Software engineering perspectives in Virtual and Software Defined Networks

Tihana Galinac Grbac
Faculty of Engineering
University of Rijeka
• Motivation

  ▫ Emerging technologies and new network paradigms influence software lifecycle processes (Open network, ‘As a Service’, IoT, service compositions,)
  ▫ Quality of Service --> Quality of Experience
  ▫ Information and communication (ICT) become enabling technology in many application domains and here we need real software engineering work:
    • If service is realized by software what is changed in quality requirements on software
    • How network complexity reflected on software systems
Software engineering?

- "the systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software," (“Systems and software engineering - Vocabulary,” ISO/IEC/IEEE std 24765:2010, 2010.)
- and "the establishment and use of sound engineering principles in order to economically obtain software that is reliable and works efficiently on real machines", ("Software Engineering". Information Processing (North-Holland Publishing Co.) 71: 530–538. 1972.)
How we can define software engineering profession?
What kind of software do engineers produce today?

- Complex
- Reliable
- Distributed
- Real time demands
- ....
Emeering network technologies

- Convergence of IT and communication software
- End user devices with unbounded possibilities
- Virtual infrastructure providing just imagination of stabile phisical hardware
- Problems with vendor locking solutions
- Open platform movement
- Problem with interoperability
- Middleware platforms
More and more software systems tend to evolve towards complex software systems (e.g., IoT) and systems of systems (SoS).

Interconnection of peripheral systems over distributed network into system of systems (IoT).

Main trends in telecommunication network
Software in ‘Internet of Service’

- In service oriented architecture (SOA) software is provided ‘as a service’
- In that concept ‘of service’ is referring to a technical understanding of software functions provided as Web service
- IoS combine that services and integrate functionalities that led to complex service chains
- Usually these service chains are developed by number of providers and offered to number of users
- Service chain composition is happening at layers above network layer
- Problem is how to secure quality of these service chains
- We need algorithms for autonomous control for a reliable IoS
Service Oriented Architecture

Software function developers

- v1
- v2
- v3

Software specialized for particular function (Example: Signal denoising)

Software application developers

- Medicine
- Pharmacy
- Ecology
- Agromony

Application Software (Example: Medical software, cancer detection from MR)

Equipment providers

Different equipment used is service realization

*Figures reused from http://www.freedigitalphotos.net/
Modern computing?

Internet of things

Vehicle, asset, person & pet monitoring & controlling

Agriculture automation

Energy consumption

Security & surveillance

Building management

Embedded Mobile

M2M & wireless sensor network

Connecting everyday things

Smart homes & cities

Telemedicine & healthcare

Modern computing?

Modern computing?

The Internet of Things

https://www.theregister.co.uk/2014/05/07/freescale_internet_of_things/
Software Defined Network

Application plane

SDN Application

SDN Northbound interfaces

Network Control plane

Northbound API (e.g. REST API, Python API, Java API)

Control functions implemented in modules

Southbound API (e.g. Open Flow)

Global Network View
- Flow tables of SDN devices,
- Topology of SDN device,
- Traffic statistics,
- SLA agreements

SDN Southbound interfaces

Data plane

SDN Device

Southbound API

Forwarding function

Flow table

Update

Read

SDN Device

Southbound API

Forwarding function

Flow table

Update

Read

SDN Device

Southbound API

Forwarding function

Flow table

Update

Read

SDN Device

Southbound API

Forwarding function

Flow table

Update

Read
Modern networks and software

- Software become central part of the modern network

- Requirements:
  - It should run on any hardware,
  - serve to many users,
  - satisfy their complex communication needs,
  - deliver proper ICT service, effectively and efficiently and
  - has to be flexible on network context, information context, communication context, ....

- Modern network should provide **reliable and robust** ICT services (resistant against system failures, cyber-attacks, high-load and overload situations, flash crowds, etc.)
Revolution or evolution of software systems

- Future: Communicating software systems distributed over the network, autonomously managed
  - Networks of networks, Systems of systems,
  - Interconnected by Internet network
- Software services realized as service chains ad-hoc established per each user or group of users
Automation of software engineering knowledge

- Software lifecycle phases:
  - Software requirements
  - Software design
  - Implementation
  - Test
  - Maintainence
Key problems with software engineering evolution

- More and more software systems tend to evolve towards complex software systems (e.g. IoS)
- Interconnection of peripheral systems over distributed network into system of systems (IoT)

Key problems become:
- Can we develop foundations on software behavior?
- How can we measure software behaviour in network?
- Can we predict and simulate software behaviour in network?
- How to manage complex software system?
- Are we able just by observing properties of system parts to predict and model its overall behaviour?
Conclusion

- Do we have enough software engineering knowledge to provide software for new network technologies?
- Can we skip software engineer from software development process?