Alexandru Ioan Cuza University of Iaşi, Romania
Faculty of Computer Science
2018

Abstract

Habilitation
Integration and Cloud Computing

by Lenuta Alboaie

This thesis presents the author`s main research and contributions since 2009 after the PhD defence. Also, active research areas and a personal development strategy are proposed. The thesis approaches the integration line of distributed systems and of Cloud Computing. Domains which have been approached in our research revolve around this axis: Semantic Web, Social Networks, Integration and IoT. An orthogonal aspect related to all these directions, which appears as an important area in our present and future research is represented by privacy related concepts.

The Thesis is structured in five chapters with references for each. All chapters describes the context and the author's most significant contributions.

In **Chapter 1** we present multiple aspects related to the concept of integration. We have sought to identify exactly those integration-related elements that would show our position on this concept the present thesis is centered on.

In **Chapter 2** we outline an image on what Cloud Computing represents. The alert evolution of Cloud technologies, multi-cloud adoption, IoT development led to complex systems wherein the integration issue represents a veritable challenge. With an expertise of over 15 years both on teaching and research in the field of distributed systems, we began this chapter from incipient notions of Cloud Computing, such as Cluster/ Grid Computing approached by the author since 2000 and we led the presentation path towards emerging trends in Cloud Architectures (Microservices, Serverless Computing, Fog Computing, Edge Computing).

Throughout the chapter we have sprinkled results that have been obtained in synchronicity with the technological evolution of the period.

In the *SOA principles applied* section, based on the experience achieved since the very occurence of the SOA concept by writing a book in 2006 on Web Services, we present a SOA architecture developed during a research project ([Telemon07]) concluded in 2010.

In this project, implemented into a consortium of three universities and an IT company I activated as a member specialised on Web Services. In that period when there were present numerous legacy e-health systems, creating problems for the medical professionals (e.g. spend time to search (vital) information located in many places), we proposed a system based on the SOA paradigm providing functional solutions to the aforementioned issues. Furthermore, the practice in the field made possible to coordinate in 2014 a project called SIMAPS - Monitoring and assistance system for people with special needs, implemented by a consortium of three universities and an IT company. Its goal was to develop a services oriented architecture system, which could be used on intranet into hospitals or on Internet (e.g. hosted in Cloud), in order to keep the contact between the patient and the clinician.

Also in this section, me make notice on the applied research results, achieved on doctoral study in the field of Social Network during a research project MUKE - Multimedia and User Credibility Knowledge Extraction. As a member, I contributed both to the SOA architecture of the system and to create an images recommendation system module. This employs ratings to compute similarities and with social profiling to introduce diversity in the list of suggestions.

The teaching activity in the web semantic field initiated in 2009 has opened novel research horizons. In 2012 I began to coordinate a research project (Cloud semantic engine for governmental heterogeneous information from multiple data sources) whose implementation consisted in successfully combining the trends of Semantic Web, services oriented architectures and Cloud Computing.

At the moment this paper is written, we are in full technological effervescence regarding potential Cloud architecture alternatives. In this respect, in the second chapter, we have compiled an overview on the current status and the potential future, enabling us to pinpoint our position both on research and experimental development levels.

In **Chapter 3** we conduct an almost didactical survey of the challenges, characteristics and solutions associated to integration, taking into account aspects such as protocols used, data format, invocation calls performed, application specific characteristics, monitor and logging, mechanisms to detect and correct errors.

In order to develop complex services oriented architecture, aspects such as messages routing, data transformation, service orchestration, transactions execution are part of the integration problem and it is solved by employing ESB - Enterprise Service Bus solutions or iPaaS systems. In the third sections, these were assessed both technically and from their impact perspective. Also, it was depicted an exhaustive image of the ESB, SOA or BPM roles in a real system.

Taking into account the expertise on semantic web I have previously mentioned, in this section we approached integration also from this perspective, this time conducting modelling series on semantic data integration level.
Chapter 3 concludes with a suggestively named subchapter *Integration Roadmap*, wherein we encompass the general dimension of the problem and make a possible proposal on the way our approach described in chapter 4 and 5 may be able to successfully address the existing integration open problems.

In the **4th chapter** we introduce a series of widely applicable results. Our proposal - named Swarm ESB - occured at the proper moment, both from the international technological context perspective and as a way of addressing the market requirements. At the end of 2010, studies indicated that one out of five SOA projects failed due to multiple causes: from social and organisational ones to technical ones. As mentioned in the fourth chapter, one of the major failure triggers was related to high expenditures, in the context in which SOA architectures require architects, ESB experts, programmers, security experts, testers, business process modelers et.al. The whole mechanism itself needs a training process and after that one may begin talking about productivity. If this process proves to be challenging for large companies, the smaller businesses are even more reluctant to adopt expensive solutions. Our expertise on SOA architectures enabled us to understand the big picture with its strong and weak points. Therefore, knowing that most of the time solutions do not take a full SOA approach because of the technical complexity, we have introduced in our swarm based architecture only a few intuitive concepts like phases, adapters, groups, swarm description and swarm primitives and this proves to be a huge benefit for rapid learning. Through swarm communication we propose a new paradigm regarding communication mechanisms and composability of services. Swarm communication brings a new perspective compared to how service orchestration and choreography is done in current ESB systems. The goal of swarm communication, as in the case of service integration patterns from SOA, is to compose services' behaviours. Swarming, as we define it, is different from service orchestration because there is no central controller that manages the business logic and messaging sequence. The swarms are different from choreographies because standard choreographies are descriptive only (e.g. WSCDL), or they are only a representation resulted from the execution of multiple entities whose behaviors together carry out an integration process. We can say that swarming is somewhere in the middle between service orchestration and other approaches for service choreographies. The lack of a central orchestrator makes a swarm based system inherently more scalable. By using swarm communication, a system will have the ability to dynamically scale, meaning it will be capable of handling variations in capacity requirements in an automated mode.

Furthermore, the increasing popularity of microservices, an architectural style implying systems based on very granular and independent sets of services that collaborate, created the requirement for novel approaches and pragmatic languages able to integrate and coordinate them. The swarms are providing a natural environment to create and compose microservices

and may be viewed as a new distributed programing technique meant to facilitate these software development aspects. The adaptors of a SwarmESB architecture are implicitly hosting processes that host microservices at their turn, while the swarm communication may intrinsically provide integration between heterogeneous systems or implement various business processes. Furthermore, it is worth mentioning that our vision on which we began to work upon in 2011 and published in 2013 was reinforced by other researchers and practitioners as indicated in section 4.1.

These steps have built the core functionalities or *Functional requirements* for a light ESB. An ESB may be viewed as a distributed backbone upon which an enterprise can build various SOA architectures or microservices oriented architectures. Different business models may be designed over these and in section 4.2 we indicate how swarms are also providing a mechanism to describe long and short living business processes.

The swarms are keeping the benefits of asynchronicity to communicate through messages, while the phases of communication are accurately identified and set; this communication becomes an executable choreography that has a global effect in the system. The executable choreographies are introducing new types of abstractions that the machines may formally (automatically) verify and thus provide user trust. Due to the fact that these choreographies may be formally verified, we may ascertain that they behave as true verifiable environments for cloud applications. From this perspective, three executable choreographies were proposed: verifiable, encrypted and serverless. As mentioned in 4.2.3, the platforms that are allowing executable choreographies are still incipient. However, they hold the potential to contribute to the development of services oriented architectures and cloud applications from various perspectives, including privacy.

Beside properties such as scalability and availability we have also envisioned creating mechanisms able to ensure the privacy of systems built on SwarmESB. The need also resided in a real project, SIMAPS, where the SwarmESB could be employed as middleware for the desired SOA architecture. However, at that time, there were no privacy mechanism available, mandatory for the medical data similar to what we are working on at the moment.

Since 2015, once the PrivateSky project was written, we have begun research also on this field. There were identified several levels regarding data privacy in executable choreographies based systems, by estimating the risk of unauthorised user accessing private data. Potential unauthorized access may come from users within the organization who access private data that is outside the scope of their post duties, or may be caused by external attacks. A series of preliminary results regarding the privacy by design implication of executable choreographies are presented in section 4.2.4.

As previously observed in available publications, our vision and models were mostly at least on par with the technical evolutions in the field. Following this trend we approached in section 4.2.5. the connection between SwarmESB and IoT. Internet is a network of networks, but IoT is going to be more about integration then about networking. The best architecture for

IoT systems seems to be a bus architecture. A bus provides an integration environment for systems that can easily evolve by connecting and disconnecting “smart things” like sensors, devices, services under an orchestration dictated by the constantly changing requirements. The concept of “integration bus” is essential for our proposal: a new architecture and a new approach to build integrated applications for IoT around ESB like systems. Our tests proved that even if a series of modifications such as replacing the messages management system with an IoT specific one were required, SwarmESB may be used in an IoT environment. For IoT we envision that an evolved version of SwarmESB could be part of the standard software provided in network routers or could stay in the cloud as part of an iPaaS, each instance being easily configured and programmed to solve specific integration use cases. Until now we succeeded in unifying the integration aspects (like message routing and transformation, monitoring) with long living processes (workflows) in a common execution model. In the future, reusing existing programming models, multiple SwarmESB instances will be integrated at a higher level of integration, providing a controlled environment with respect to security, privacy and enabling new types of applications for IoT and cloud. We envision a world where there will be some standard protocols to describe integration using swarm communication and to report knowledge about the user cases performed or to be performed by the integration layer.

Chapter 5 brings into focus recent research, meant to develop technologies for PrivateSky project and the subsidiary agreements (signed with IT companies) currently established in the field such as Personal Assistants or smart-society.

In section 5.1 we suggest an approach on the concept level, however, based on executable choreographies enabling software architectures in which private data storage places are under the strict control of the user's personal assistant. Being a software system, a personal assistant may be able to authorize or refuse access to private data in real time but also to take into consideration in an intelligent manner all the user’s preferences.

On the other hand, our experimental and practical concerns regarding the integration between various cloud systems have led us to the observation that there is a tight link between two efforts: building personal assistants and following the smart systems trend. By “smart systems”, we understand the complex integrated systems including mobile applications, software systems for smart cities, smart communities and other various applications with IoT flavour. Smart systems integrate technology, organizations and people in order to accomplish complex processes that are controlled by computer systems.

In this environment, in section 5.1 we propose a roadmap towards personal assistants through privacy and integration perspective, roadmap that will follow/develop/implement in the following years in PrivateSky.

In the sections 5.2 and 5.3, systems developed on SwarmESB in the 2017 PrivateSky version are described. In the section 5.2 we have analyzed the potential to adapt SwarmESB to the requirements of developing a decentralised application (DApp). In section 5.3, starting from the real situation of our Romanian health insurance system, we have modeled and implemented a

system based on swarm communication, centered on user data privacy, its main purpose being to help software services providers and public institutions to comply with the General Data Protection Regulation.

The tests and the conducted analysis, the market requirements as well as the technological developments were and are a permanent challenge in our research. We evolve in a competitive environment where the designed technologies are permitting or hindering the project development. Under these auspices, in the section 5.4 we sketch the current vision on the PrivateSky, based on core ideas developed in SwarmESB. However, new extensions are needed in order to provide support beside developing applications requiring integration, systems with serverless architectures, systems that offer properties similar to blockchain systems (distributed ledgers for decentralised data storage, support for smart contracts in order to provide decentralised data processing) or to ensure integration between blockchains.
The current developments relying on our scientific publications in the choreographies field are on various levels according to NASA Technology Readiness Levels classification. In this moment we sustain continuous efforts of industrial research and experimental development in order to achieve the maximum maturity level. The ProjectSky project involves developing a stable platform while in parallel customised applications are created in accordance with companies and market requirements. One may properly conclude that the executable choreographies are a significant contribution to the progress of future Internet software development. Taking into account the outstanding potential in the area of personal data protection, the PrivateSky technology will enable the development of new types of integrated and interoperable software applications that can be used to solve major societal problems. Through its capacity to provide management in cloud, the PrivateSky platform will enable the development of IoT application or complex e-government or smart-city applications. It is worth nothing that research results in the security and private data protection will have significant applicability for the industry.

The thesis ends with a section where we succinctly present the fortuitous context in which the teaching activity merged with the research and academic management activities.