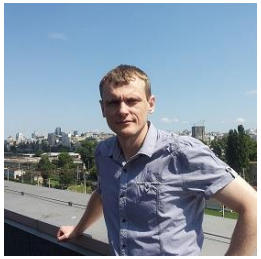
Software Engineering

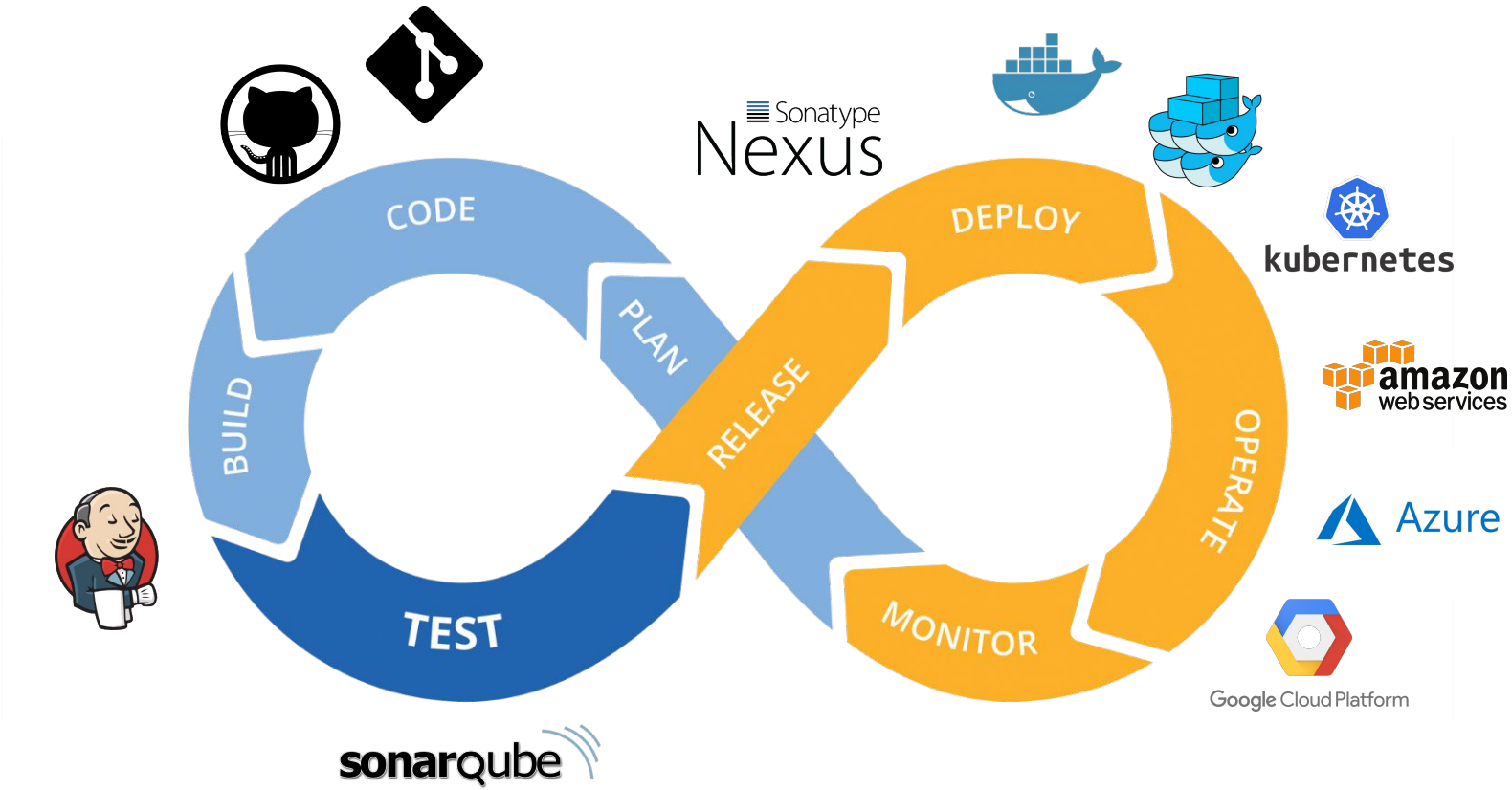
Hello!

My name is **Vladyslav Kurmaz**

You can find me at:
vladislav.kurmaz@gmail.com
vladyslav.kurmaz@globallogic.com
Linkein, Github, Facebook, G+



Development pipeline



Software & Hardware development

- Common parts
  - Math, Finite State machine
  - Tools
  - Languages
  - OS
- Differences
  - Output: Executables(files) vs Executables+Physical stuff
  - Hardware development is more difficult from debug and testing perspective
  - Software is more easy to replicate

Roles

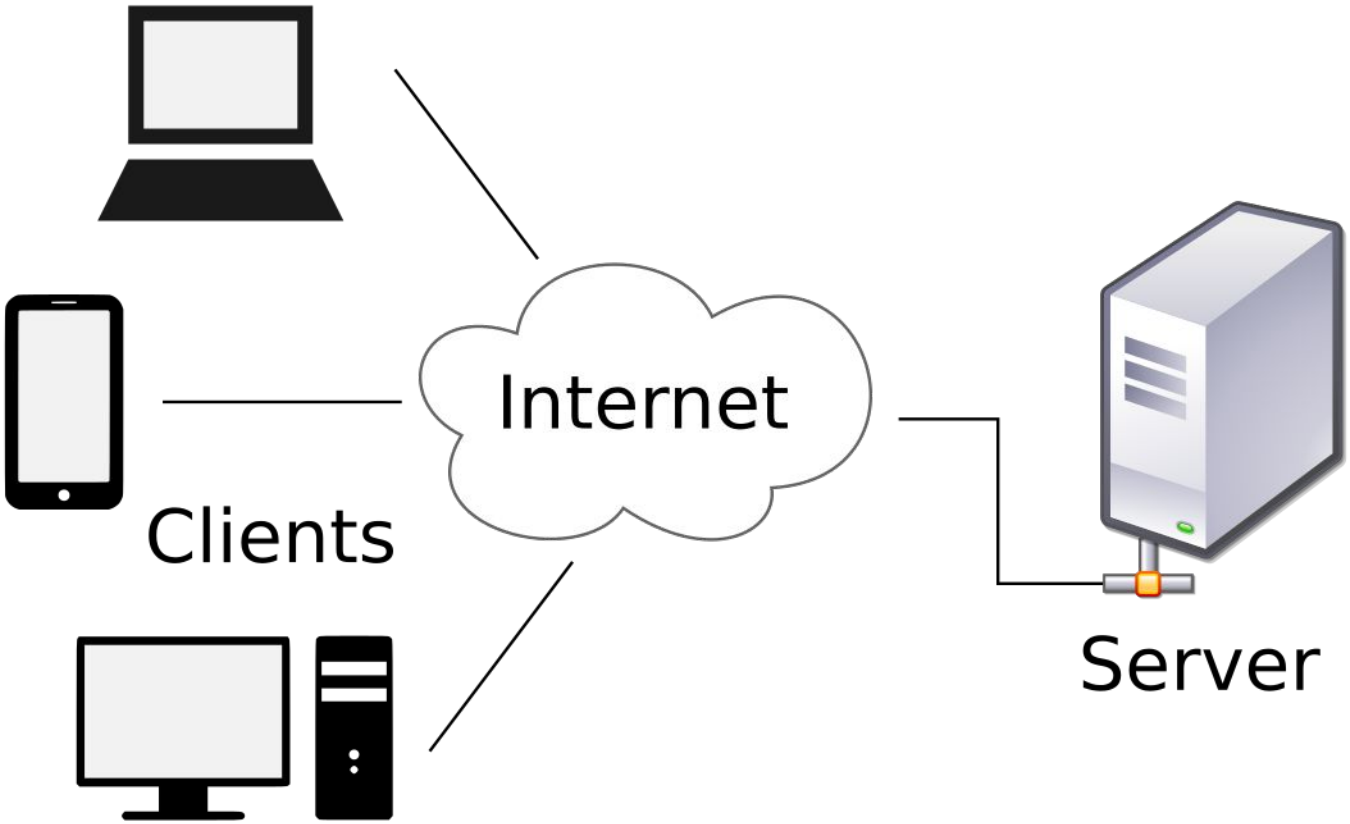
- Software developer
- Database developer
- Tester
- Business analyst
- Release manager
- DevOps
- Administrator
- Solution Architect

Technology Stacks

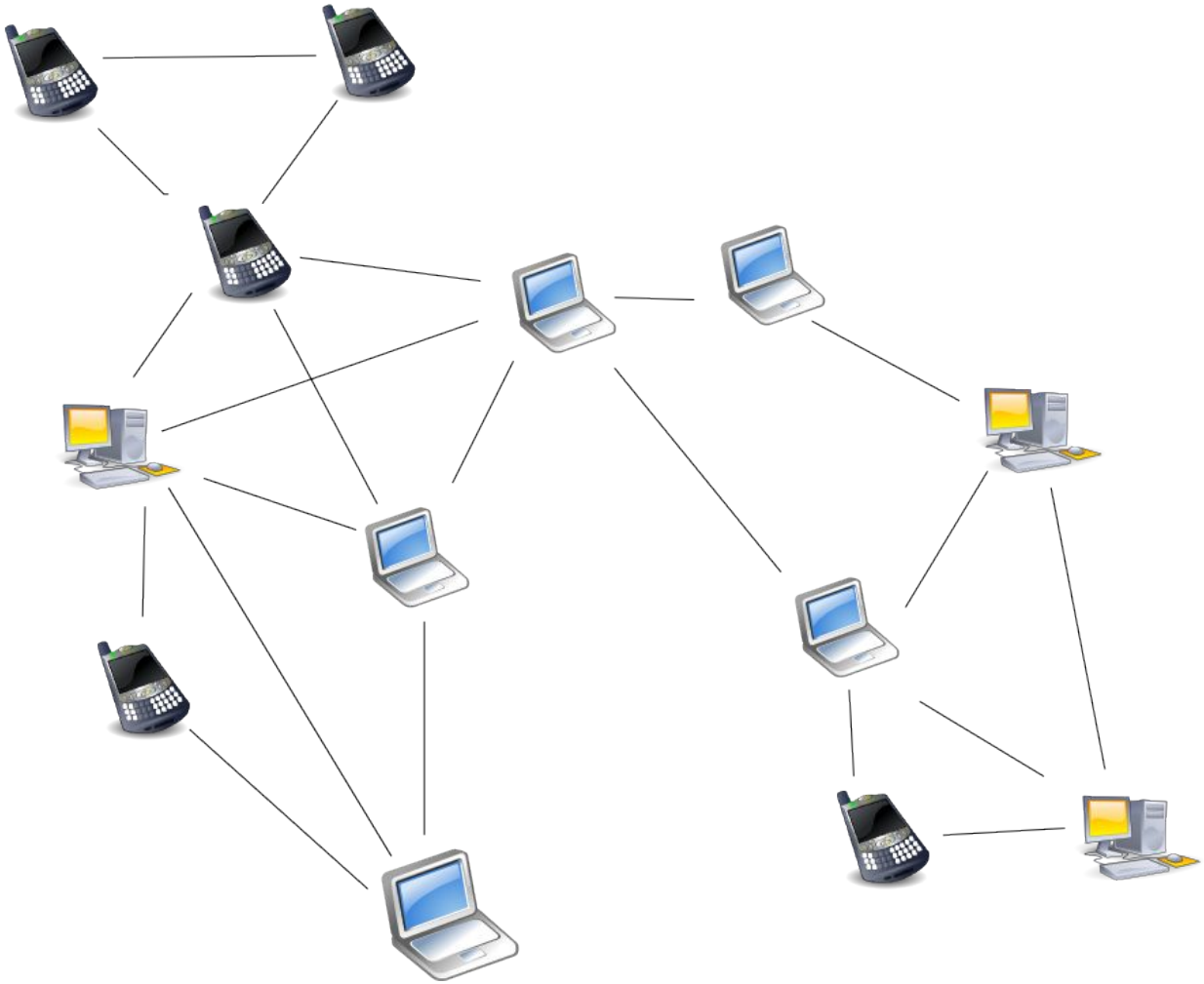
- Cloud provides, On-Premise, On-Demand
- OS – Windows, Linux, Hybrid
- Programming languages (Compilers & Interpreters, C/C++/Java/C#/PHP/Python/Go/D/Erlang)
- Data storages – RDB, GraphDB, NoSQL
- Frameworks, Ecosystems (Boost/Spring/Laravel, Java/.Net)
- Tools, libraries

Types of Architecture – Standalone application

Types of Architecture - Client-Server application



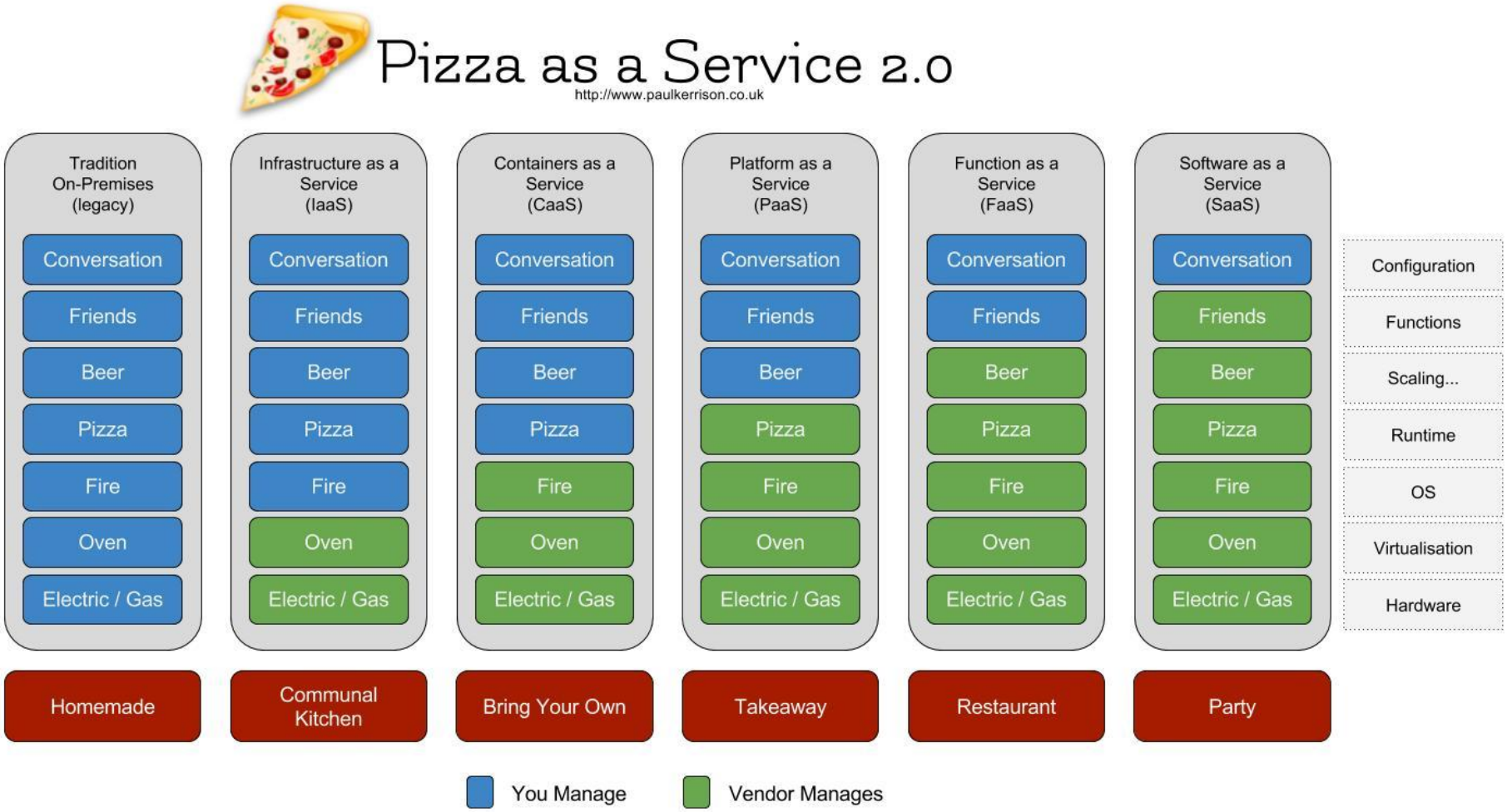
Types of Architecture – Peer-2-Peer



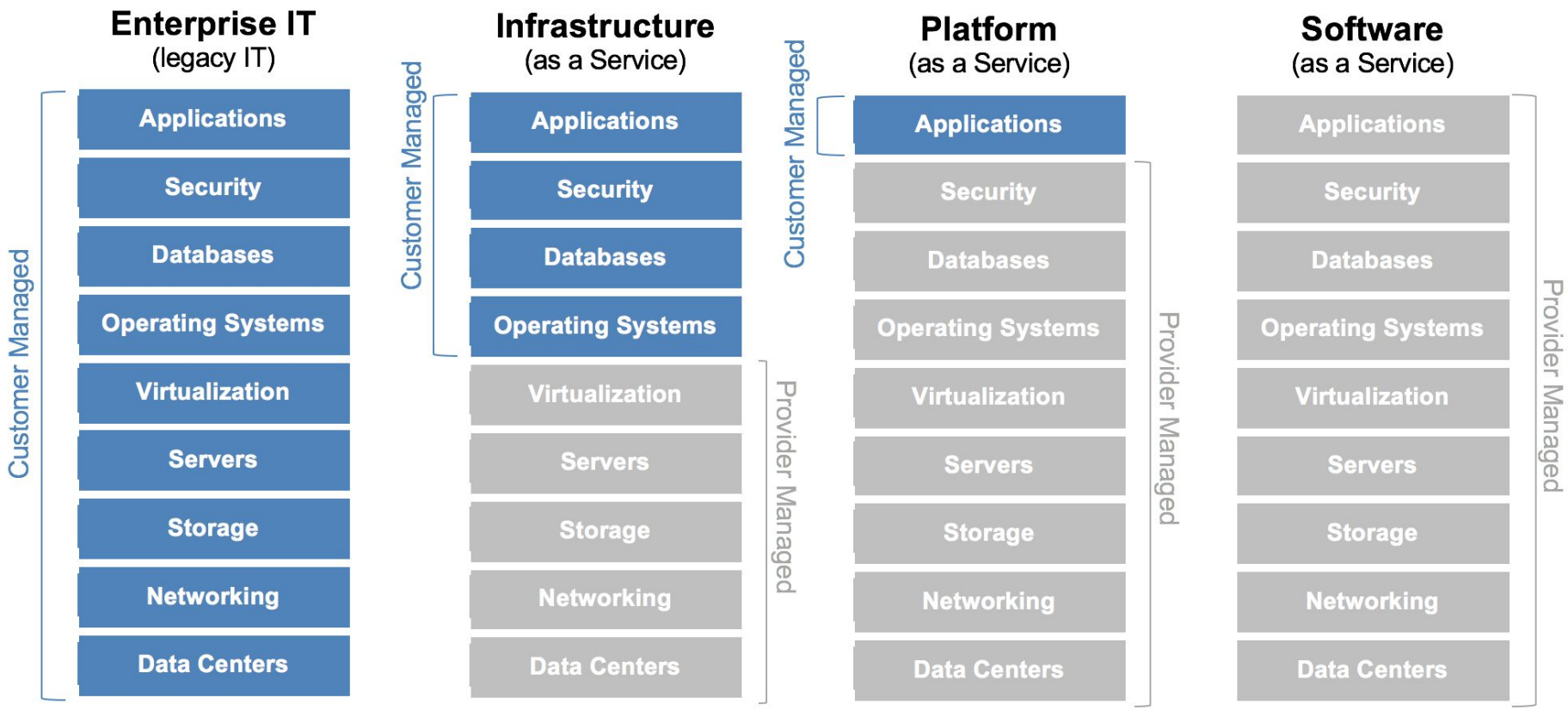
Types of Architecture - Monolith vs Micro services

Cloud is just someone else's computer

Clouds



Clouds


Enterprise IT
(legacy IT)
Infrastructure
(as a Service)
Platform
(as a Service)
Software
(as a Service)
Customer Managed
Provider Managed
Applications
Security
Databases
Operating Systems
Virtualization
Servers
Storage
Networking
Data Centers

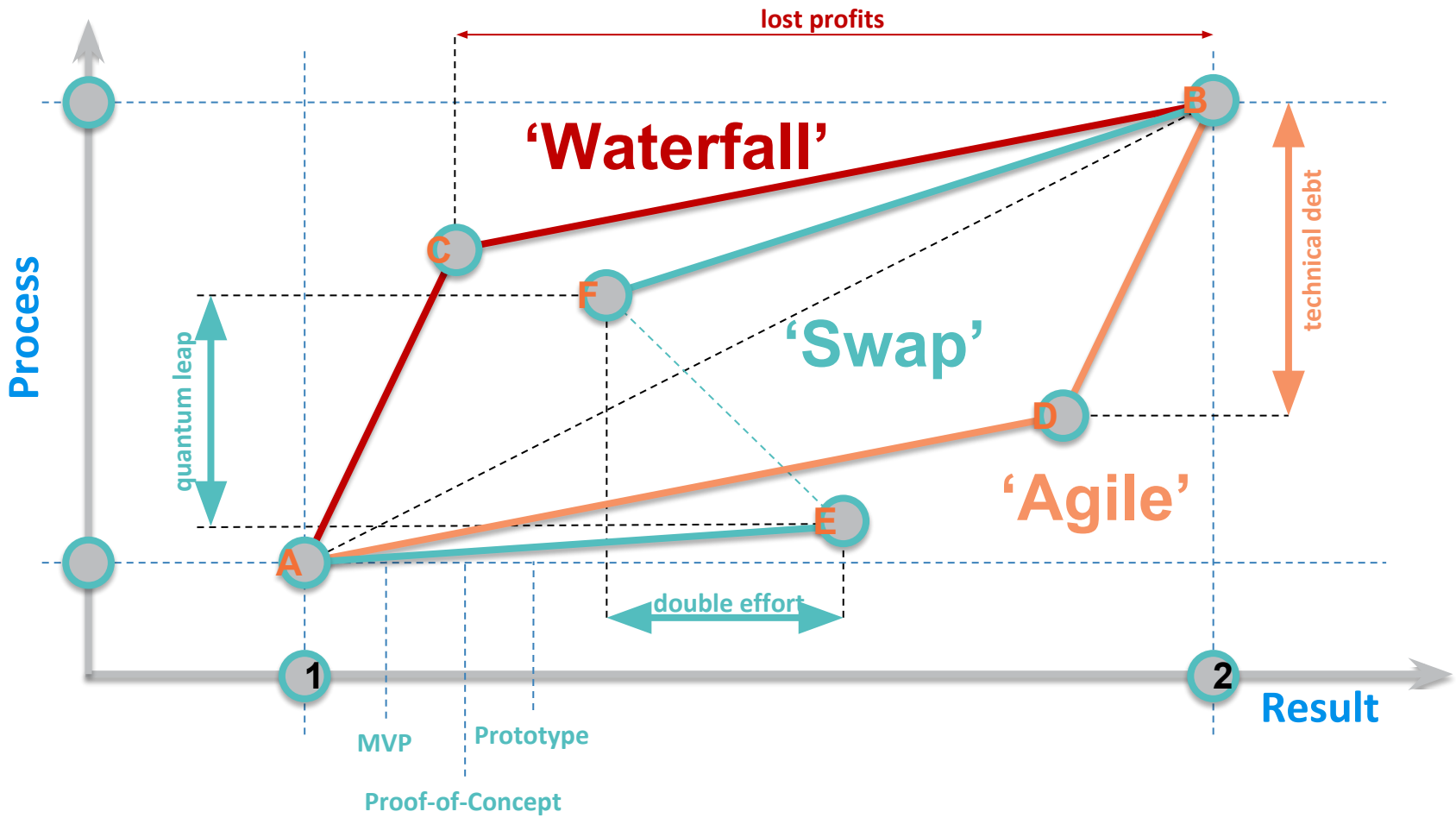
#noOPS #noPM #noDEV

- noOPS – **maturity** of the dev tool and frameworks are already raised at the level where every developer **can handle all/subset** of **Ops** activities
- noPM – communication & collaboration tools allow to build **peer-2-peer** channels and avoid bottlenecks
- noDEV – **Best Software is a Software was never written**. Huge number of open source and proprietary software shits development into integration side

Development Methodologies

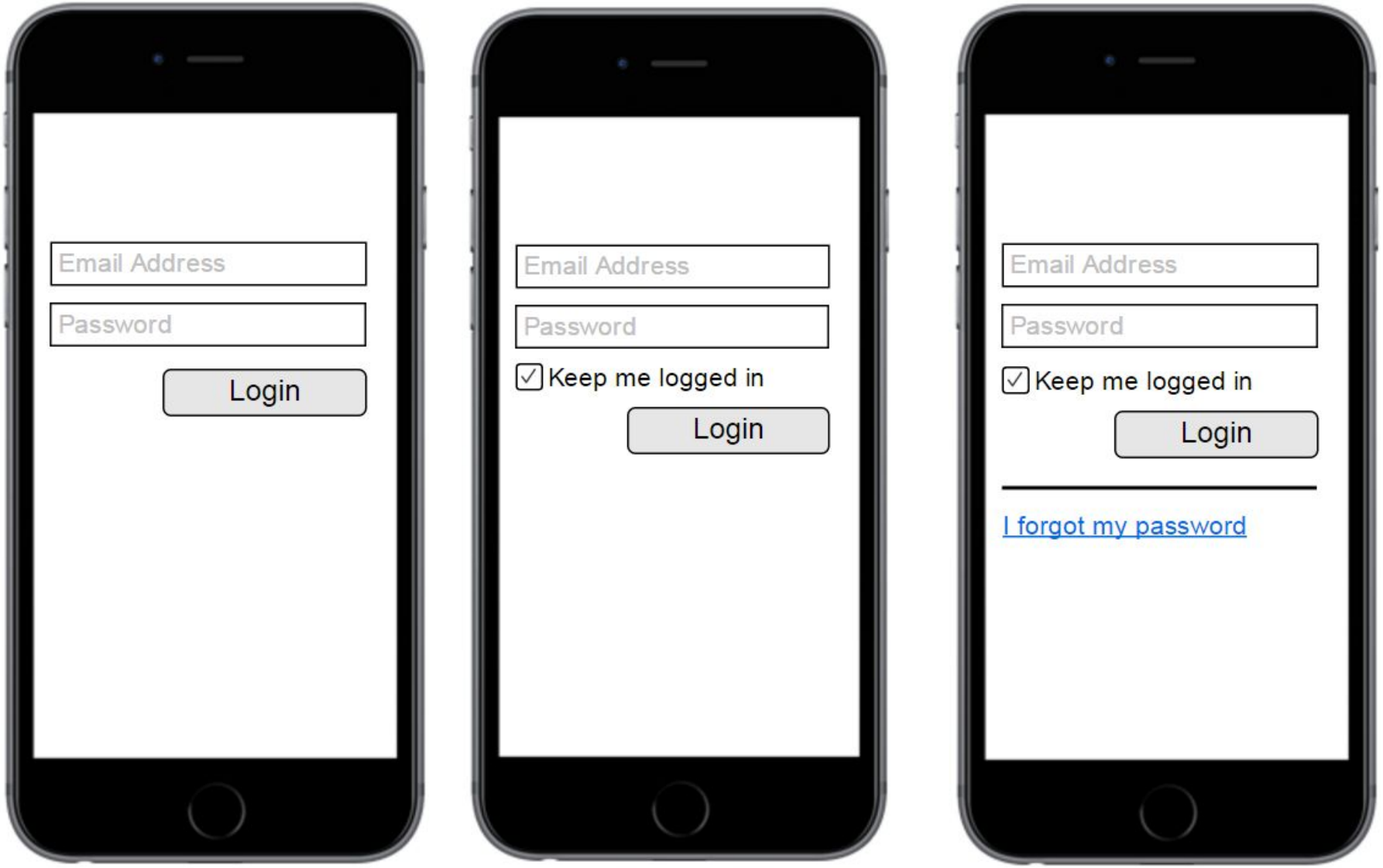
Waterfall, Agile and more …

Result and Process, Result vs Process

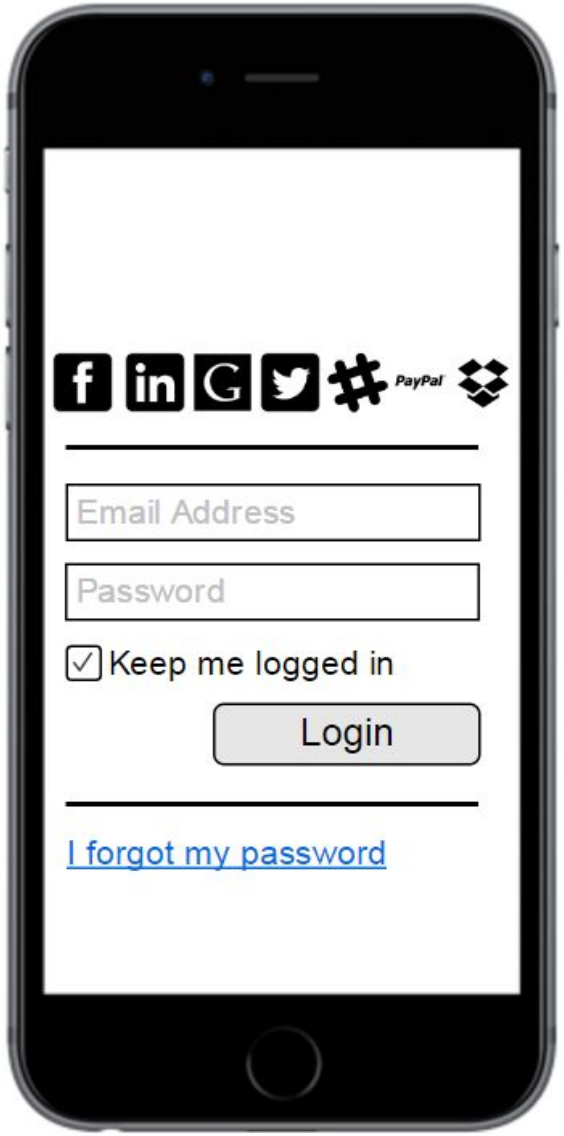


External software quality factors

Evolution


Email Address
Password
Login
Email Address
Password
Keep me logged in
Login
Email Address
Password
Keep me logged in
Login
I forgot my password

Evolution


PayPal
Email Address
Password
Keep me logged in
Login
I forgot my password


Continue

Evolution

```
void login(
    std::string const& email,
    std::string const& pass) {
}
```

```
class provider {};
class service: public provider {};
class facebook: public provider {};
class linkedin: public provider {};
// etc.
template<bool const T>
void login(T liveForever) {
}

template<class T, class... Args>
void login(T provider, Args... args) {
    // login with provider
    login(args...);
}
```

Key factors

- Correctness (Корректность)
- Robustness (Устойчивость)
- Extendibility (Расширяемость)
- Reusability (Повторное использование)

- Compatibility (Совместимость)
- Efficiency (Эффективность)
- Portability (Переносимость)
- Easy of Use (Простота использования)
- Functionality (Функциональность)
- Timeliness (Своевременность)
- Verifiability (Верифицируемость)
- Integrity (Целостность)
- Repairability (Восстанавливаемость)
- Economy (Экономичность)

Legacy systems & Technical debt & Refactoring

Legacy system - How is this happening

Legacy system is a system where technical expertise/knowledge was lost

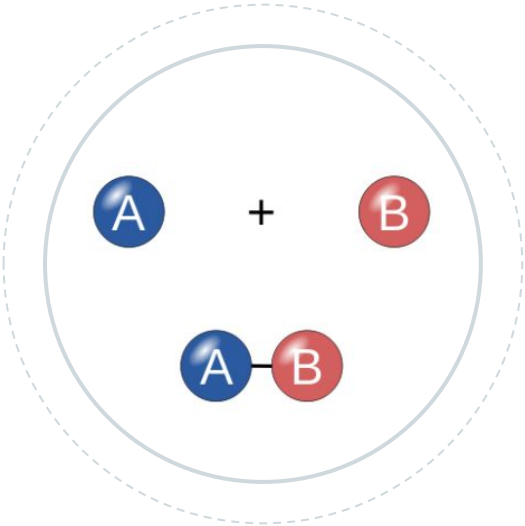
- Software with a history (~5+ years)
- A set of major releases
- Long evolution of source code
- Extending with new features
- Reuse inside new domains
- Integration with other systems

Legacy system is your code after one month without unit-tests and documentation

Refactoring

Increasing internal software quality without changing users' behavior (system interface)

- Decrease code coupling
- Clarify interfaces
- Remove God-objects
- Rewrite code using patterns
- Improve documentation
- e.t.c



Decomposition
Composition
Verification

Why is Software Development complicated?

- Speed of tech changes, legacy products
- Misunderstanding between business owner, management and tech staff
- Flexibility of tech tools
- Math model will never be equal the original system

Cycle of life