, välimuistit ja kuormantasauspalvelimet, ulkoiset tietoturvaratkaisut

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

Tämän päivän haasteita ja ratkaisuja

- AI
  - Opetusmateriaalin optimointi, AI:n opetus, AI:n käyttämä energia
  - Tarpeellisuus, Rajoittaminen, Mallin valinta, Mallin opetus ja konfigurointi
- Lohkoketjut ja kryptovaluutat
  - Konsensusalgoritmi, redundantti tieto, tiedonsiirto
- IoT
  - Sensorin taajuus, kommunikointikanavan valinta, tiedonsiirron taajuus,

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Ohjelmointikielten erot

- Vertailu on kuitenkin äärimmäisen monimutkainen tehtävä, koska kielen suorituskykyyn vaikuttaa
  - kääntäjän, virtuaalikoneen, roskienkerääjän, käytettävissä olevien kirjastojen jne. laatu.
- Ohjelmistoa voi nopeuttaa parantamalla sen lähdekoodia, mutta myös "vain" optimoimalla sen kirjastoja ja/tai käännöprosessia

Table 2: Languages sorted by paradigm

Paradigm	Languages
Functional	Erlang, F#, Haskell, Lisp, Ocaml, Perl, Racket, Ruby, Rust;
Imperative	Ada, C, C++, F#, Fortran, Go, Ocaml, Pascal, Rust;
Object-Oriented	Ada, C++, C#, Chapel, Dart , F#, Java, JavaScript, Ocaml, Perl, PHP, Python, Racket, Rust, Smalltalk, Swift, TypeScript;
Scripting	Dart, Hack, JavaScript, JRuby, Lua, Perl, PHP, Python, Ruby, TypeScript;

Energy Efficiency across Programming Languages - How Do Energy, Time, and Memory Relate?, SLE'17, 2017

Ohjelmointikielten vertailu

	Energy
(c) C	1.00
(c) Rust	1.03
(c) C++	1.34
(c) Ada	1.70
(v) Java	1.98
(c) Pascal	2.14
(c) Chapel	2.18
(v) Lisp	2.27
(c) Ocaml	2.40
(c) Fortran	2.52
(c) Swift	2.79
(c) Haskell	3.10
(v) C#	3.14
(c) Go	3.23
(i) Dart	3.83
(v) F#	4.13
(i) JavaScript	4.45
(v) Racket	7.91
(i) TypeScript	21.50
(i) Hack	24.02
(i) PHP	29.30
(v) Erlang	42.23
(i) Lua	45.98
(i) Jruby	46.54
(i) Ruby	69.91
(i) Python	75.88
(i) Perl	79.58

	Time
(c) C	1.00
(c) Rust	1.04
(c) C++	1.56
(c) Ada	1.85
(v) Java	1.89
(c) Chapel	2.14
(c) Go	2.83
(c) Pascal	3.02
(c) Ocaml	3.09
(v) C#	3.14
(v) Lisp	3.40
(c) Haskell	3.55
(c) Swift	4.20
(c) Fortran	4.20
(v) F#	6.30
(i) JavaScript	6.52
(i) Dart	6.67
(v) Racket	11.27
(i) Hack	26.99
(i) PHP	27.64
(v) Erlang	36.71
(i) Jruby	43.44
(i) TypeScript	46.20
(i) Ruby	59.34
(i) Perl	65.79
(i) Python	71.90
(i) Lua	82.91

	Mb
(c) Pascal	1.00
(c) Go	1.05
(c) C	1.17
(c) Fortran	1.24
(c) C++	1.34
(c) Ada	1.47
(c) Rust	1.54
(v) Lisp	1.92
(c) Haskell	2.45
(i) PHP	2.57
(c) Swift	2.71
(i) Python	2.80
(c) Ocaml	2.82
(v) C#	2.85
(i) Hack	3.34
(v) Racket	3.52
(i) Ruby	3.97
(c) Chapel	4.00
(v) F#	4.25
(i) JavaScript	4.59
(i) TypeScript	4.69
(v) Java	6.01
(i) Perl	6.62
(i) Lua	6.72
(v) Erlang	7.20
(i) Dart	8.64
(i) Jruby	19.84

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

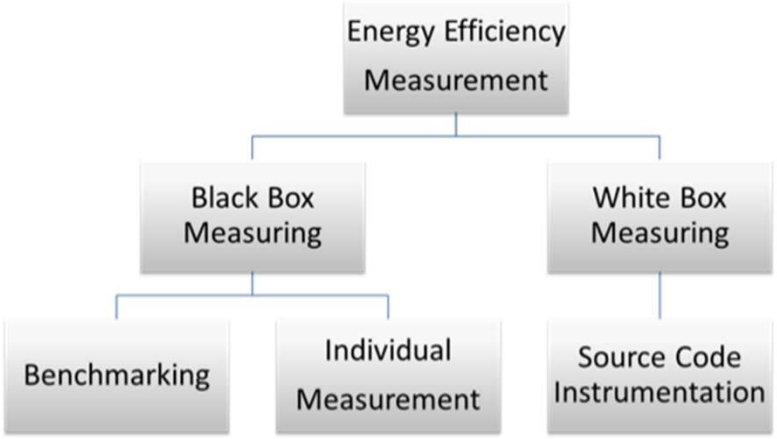
TURKU AMK

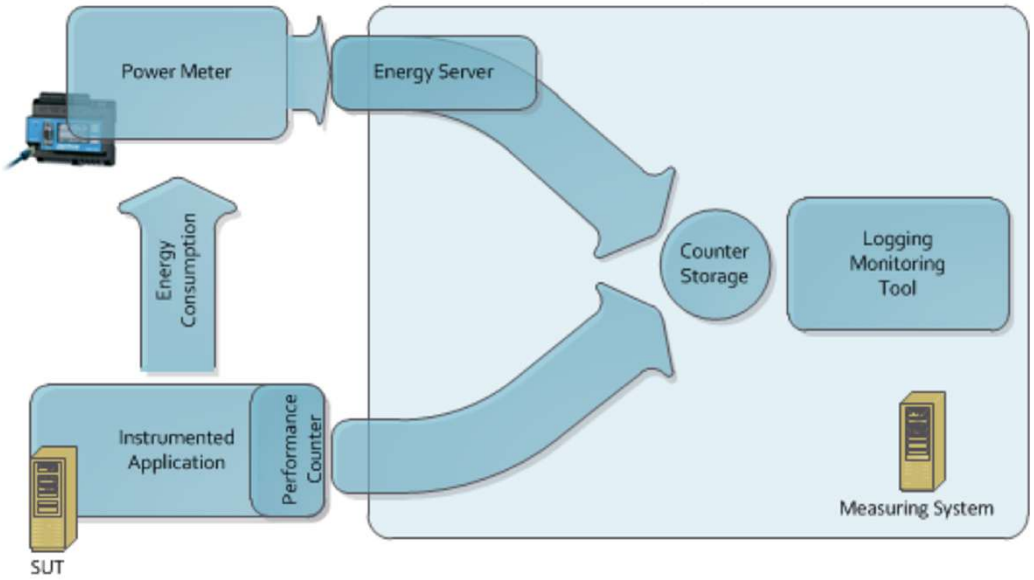
OPETUSHALLITUS RAHOITTAA HANKETTA

Ohjelmistojen mittaamisen vaikeus

- Yleinen ohjelmistojen mittaamiseen tarkoitettu mittari ja menetelmä

$$EnergyEfficiency = \frac{UsefulWorkDone}{UsedEnergy}.$$

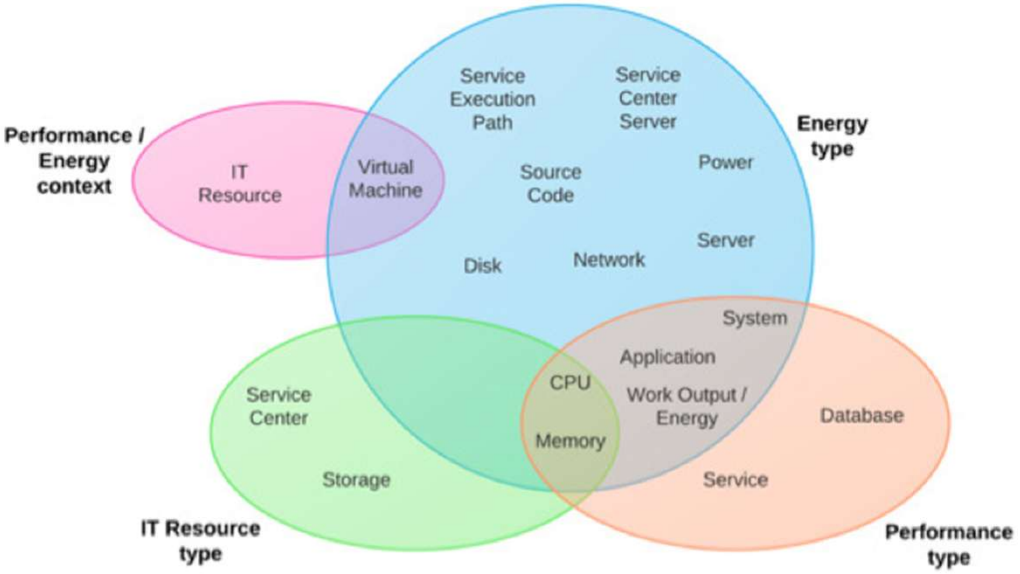




Johann et al.: How to Measure Energy-Efficiency of Software: Metrics and Measurement Results, GREENS, 2012

Millaisia mittareita ohjelmistoille löytyy?

Metrics Type	Total	Measurement Unit(s)
Energy	48	Joule (J), Index, Watt (W), Ampere (A) Kilowatt-hour (kWh), Number, byte/kWh
Performance	19	GFLOPS/kWh, Computing Unit/kWh, Percentage (%), Seconds (s), Index, Number
Utilization	17	Percentage (%), Megabyte (MB), Megahertz (MHz), GB/s
Economic	9	Dollars ($)
Performance / Energy	2	GFLOPS/Watt, Index
Pollution	1	CO_2 units

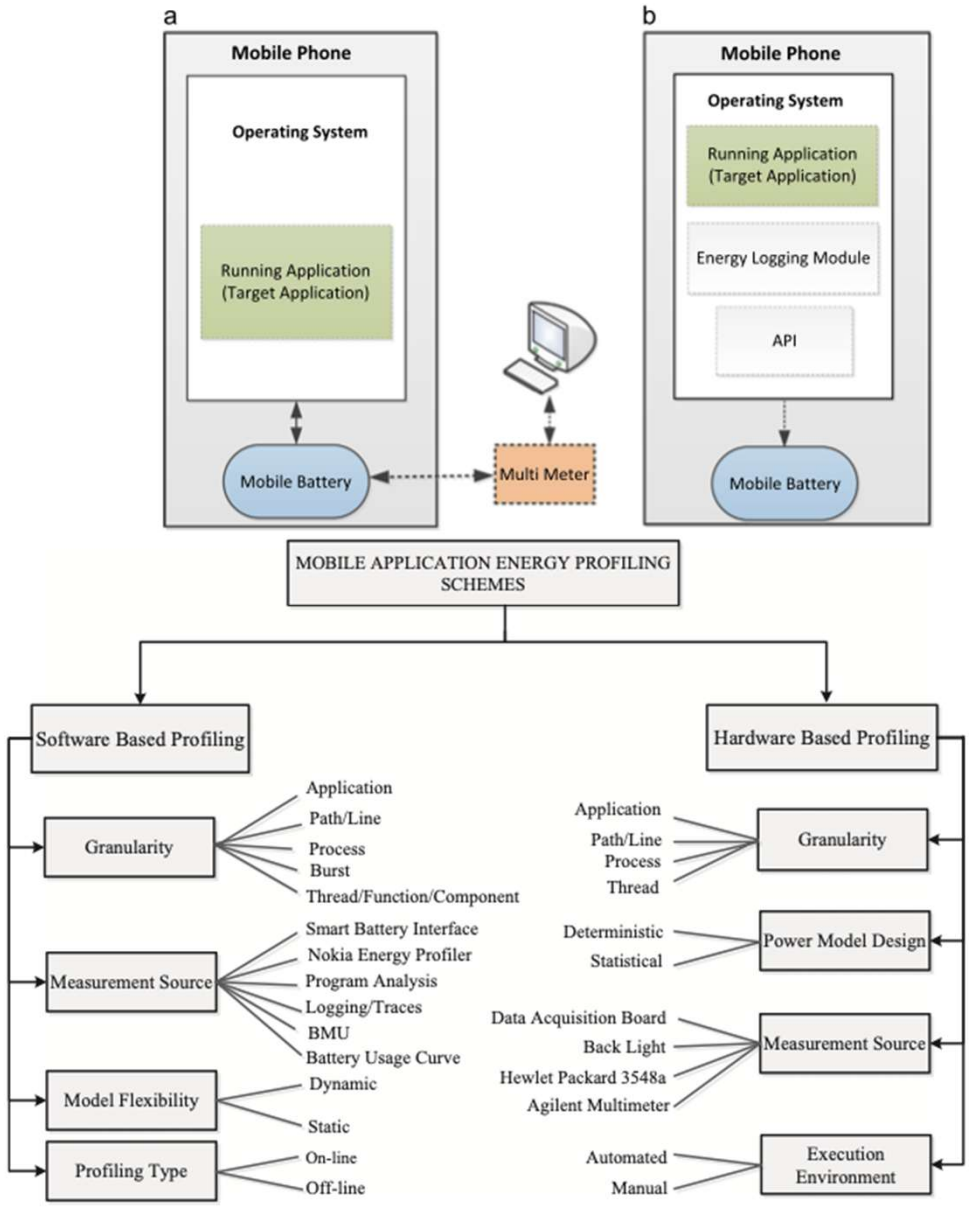


Bozzelli et al.: A SLR on green software metrics, 2014

Ohjelmistojen mittaaminen

- Laitteistopohjainen vs. ohjelmistopohjainen
- Laitteistopohjainen tarkka määritetyssä ympäristössä
- Ohjelmistopohjainen perustuu esim. akun kulutukseen, ja tarkkuus riippuu tasosta jolla akun kulutusta voidaan mitata

Ahmad et al.: A Review on mobile application energy profiling: Taxonomy, state-of-the-art, and open research issues, JNCA, 2015



Ohjelmistojen vaikutusten mittaaminen

- Mittaaminen haastavaa
  - Koska softa on täysin riippuvaista raudasta:
  - Erilaiset hostaus-palvelut, infra ja päätelaitteet
  - Eri ohjelmointikielet ja ympäristöt (IDE)
  - Eri käyttöjärjestelmät
  - Suora ympäristömittaus kohtuullisen mahdotonta
- MitViDi-hanke tehtiin ratkaisemaan haasteita

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY
LUT University
UNIVERSITY OF EASTERN FINLAND
TURKU AMK
OPETUSHALLITUS RAHOITTAA HANKETTA

MitViDi-hankkeen lähestymistapa

- Mittarit hankkijoita palvelemassa
  - Yksinkertaisia ja helppokäyttöisiä
  - Soveltuvia julkisiin hankintoihin
  - Ilmasto- ja ympäristöohjelmaa tukevia
  - Mahdollisia tuottajille toteuttaa
  - Muunnettavissa itsekäytettäväksi työkaluksi
- Ratkaisu: mittarikortit
- Kolme tasoa: Perus, edistynyt, huippu

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY
LUT University
UNIVERSITY OF EASTERN FINLAND
TURKU AMK
OPETUSHALLITUS RAHOITTAA HANKETTA

MitViDi-hankkeen lähestymistapa

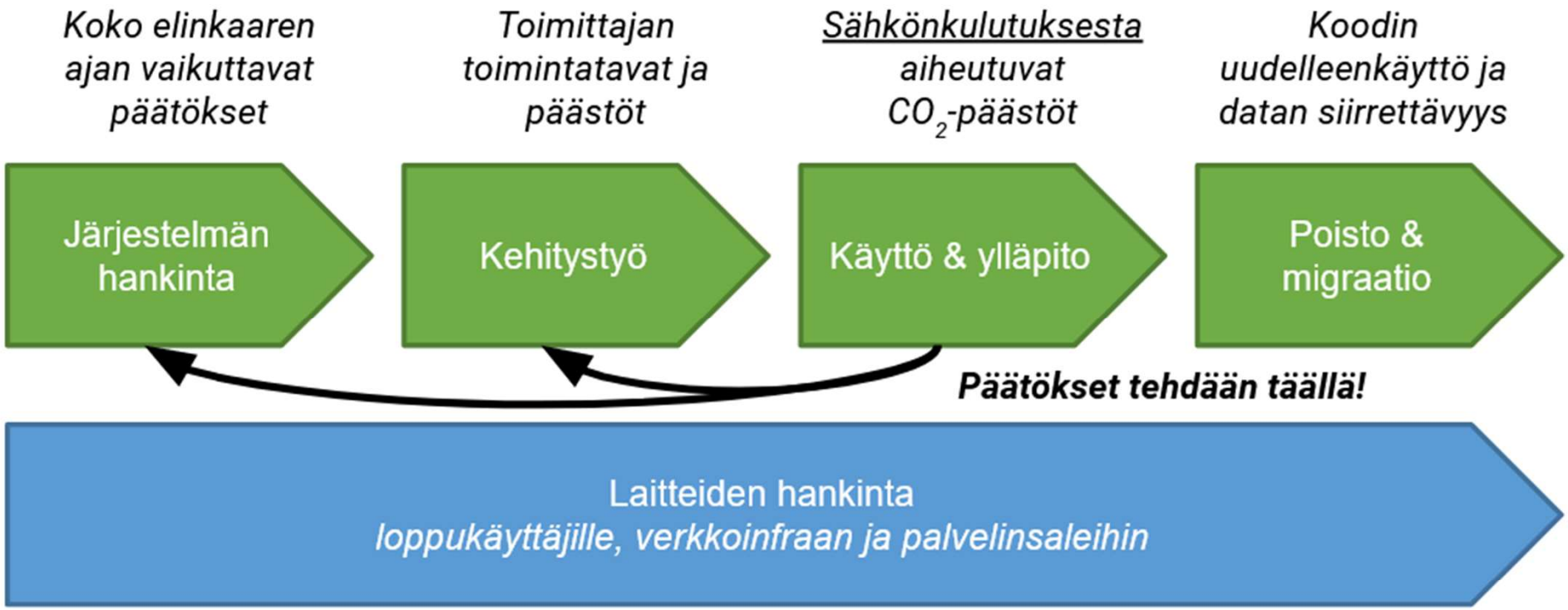
Perustaso
- Vihreyttä edistävät toiminnalliset ominaisuudet
- (Esim. virranhallinnan tuki, offline-toiminnallisuus)

Edistynyt taso
- Vihreyttä tukevat laadulliset / arkkitehtuurin ominaisuudet
- (Esim. toimivat rajapinnat, taaksepäin yhteensopivuus)
- Kehittäjän toiminnan ympäristöystävällisyys
- Ohjelmistokehittäjien kestävän koodauksen osaaminen
- Kehittäjäorganisaation toiminnan kestävyys

Huipputaso
- Hiilijalanjäljen optimointi kokonaisvirrankulutusta vähentämällä
- Front-end: Mallinnus skenaarioiden virrankulutusta mittaamalla
- Back-end: Mittaus tai mallinnus (CPU/GPU, muisti, i/o, verkko)
- Verkkoliikenne: Mallinnus palvelimen verkkoliikenteen perusteella

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY
LUT University
UNIVERSITY OF EASTERN FINLAND
TURKU AMK
OPETUSHALLITUS RAHOITTAA HANKETTA

MitViDi
Mittarit vihreän digitalisaation julkisiin ICT-hankintoihin



Vipuvoimaa EU:lta 2014–2020

Esimerkki: Ohjelmisto tukee järjestelmän virranhallintaa

Mittarin lyhyt kuvaus | Ohjelmisto ei saisi estää järjestelmää säästämästä virtaa lepotilassa tai muuten tyhjäkäynnillä. Virranhallinnan ei myöskään pitäisi haitata ohjelmiston toiminnallisuutta, eli käytettävyys ei häiriinny tai dataa menetetä.

Ohjeistus kriteerin käytöstä hankinnan aikana | Kriteerin tulisi olla minimivaatimus.

Arvio kriteerin toteutuksen ympäristövaikutuksista | Virranhallinnan tukeminen mahdollistaa mittavat säästöt virrankulutuksen osalta. 2000-luvun alussa tehtyjen arvioiden mukaan tyypillisen toimistokoneen virrankulutus on noin 60 wattia normaalissa käytössä ja noin 5 wattia lepotilassa. Laajamittainen virranhallintatuki säästäisi parhaimmillaan 36 TWh vuodessa ja 2,7 miljardia dollaria.

OPETUSHALLITUS RAHOITTAA HANKETTA

GreenICTComp

4. Manifestit/julistukset liittyen kestäviin ohjelmistoihin

OPETUSHALLITUS RAHOITTAA HANKETTA

LUT University
UNIVERSITY OF EASTERN FINLAND
TURKU AMK
TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

KARLSKRONA MANIFESTO
FOR SUSTAINABILITY DESIGN

Becker at al.: Requirements: The key to sustainability, IEEE Software, 2016

SUSTAINABILITY PRINCIPLES FOR SOFTWARE ENGINEERING

The following principles are based on "Sustainability Design and Software: The Karlskrona Manifesto."[1]

- Sustainability is systemic; a system can never be treated in isolation from its environment.
- Sustainability is multidimensional; the five key dimensions are economic, social, environmental, technical, and individual.
- Sustainability is interdisciplinary; sustainability design in software engineering requires an appreciation of concepts from other disciplines and must work across disciplines.
- Sustainability transcends the software's purpose; any software can impact the sustainability of its socioeconomic, sociotechnical, cultural, and natural environments.
- Sustainability is multilevel; it requires us to consider at least two spheres during system design: the system under design and its sustainability, and the wider system of which it will be part.
- Sustainability is multi-opportunity; it requires us to seek interventions that have the most leverage on a system[2] and to consider the opportunity costs.
- Sustainability involves multiple timescales; it requires long-term thinking to address the timescales on which sustainability effects occur.
- Sustainability isn't zero-sum; changing a system's design to consider the long-term effects doesn't automatically imply making sacrifices now.
- System visibility is a necessary precondition and enabler for sustainability design. This is because only a transparent status of the system and its context, made visible at different abstraction levels and perspectives, can enable system designers to make informed responsible choices.

For more on this, see www.sustainabilitydesign.org.

Sustainability design manifesto

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

Becker at al.: Requirements: The key to sustainability, IEEE Software, 2016

Task	Standard current practice	Focus of future practice
Mind-setting	The world is a puzzle, and we should solve the problem.	The world is complex, and we should first understand the dilemmas.
Determination of the project objective and the system purpose, boundary, and scope	Focus on the immediate business need and key system features. Don't question the project's or system's purpose.	Emphasize how the project can affect sustainability in all dimensions. Strive to advance sustainability in multiple dimensions simultaneously. Experiment with different system boundaries to understand the alternative impacts.
External constraint identification	See constraints as imposed by the direct environment of the system and its technical interfaces. Minimize the constraints considered, but include legal, safety, security, technical, and business resources.	See constraints in each dimension as opportunities. Look for constraints from additional sources, starting with company corporate-social-responsibility policies, legislation, and sustainability standards.
Stakeholder identification	Minimize the number of stakeholders involved, and focus on those who have influence. Focus on internal stakeholders, and exclude unreachable stakeholders.	Maximize stakeholder involvement in an inclusive perspective integrating external stakeholders, and involve those who are affected. Assign a dedicated role to be responsible for sustainability, and introduce surrogate stakeholders to represent outside interests.
Success criteria definition	Focus on the financial bottom line at project completion. Measure the business outcome and financial return on investment.	Focus on advancing multiple dimensions simultaneously, including financial aspects, and take into account that most effects occur after project completion.
Requirements elicitation	Focus on the features and immediate effects the stakeholders want.	Help the stakeholders understand the system's enabling effects. Use creativity techniques and long-term scenarios to forecast the potential structural impact.

Risk identification	Identify risks that threaten timely project completion within the budget.	Include the effects on the system's wider environment. Include enabling and structural effects and risks that can develop over time.
Tradeoff analysis	View tradeoff analysis as a prioritization and selection problem, and let the key stakeholders decide.	Strive to transform sustainability tradeoffs into mutually beneficial situations. Ensure that a wider range of stakeholders (or their surrogates) discuss sustainability tradeoffs.
Go/no-go decision	Base the decision on feasibility, financial costs and benefits, and risk exposure to project participants—that is, internal stakeholders.	This continues to be an internal business decision but is documented to show to external audiences that it took into account sustainability indicators and enabling effects. The decision is based on a consideration of positive and negative effects in all five dimensions.
Requirements validation	Let key stakeholders verify that their interests are captured.	Ensure broad community involvement focused on understanding effects.
Project completion	Verify whether success criteria are met on the completion date. After that, focus on maintenance and evolution.	Evaluate the effects in all five dimensions over a certain time frame after completion, aligned with the expected timescale of effects.
Requirements documentation	Current templates ignore long-term effects and sustainability considerations.	Templates require information about sustainability as a design concern and support analysts with checklists.

Kestävä agile manifesti

1. Ihmiset ja planeetta ennen liikevoittoa

- Asetamme paikkojen ja ihmisten hyvinvoinnin etusijalle lyhyen aikavälin taloudellisten hyötyjen sijaan, mikä edistää uudistuvaa ja oikeudenmukaista maailmaa.

2. Adaptiivisuus ennen Jäykkyyttä

- Suhtaudumme muutokseen mahdollisuutena kasvuun, resilienssiin ja kestävyyteen ja arvostamme sopeutumiskykyä jäykkien suunnitelmien ja rakenteiden sijaan.

3. Runsaus ennen niukkuutta

- Ymmärrämme, että globaalit haasteet vaativat yhteistyösuhteiden ja yhteistyön kehittämistä

4. Arvo ennen kulutusta

- Pyrimme vähentämään energian, materiaalien ja liikenteen kulutusta kaikessa toiminnassamme

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Kestävä agile vs. agile manifestin periaatteet

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

1. Sustainable Practices
2. Embrace and Respond to Global Challenges
3. Deliver Sustainable Value Frequently
4. Collaborate with Diverse Stakeholders
5. Empower Agile Sustainability Leaders
6. Support Academic Adoption of Agile Sustainability
7. Innovate for the Well-being of Society and the Environment
8. Evolve Agile for Modern Sustainability Needs

https://www.agilealliance.org/sustainability-manifesto/

1. **Satisfy Customers Through Early and Continuous Delivery**
2. **Welcome Changing Requirements Even Late in the Project**
3. **Deliver Value Frequently**
4. **Break the Silos of Your Project**
5. **Build Projects Around Motivated Individuals**
6. **The Most Effective Way of Communication is Face-to-face**
7. **Working Software is the Primary Measure of Progress**
8. **Maintain a Sustainable Working Pace**
9. **Continuous Excellence Enhances Agility**
10. **Simplicity is Essential**
11. **Self-organizing Teams Generate Most Value**
12. **Regularly Reflect and Adjust Your Way of Work to Boost Effectiveness**

https://businessmap.io/agile/project-management/principles

Agile manifesto kestävyysnäkökulmasta (LUT opiskelijat)

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software **in which the principles of sustainability are laid down from the first stage**
2. Welcome **profitable and reasonable** changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
3. Deliver working software frequently **depending on the completion stage and frequency set by the customer and development team.**
4. Business people and developers must work together daily throughout the project **while addressing all three dimensions of sustainability.**
5. Build projects around motivated individuals **who know how to achieve the principles of sustainability in the final product.** Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team **is virtual meeting**.
7. Working software **and its impact on the economy, environment, and society** are the primary measures of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to make the most sustainable technical and design choices will enhance **both sustainability and** agility.
10. **Develop just enough to get the job done.** (simplicity)
11. The best architectures, requirements, and designs emerge from self-organizing teams **with a sustainable mindset**.
12. At regular intervals, the team reflects on **how to improve the sustainability aspect of the project**, then tunes and adjusts its behavior accordingly.

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Leverage points

“Vipupisteet ovat paikkoja monimutkaisen järjestelmän sisällä (yritys, talous, keho, kaupunki, ekosysteemi), joissa pieni muutos yhdessä asiassa voi saada aikaan suuria muutoksia kaikessa”
Meadows D.H., 1999

Penzenstadler B. et al.: Software Engineering for Sustainability - Find the Leverage Points!, IEEE Software, 2018

Table A. Leverage points.[3]

Leverage point	Description
LP 12	Constants, parameters, and numbers. Tweaking parameters allows change to the intensity of the flows in systems but rarely alters the underlying dynamics.
LP 11	The sizes of buffers and other stabilizing stocks, relative to their flows. Stabilize a system by adjusting the capacity of its buffers, and make it more efficient by optimizing the flow.
LP 10	The structure of material stocks and flows (such as transportation networks and population age structures). Physical structure is crucial in a system but often hard to change; therefore, the leverage point is in proper initial design.
LP 9	The lengths of delays, relative to the rate of system change. A system cannot respond to short-term changes when it has long-term delays.
LP 8	The strength of balancing feedback loops, relative to the impacts they respond to. Balancing feedback loops help systems to self-correct by monitoring and adjusting according to the system goal.
LP 7	The gain around reinforcing feedback loops. Reinforcing feedback loops can be sources of system instability or mechanisms to amplify desired change, so adjusting their strength affects how the system responds to change.
LP 6	The structure of information flows. This can create a new feedback loop that was not there before. Altering the structure of information flows enables more agency by users.
LP 5	The rules of the system, including incentives, punishments, and constraints. Social rules include constitutions, laws, standards, policies, and incentives. Changing the rules of a system can change the behavior of the society under them.
LP 4	The power to add, change, evolve, or self-organize system structure. In biology, this is called evolution; in society, we call it empowerment. In systems terms, it is called self-organization, the strongest form of system resilience.
LP 3	The goals of the system. Changing the goal of a system is a powerful strategy to effect change but can be hard to achieve.
LP 2	The mind-set or paradigm out of which the system arises. Paradigms are a shared set of deep beliefs about how the world works. They are the hardest to change in a system, as society will fiercely resist any challenges to its paradigms.
LP 1	The power to transcend paradigms. This final and most effective LP is about being unattached to existing paradigms; there is no certainty in any particular worldview.

Keskustelu: Kuinka hyvin manifestot tukevat ohjelmistojen kestävyyttä

TIEKE
TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA

Linkkejä

- https://www.imda.gov.sg/-/media/imda/files/infocomm-media-landscape/sg-digital/microsoft-imda-digital-sustainability-guideline.pdf
- https://greensoftware.foundation/articles/10-recommendations-for-green-software-development
- https://engineering.leanix.net/blog/sustainable-green-software-engineering/
- https://www.tcs.com/what-we-do/research/white-paper/greening-software-net-zero-emissions-sustainability
- Miro: työkalu - https://miro.com/app/board/uXjVP5KUB9Q=/

TIEKE TIETOYHTEISKUNNAN KEHITTÄMISKESKUS RY

LUT University

UNIVERSITY OF EASTERN FINLAND

TURKU AMK

OPETUSHALLITUS RAHOITTAA HANKETTA