VIRTUALISATION TECHNOLOGIES IN SOFTWARE ENGINEERING MANAGEMENT

Tihana Galinac Grbac
Faculty of Engineering
University of Rijeka

EVOSOFT: UIP-2014-09-7945

Instalation Research Project
Evolving Software Systems: Analysis and Innovative Approaches for Smart Management (EVOSOFT)
Title: Virtualisation technologies in software engineering management
MOTIVATION

Complex system behaviour
As we analyse within EVOSOFT and ACROSS projects
Key problems with software evolution

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

- Complex systems did not evolve accidently
- Huge effort is invested - there must be a great interest to grow into complex system
- Developed in sequence of projects over decades
- Mostly perform tasks that are
  - of crucial importance for community (defense, energy, public services, banking, health)
  - for very large number of end users (telecommunication)
- Quality is of crucial importance
Key problems for future software engineering profession

• 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 behavior?

• These are the main objectives of the projects:
  • Evolving Software Systems: Analysis and Innovative Approaches for Smart Management EVOSOFT (HRZZ) and
  • Autonomous Control for Reliable Future Networks and Services ACROSS (COST)
REQUIREMENTS FOR MANAGING COMPLEX SOFTWARE SYSTEMS

Motivation for our studies within EVOSOFT and ACROSS projects
System requirements for future systems

- Generic software to serve many users and many user needs
- Parallel execution of multiple different requirements, for number of users
- Software independent of technology and hardware
- High availability of services (software, hardware, platform) for its users
  - If certain malfunction happen the peers has to be timely informed, and all related resources properly released, avoid congestion situations
- Properly dimensioned – avoid load, runtime dimensioning
- Response by the required time
  - Real time system, a system with a real-time constraints
- Interoperable with other vendors equipment
System requirements for future systems (cont.)

- Easy to maintain and manage
  - System divided into logical functions
  - Well defined and separated logical functions
  - Easy to trace system dynamics
  - Easy transformed from object code back to original code
  - Pricing and billing scheme
  - Runtime quality secured, ad hoc reconfiguration
Solution - Migration to virtualized environments

- Solution in new software abstractins, network management concepts
- Service orientation
- Trend is to provide everything ‘as a service’
- Network is provided to its users ‘as a service’ by providing:
  - Infrastructure (processing, memory)
  - Operating platforms
  - Software applications
- Numerous users may get any network resource as a service and pay per use
Software specialised for specific function (Example: Signal denoising algorithm)

Software for specific functions is needed to accomplish functionalities of application software in different application domains (e.g. Knee analysis)

Application domain
Different kinds of terminals and end equipment

*Photos used from http://www.freedigitalphotos.net/*
Virtualisation requirements

- Hardware/service providers may be anybody and anywhere
- Service provider may be anybody, without exhaustive testing or certification software service
- Dynamic contracting among
  - Cloud providers (hardware and platform providers)
  - service provider and service user
  - Cloud providers and its users
- Traditional telecommunication networks were developed by few development organisations driven by standards
- Exhaustive network testing of end user functions have proceeded before network use
- In new virtualisation environments ad hoc system creation is possible
Management of Services

• We need to understand service and environment behaviour
• Services may be measured within the Cloud environment
• Service price and SA may be evaluated and compared among number of executed cases
• Entity behaviour may be determined from history:
  • Service (quality of service executions, popularity)
  • Service requester (his most favorite services)
  • Service provider (quality of his services)
Service Chain Management

- End user requirements may be realised through composition of services in service chains
- Dynamic management of each service may affect the service chain performances, quality of service

Group of users interested in complex service

Services are dependent

Service chain
SERVICE ORIENTATION AND VIRTUALISATION

A case study
Virtualisation features (openStack)

- Automatic Scaling
- Load Balancing
- Service Orchestration
- Runtime CRUD operations
- Runtime Reconfiguration
- Service Chain and Service Group
Student projects in virtualisation technologies for software engineering management

- Faculty of Engineering, University of Rijeka
- Graduate Study Programme of Computing
- Course Software Engineering Management, 7 ECTS
- 4 hours lectures – methods and technologies for software engineering management, discussions, reflections on projects
- 2 hours project – real project that present the use of some of the software engineer technology
- Motivated by industry
Free call service in Intelligent Network

- Service concept existing traditional telecommunication

So, what is novel?
Service/feature composition is well known problem in switching systems

*Book: 100 years of telephone Switching
http://books.google.ro/books?id=07NmhqkOqwsC&printsec=frontcover#v=onepage&q&f=false

*I. Lovrek, lectures, Faculty of Engineering and Computing, Zagreb,
Figure 12. Free call service in Intelligent Network
Benefits of service binding through Service Registry

- **Service provider may**
  - Dynamic change of web services during runtime
  - Perform CRUD operations (Create, Run, Update, Delete) @runtime
  - do not have to maintain track of Service users to perform service management

- **Service requestor may**
  - choose among number of services without explicitly knowing service address (binding)
  - may switch among service provides @runtime

- **Cloud provider may develop recommender systems to secure justice and harmony for its users**
  - Measurements of service, service provider’s and service consumer’s behaviour

- **track record of list of available services via service registry**
Problem:

- In traditional networks system verification activities have secured system reliability
- How these system properties will be secured in terms of these dynamic systems?
- How can we predict and model system behaviour in such dynamic environment?
- Can we predict service composition behaviour from local properties of each service in composition?
How can we secure reliable operation of stateful service chains in Cloud

- Run time testing
  - Testing combination space is reduced with additional knowledge from the runtime environment

- Behavioural type theory encompasses concepts such as interfaces, communication protocols, contracts, and choreography.

- As structural principle for building reliable software systems

- Idea:
  - to codify the structure of communication to support the development of reliable communication-oriented software.
  - to encode as types the communication structure of modern computer systems and statically verify behavioural properties about them
Managing Contracts

• Traditionally the end user services were few, and tested in network that is built based on fixed and known contracts.
• The most commonly used methods for ensuring the correctness of a system are simulation and testing.
• Exhaustive for any reasonably complex system is impossible.
• Errors can sometimes occur only for specific execution sequences which are difficult if not impossible to reproduce or debug, making an exhaustive analysis necessary.
• Cloud network is introducing dynamic contracting at all layers – impossible to test all situations.
• A Service Agreement (SA) represents a binding agreement between the provider and customer of a cloud service.
STANDARDISATION AND RESEARCH DIRECTIONS
Standardisation: Cloud Services

- **European Telecommunications Standards Institute (ETSI)** launched the Cloud Standards Coordination (CSC) *
  - Summarize relevant standards for Cloud services to their users and service providers
  - Identify and Collects Cloud Service Use Cases

- **Service Measurement initiatives:**
  - Service Measurement Index
  - Cloud Services Measurement Initiative Consortium (CSMIC)

*European Telecommunications Standards Institute (ETSI) Cloud Standards Coordination (CSC) report:
Standardisation: Network Function Virtualisation: ETSI GS NFV

• **Network Function Virtualisation: ETSI GS NFV**
  - Aims to transform the way that network operators architect networks by evolving standard IT virtualisation technology to consolidate many network equipment types onto industry standard high volume servers, switches and storage, which could be located in a variety of NFVI-PoPs including datacentres, network nodes and in end user premises.

• **NFV Managmant and Orchestration Architecture**
  - Published E2E Arch, REQ, Use Case, Terminology documents in:
  - ETSI NFV Open Area:
    - http://docbox.etsi.org/ISG/NFV/Open/Published/
  - Published ETSI NFV white paper:

*Network Functions Virtualisation (NFV); Use Cases
http://www.etsi.org/deliver/etsi_gs/nfv/001_099/001/01.01.01_60/gs_nfv001v010101p.pdf
Research direction 1: How can we secure reliable operation of stateful service chains in Cloud

• Run time testing
  • Testing combination space is reduced with additional knowledge from the runtime environment

• Behavioural type theory encompasses concepts such as interfaces, communication protocols, contracts, and choreography.

• As structural principle for building reliable software systems

• Idea:
  • to codify the structure of communication to support the development of reliable communication-oriented software.
  • to encode as types the communication structure of modern computer systems and statically verify behavioural properties about them
Research direction 2: QoS Aware service compositions

- Adaptive execution of scientific workflows
- More efficient and diverse service composition
- A very large-scale reliable service composition
  - Find and consume the current best-fit
  - Among the multiple implementations or deployments of the same service.

*Pradeeban Kathiravelu, Tihana Galinac Grbac, Luís Veiga: Building Blocks of Mayan: Componentizing the eScience Workflows Through Software-Defined Service Composition, Accepted for ICWS 2016, San Francisco, USA.
QoS Aware service compositions

- Based on performance measurements collected in controller

*Pradeeban Kathiravelu, Tihana Galinac Grbac, Luís Veiga: Building Blocks of Mayan: Componentizing the eScience Workflows Through Software-Defined Service Composition, Accepted for ICWS 2016, San Francisco, USA.*
QoS Aware service compositions
Conclusion

- Service management challenges are originating from virtualizing execution environment and enabling dynamic change:
  - Change of software development paradigm from developing software for particular hardware to developing a generic service
  - Runtime adaptation mechanisms based on history behaviour (of services, cloud environment, etc.)
  - Need for extensive empirical studies, integration of empirical measurements and studies, efficient analyses algorithms
  - Open and standard interfaces, design rules for composing complex systems have to be inherently integrated into development environments and technologies
  - New information management concepts that would hide private details but provide benefit for autonomous system control
  - Reliable autonomous systems have to solve challenges of service management
  - Importance of reliable autonomous systems are for cloud systems in mission critical domains such are energy networks, health care, automated home environments
Hržić, F., Poljančić N., Galinac Grbac, T: Secure operations as congestion control mechanism within OpenStack based Cloud Laboratory, International Conference on Smart Systems and Technologies (SST), 2016, Osijek, Croatia

Pradeeban Kathiravelu, Tihana Galinac Grbac, Luís, Veiga: Building Blocks of Mayan: Componentizing the eScience Workflows Through Software-Defined Service Composition, Accepted for ICWS 2016, San Francisco, USA.

Tankovic, N; Galinac Grbac, T; Truong, H-L.; Dustdar, S: Transforming vertical Web applications into Elastic Cloud Applications, Proceedings of International Conference on Cloud Engineering (IC2E 2015), 9-12 March, 2015, Phoenix, USA.

Galinac Grbac T., Runeson P: ’Plug-in’ Software Engineering Case Studies, CESI – ICSE 2016 Workshop, Austin, USA


Galinac Grbac T., Runeson P, ’Plug-in’ Software Engineering Case Studies, CESI – ICSE 2016 Workshop, Austin, USA