Hasso Plattner Institut
Digital Engineering · Universität Potsdam

ERGO
A Munich Re company

Sustainability in Software Engineering – a point of view and call for action

Bernhard F. Kraft

HPI Potsdam

March 31, 2022

Public

Sustainability – a definition

ERGO
A Munich Re company

“meeting the needs of the present without compromising the ability of future generations to meet their own needs.”

(United Nations World Commission on Environment and Development (“Brundtland Commission“), 1987)

The 17 UN sustainable development goals

ERGO
A Munich Re company

Software engineering – Focus SDG‘s – a suggestion

ERGO
A Munich Re company

4 - Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all

5 - Achieve gender equality and empower all women and girls

HR related focus areas

12 - Ensure sustainable consumption and production patterns

13 - Take urgent action to combat climate change and its impacts

Usage-based focus areas

Expectation Management

ERGO
A Munich Re company

Software will – for the forseeable future – require physical hardware and energy to run.

12 - Ensure sustainable consumption and production patterns

13 - Take urgent action to combat climate change and its impacts

The goals we can commit to are:

1. **minimize power consumption**
2. **increase use of renewable power sources**
3. **reduce physical hardware footprint**

When it comes to energy efficiency, there are opportunities and threats

ERGO
A Munich Re company

```
    private func checkForCheat () -> Bool{
        var cheatFound = false
        cheatSet.removeAll(keepingCapacity: true)
        var checkSet:[SetCard]
        if game.dealtCards.count >= GameConstants.dealSize {
            for i in 0..<game.dealtCards.count-2{
                for j in i+1..<game.dealtCards.count-1{
                    for k in j+1..<game.dealtCards.count{
                        checkSet = [game.dealtCards[i], game.dealtCards[j], game.dealtCards[k]]
                        if !cheatFound && game.match(keysToMatch: checkSet)  {
                            cheatButton.isEnabled = true
                            cheatFound = true
                            cheatSet = checkSet
                        }
                    }
                }
            }
        }
        return cheatFound
    }

    private func addselectedCardToMatchingSet(_ sender:CardView){
//        print ("\(sender)")
        if let _ = cardView.setCardViews.firstIndex(of: sender) {
            let id = game.dealtCards[cardView.setCardViews.firstIndex(of: sender)!].id
            print("added card \(String(describing: id))")
            selectedCards[id] = sender
            sender.selected = true
        }
    }

    private func processMatch(matchingCards:[SetCard]){

        matchSet = Array(selectedCards.values)

        for matches in selectedCards{
            let card = game.dealtCards.filter {$0.id == matches.key}
            game.dealtCards.remove(at: game.dealtCards.lastIndex(of: card.first!)!)
            game.matchedCards.append(card.first!)
            matches.value.selected = false
            if cardView.setCardViews.firstIndex(of: matches.value) != nil {
                cardView.setCardViews.remove(at: cardView.setCardViews.firstIndex(of: matches.value)!)
```

Opportunities

- (Energy-) Optimized and modular code, reduced redundancies
- **AI models**: where is the tradeoff between “good enough” for use case and excessively trained?
- **Cloud usage**: resist the convenience to “push the button” more than needed (user training!)
- Leverage a **serverless architecture** (AWS lambda, Azure Functions) where possible
- Measure **actual processing needs** through dynamic code analysis

Threats

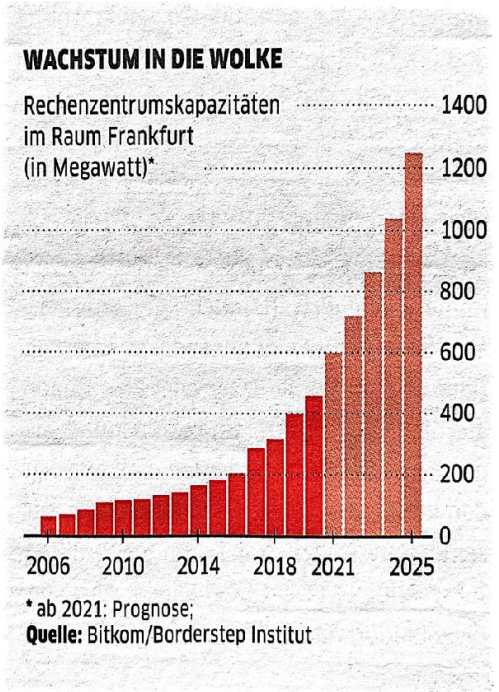
- User experience - **rising customer expectations** driven e.g., through metaverse, VR, 3D
- **blockchain and crypto currencies** –benefits, but very high processing requirements
- **AI** - ubiquitous and excessively trained

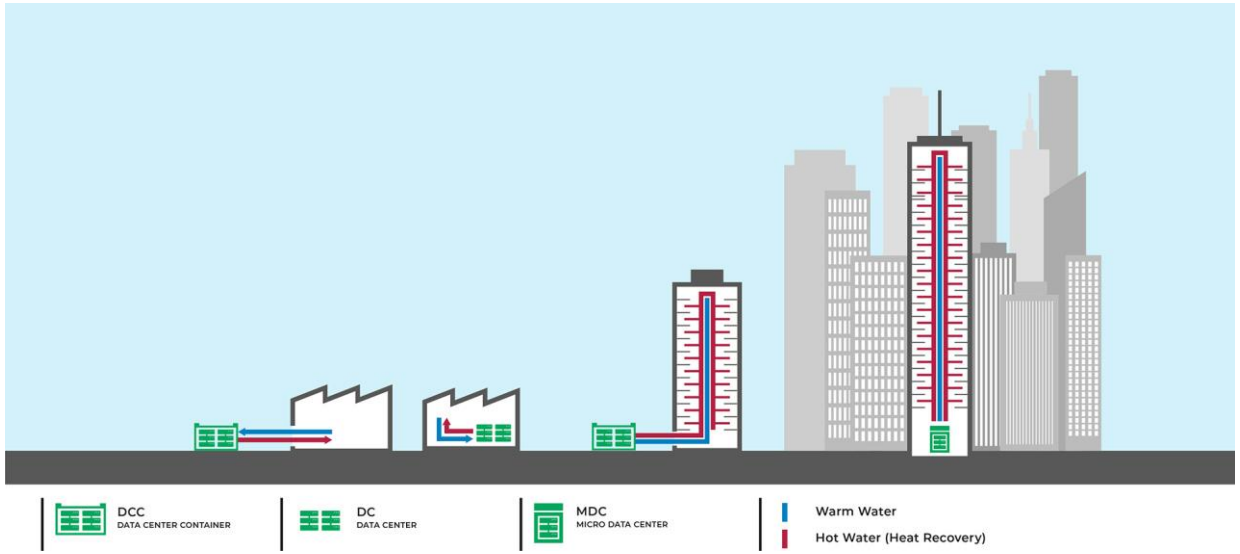
Datacenter consolidation, proper location and the cloud move can help minimize physical footprint

ERGO
A Munich Re company

Physical Footprint

- **No excess hardware** (consider capacity peaks, test- and development instances, backup and standby, …)
- **Facility footprint - minimized** (consider building, heating, cooling, staff needs, security, transportation (staff getting to and from), …)
- **Facility location** – consider cooling needs. Or use as local heating utility in cities
- **Use beyond financial depreciation** – use as long as it is useful, then find an aftermarket
- **Carbon-intensity of energy used** (e.g. sustainable source vs. carbon-based)





© Cloud&Heat Technologies GmbH, Dresden

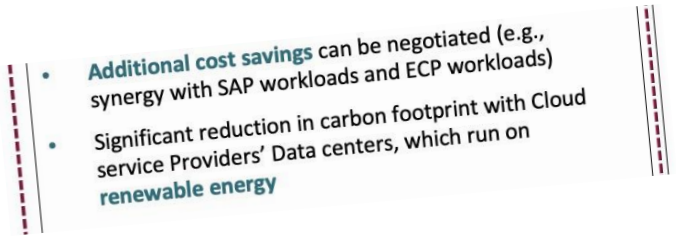
public

What you should be doing now…

ERGO
A Munich Re company

Software & Hardware Selection

- Include **sustainability criteria** and begin to weigh them such that they become relevant (vendor sustainability, carbon footprint of solution alternatives, etc.) – this also begins to change the thinking in your organization



Software Programs

- Prefer **deployment models that reduce the carbon footprint** (e.g. on demand cloud usage, serverless computing)

Data Center Reduction & Cloud Move

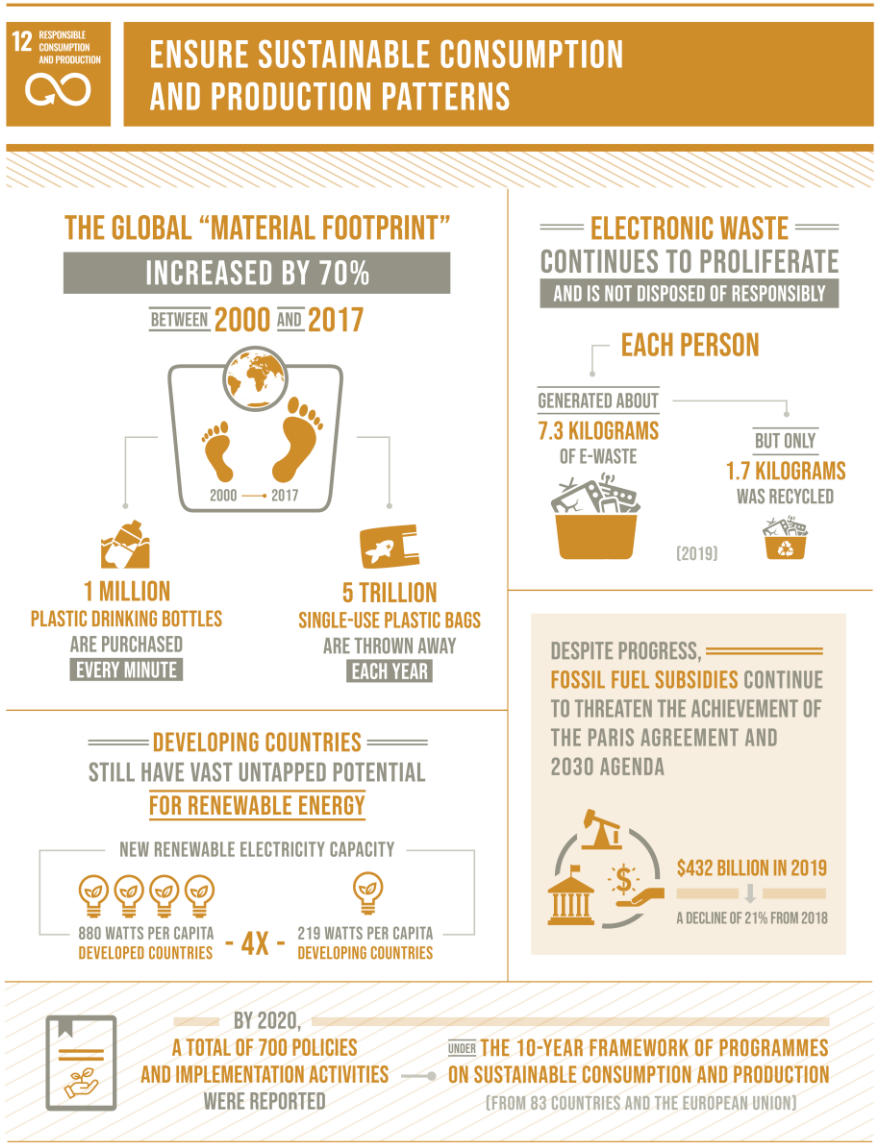
- **Look at your processing needs** holistically – future size, location, energy supply
- **Move to cloud aggressively** – but understand how your cloud service provider supports the sustainability goals (see previous page)

ERGO

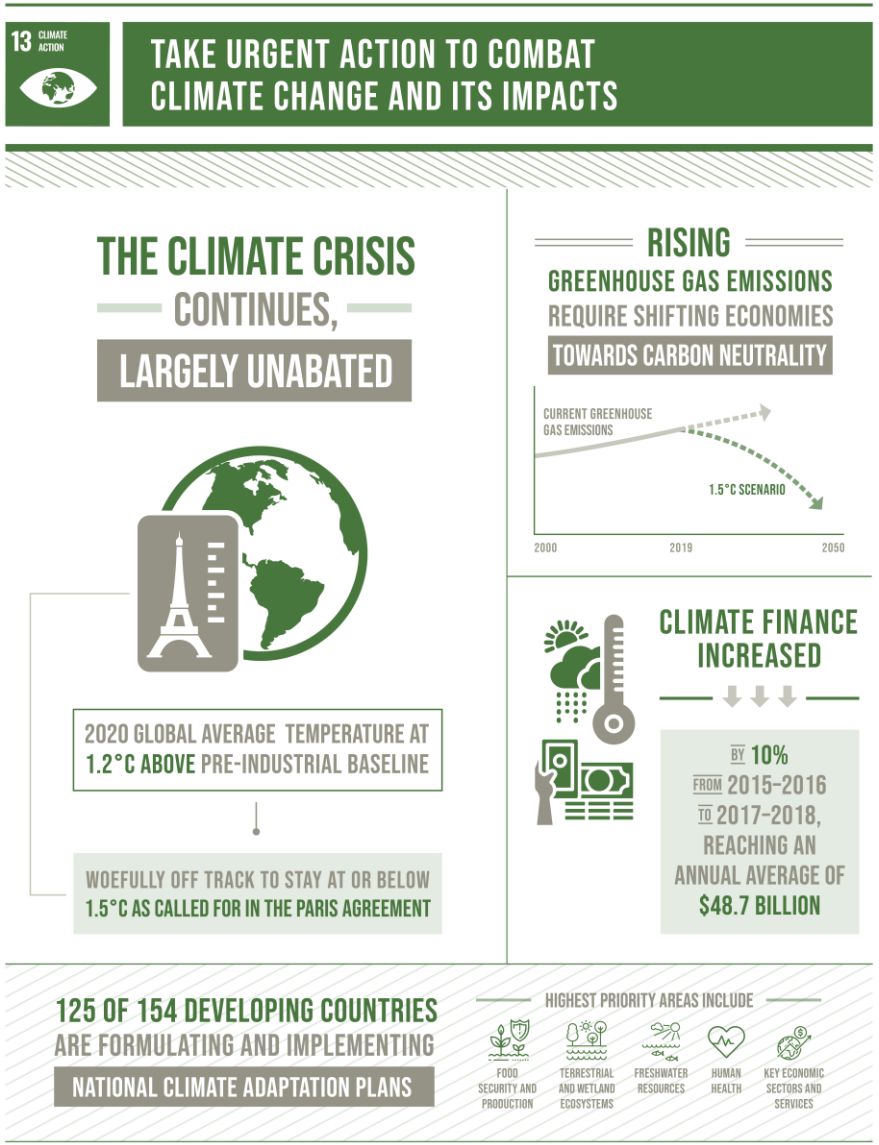
A Munich Re company

Backup

ERGO
A Munich Re company



THE SUSTAINABLE DEVELOPMENT GOALS REPORT 2021: UNSTATS.UN.ORG/SDGS/REPORT/2021/



THE SUSTAINABLE DEVELOPMENT GOALS REPORT 2021: UNSTATS.UN.ORG/SDGS/REPORT/2021/