Glossary

**Adapter:** An adapter is a piece of software that enables interoperability between two disparate software components. By converting data or messages from one format to another, adapters can be used to bridge the communication gap that may exist between independently developed software components.

**Admission Control Algorithms:** Algorithms that enable Cloud providers to decide which client requests can be fulfilled without running out of computing resources and/or minimizing the penalty.

**Agile Software Development Process:** An evolutionary and iterative approach to software development with focuses on adaptation to changes.

**Analysis Process:** An analysis process in data-intensive science is the subset of a scientific process that is concerned with data analysis activities. A typical analysis process would involve activities to do with data manipulation or data transformation.

**API Description Language:** A markup language that enables developers to describe the functionality of the API and its operations to users/clients.

**API Discovery:** The process of identifying available and appropriate Web services by clients (service consumers), that is facilitated through automation.

**Architectural Drift:** Insensitivity to a system’s architecture that, with increasing evolution, leads to its in-adaptability and a lack of coherence and clarity of form.

**Architectural Erosion:** Violations of a system’s architecture that leads to significant problems in the system and contributes to its increasing brittleness.

**Architectural Style:** A set of constraints on the concepts of an architecture and their relationships.

**Architectural Style:** Reusable solution that defines the common architectural traits of a family of software systems by describing a structural organization.

**Architectural Violations:** Violations of a system’s architecture, usually an architectural drift or architectural erosion.

**Aspect-Oriented Programming:** Aspect-oriented programming (AOP) is a style of programming that attempts to abstract out features common to many parts of the code beyond simple functional modules and thereby improve the quality of software.

**Aspect Oriented for Business Process Execution Language:** Aspect Oriented for Business Process Execution Language (AO4BPEL) is an aspect-oriented extension to WS-BPEL that allows for modular and dynamically adaptable Web service compositions.

**Autonomic Computing:** A computing system characterized by one or more self-managing characteristics including self-configuration, self-healing, self-optimization, and self-protection.

**Autonomic Services:** Services that implement autonomic characteristics that involve involuntary actions by the service-driven system in response to the changing business and environmental condi-
tions are called autonomic services. Autonomic services are expected to satisfy one or more of the autonomic computing attributes such as self-healing, self-configuring, self-optimizing, and self-protecting. The term “autonomic computing” was motivated by the human autonomous nervous system that enables involuntary actions such as heart beats.

**Backup Service:** In the context of mobile task execution, the backup service which is run on a stationary device ensures that mobile tasks providing data for subsequent tasks will be executed properly in exceptional situations.

**Bag-of-Tasks:** Bag-of-tasks refers to the jobs that are parallel among which there are no dependencies. Jobs can be executed out of the submission order, such as video encoding/decoding, etc.

**Broker Reference Architecture:** A Cloud service broker assumes some common components, connections and interactions to be present to fulfill its tasks. A broker reference architecture defines some common components, connectors, and interactions along these connectors for a service broker.

**Business Intelligence:** Business Intelligence is an umbrella term that includes the applications, infrastructure and tools, and best practices that enable access to and analysis of information to improve and optimize decisions and performance.

**Business Process Execution Language:** Business Process Execution Language (BPEL and variants WS-BPEL, BPEL4WS) is a language used for composition, orchestration, and coordination of Web services. It provides a rich vocabulary for expressing the behavior of business processes. It uses an XML-based vocabulary to specify and describe business processes.

**Case-Based Reasoning:** A machine learning method consisting of solving a new problem according to previously solved similar problems.

**CEP:** Complex Event Processing allows multiple arbitrarily structured temporally related events to drive computation.

**Client-Server Architecture:** The client-server architecture is a distributed architecture that separates a service requestor (client) from a provider of service or resource (server).

**Cloud:** Infrastructure from which businesses and users are able to access services based on their requirements without regard to where the services are hosted or how they are delivered.

**Cloud Auto-Scaling:** Automatic resource provisioning mechanism to dynamically acquire and release computing capacity based on the workload, performance, and resource utilization indicators.

**Cloud Broker:** A Cloud broker is a service in a Cloud environment that mediates between service requestors and service providers. A Cloud broker facilitates initial match-making based on requirements and capability descriptions and also manages the dynamic integration.

**Cloud Bursting:** The application behavior that provides scalability according to demand, not just within the available single Cloud resources, but over multiple Clouds as well.

**Cloud Computing:** A computing paradigm wherein hardware, software, data, and network resources are provided as services to the consumers over high bandwidth networks on a pay-per-use or subscription basis.

**Cloud Computing:** A computing paradigm that facilitates the delivery of services over the Internet by means of Software as a Service, Platform as a Service, and Infrastructure as a Service.

**Cloud Computing:** A computing paradigm that provides on-demand, dynamic, and elastic provisioning of computational resources.

**Cloud Computing:** A paradigm for delivering IT services as computing utilities, using Internet enable interaction between information technology service providers and consumers.

**Cloud Computing:** Cloud computing enables convenient on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal service provider support.
**Cloud Computing:** It is a general term for anything that involves delivering hosted services over the network. These services are broadly divided into three categories: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).

**Cloud Computing:** Most researchers define Cloud computing as an innovative way of enhancing IT’s capacity to provide pay-as-you-go billing, on-demand computing, IT utility and automated delivery service. The National Institute of Standards and Technology (NIST, 2012) sees Cloud computing as a business model of enabling ease of on-demand access to a pool of shared services (e.g., data center, networks, servers, storage, virtualized application services) that can be provisioned dynamically with minimal support.

**Cloud Computing:** The elastic computing over virtualized resources, provided via pay-as-you-go model.

**Cloud Elasticity:** Refers to the ability to provision resources in the Cloud response to demand. It is one of the main advantages the Cloud offers to dynamically adapt to user workload changes.

**Cloud Host:** The virtual machine in the Cloud operated by CloudScale.

**Cloud Object:** An instance of a resource-demanding class that is distributed by CloudScale over the Cloud hosts.

**Cloud Service Brokerage:** Cloud service brokerage is the process of integrating, customizing, or aggregating Cloud services for service consumers. Widely accepted Cloud service brokerage definitions are provided by US standardization organisation NIST and the analyst Gartner.

**Cloud Service Template:** A Cloud service template is a service description that captures Cloud-specific characteristics in an abstract and instantiable format.

**COBOL:** The programming language designed for commercial business data processing used for applications that often form the backbone of the IT structure in many corporations since 1960.

**Compliance-by-Design:** Is a process of developing a software system that implements a business process in such a way that its ability to meet specific compliance requirements is ascertained. Formal methods are typically involved to automate compliance rule verification.

**Compliance:** An asymmetric property between a consumer C and a provider P (C is compliant with P) that indicates that C satisfies all the requirements of P in terms of accepting requests.

**Compliance:** Compliance is a process of measuring how well an organization or an entire industry functions to achieve specific high-level goals prescribed by law or regulatory documents.

**Compliance Fragment:** A connected process structure or a solution pattern that can be used as a reusable building block for ensuring a faster and more consistent specification and integration of a compliance solution into a process.

**Compliance Requirement:** Is a constraint or assertion derived from the interpretation of the compliance sources.

**Compliance Source:** Refers to a document that is the origin of compliance requirements.

**Compliance Target:** A generic specification, such as a business process, or a process fragment, to which compliance requirements are applied to.

**Component or Service Mobility:** An ability of a software system to move its components or services into different contexts and to different deployment nodes or service providers at the system’s runtime; a specific type of runtime reconfiguration.

**Component Technology:** Component technology refers to the use of software components for software development. Software components usually conform to a component model, and they are often hosted and managed by a component framework, which provides a controlled environment where components can be composed together to form larger applications or systems.

**Computing Platform:** A computing platform allows software to run on the computer and requires
a processor, operating system, and programming environment with related runtime libraries and user agents.

**Configuration of Architecture:** A particular way in which a system’s components or services and their connectors or bindings are composed and built into the resulting system.

**Conformance:** An asymmetric property between a provider P and a consumer C (P conforms to C) that indicates that P fulfills all the expectations of C in terms of the effect caused by its requests.

**Connector:** A conduit for messages between components.

**Consumer:** A role performed by a resource A in an interaction with another B, which involves making a request to B and typically waiting for a response.

**Context-Aware Services:** A technique which incorporates information about current situation to activate more relevant service.

**Context-Aware Services:** Services that have the ability to gather information about the environment in which they operate and use that information to change their observable behavior to adapt to the current situation. Context adaptation is generally implemented using a three phase architectural model that involves Context sensing, Context Modeling, and Context Reasoning.

**Context-Aware System:** A system reacting according to the current context of users and devices.

**Context Aware Computing:** Context Aware Computing is a contemporary trend, which accommodates the provisioning of software systems that are able to exploit contextual information, e.g., location, state, mood, in order to accommodate user requirements.

**Contract-Based Security:** A security framework that combines static analysis based on the information provided by the contract and run-time enforcement mechanisms in accordance with the policy in order to guarantee that a system is secure.

**Core Self-Organizing Services:** Low-level self-organizing services that provide core functionality such as spreading, aggregation, evaporation, or repulsion.

**Cost-Effectiveness:** Cost-effectiveness is the ability to experience an operational cost reduction. Cloud adoption achieves cost effectiveness by the use of on-demand IT services and shared multi-tenant resources.

**Data-Driven Adaptation:** The ability of a service-oriented process to use the information available within its environment and adapt its execution accordingly.

**Data-Intensive Science:** Data-intensive science is considered to be the fourth paradigm of science after the three interrelated paradigms of empirical, theoretical, and computational science. It is seen as a data-driven, exploration-centered style of science, where IT infrastructures and software tools are heavily used to help scientists manage, analyze, and share data.

**Data Object:** The object that is used to transfer data between classes or application components.

**Data Warehouse:** During the integration of data from multiple distributed heterogeneous databases and other information sources, a data warehouse is used as the repository to store the data.

**Delegation Service:** In the context of mobile task execution, the delegation service is able to delegate task execution to another mobile user at the occurrence of errors. That delegation will be performed automatically without allowing users to decline it.

**Distributed Process Execution:** The execution of process models in a distributed manner that moves beyond clusters and other centralized approaches.
**Distributed System**: A system of computers connected through a network and distribution middleware that coordinates the activities of the computers and shares their system resources to represent a single integrated computing facility.

**Dynamic (Software) Architecture**: Software architecture of a software system with rules of evolution of its structure/architecture during runtime. The system’s components and connections can be created and destroyed during runtime according to the rules from design-time.

**Dynamic Service Composition**: The process of assembling individual services at runtime to compose a dynamically adaptive composite service.

**Dynamic Software Adaptation**: A characteristic of software systems by which components can be added, removed, replaced, or composed at runtime.

**EDA**: An Event Driven Architecture is driven by events that trigger behavior in software components.

**Elasticity**: Ability to dynamically deploy and scale systems resources based on need in real time. This provides an illusion of infinite amount of resources to consumers allowing the consumer to purchase only what they need.

**Entailment Constraint**: An entailment constraint defines a dependency between tasks. As examples of such constraints, consider separation of duties (i.e., two particular tasks of a process instance must be executed by different users) and binding of duties (i.e., two particular tasks of a process instance must be executed by the same user).

**Enterprise Service Bus (ESB)**: A software architecture model used for designing and implementing the interaction and communication between mutually interacting software applications in service oriented environments.

**Environmental Processes**: Process models addressing environmental science problems normally comprising spatially related types of services to retrieve geospatial information.

**EOP**: Exertion-oriented programming (EOP) is a programming paradigm that represents concepts as “services proxies”, called exertions, that have hierarchically organized data contexts (attributes that describe the service data), control contexts (attributes that describe the service control strategy), and signatures to associated service providers. The associated providers collaborate according to control strategies imposed by control contexts to process data contexts of the service federation created at runtime.

**eScience**: Technologies that process data by software and store the resulting information or knowledge in computers.

**Exertion**: An exertion is a service-oriented process expression functioning as a proxy to a corresponding service federation created at runtime. A task exertion (or simply a task) is an elementary service provided by a single service provider. A job exertion (or simply a job) is a service composition that represents a hierarchically organized collaborative service federation (workflow). A block exertion is a concatenation of exertions having common block scope for control flow. An exertlet is a corresponding object proxy for the exertion. A netlet is a corresponding interpreted script for the exertion.

**Experiment Workflow**: Coordination of the service composition process so that experimenters can compose media-centric services according to the control and data dependencies.

**Federated Cloud Architecture**: A federated Cloud architecture assumes that individual components of the traditional Cloud stack are distributed across possibly independently controlled nodes.
**Genetic Algorithm**: The approximation algorithm based on the evolutional process.

**Handler**: A higher-order feature that is used to produce behavior in response to receiving a message in a component.

**HATEOAS (Hypermedia as the Engine of Application State)**: A fundamental constraint employed in the REST architectural style, whereby client applications determine their state and state transitions entirely based on the dynamic hypermedia information provided by the servers.

**Heterogeneity**: Variations in offered capabilities, communication protocols, operating systems, as well as the descriptive languages of network services.

**Heterogeneous Services**: Heterogeneous services comprise the contemporary instantiations of the service-oriented model, e.g., Web services, WSRF services, Peer-to-Peer services, Open Geospatial Consortium services, etc.

**Higher-Order**: Is a relative property of a language feature that lifts restrictions on its use compared to other features. Normally it is used to allow higher-order language features to be treated as data.

**Hybrid Cloud**: The hybrid Cloud is a combination of private, community, and public Clouds used by organizations that host non critical data on public Cloud that remain as unique entities but connected by public or proprietary middleware to enable application portability.

**IaaS**: Infrastructure as a service level of Cloud computing.

**Imperfect**: Debugging: The phenomenon of new faults being generated as a result of a fault fix.

**Information Security**: Information security consists of rules for maintaining confidentiality, integrity and availability of data and electronic transactions. Information security is always associated with prevention and detection of unauthorized activities in the networks or systems.

**Infrastructure as a Service (IaaS)**: Infrastructure as a service is defined as the ability to commoditize IT infrastructure resources as services to organizations on demand. It provides organizations with the ability to re-direct resources to long-term strategic business goals while outsourcing the day-to-day IT functions, such as archiving and business continuity to Cloud providers.

**Integration**: Integration describes the idea of applying a consistent user experience and complementary functionality across multiple software components to produce an “integrated” environment. Within an integrated environment, users can save on cognitive resources by learning a single model of behavior and applying the learned model to understand and operate all components within the environment.

**Intelligent Agent**: An autonomous entity which acts upon the circumstances and direct its activities towards achieving goals.

**Interoperability**: An asymmetric property between a consumer C and a provider P (C is compatible with P) that holds if C is compliant with P and P is conformant to C.
Interoperability: Interoperability refers to the ability of different software components to communicate and cooperate with one another, by resolving differences in language, execution platform, interface definition, and in the meaning of what has been communicated.

IT Compliance: IT compliance consists of meeting the policies and procedures mandated by governmental and organizational requirements with expectation of penalties for violation.

Jump Diffusion Model: The stochastic differential equation model with the jump term.

Legacy Integration: The integration and Web extension of existing (legacy) systems, especially mission-critical mainframe systems, in order to leverage existing IT assets.

Mainframe: Mainframe computer systems like IBM z/OS.

MDA: A software development approach where formal models are used to represent the system at different levels of abstraction, and model-to-model and model-to-code transformations are executed by automated or semi-automated transformation tools.

Media-Centric Service: A software function that focuses on media contents processing and delivery such as producing, transporting, and displaying.

Message: The key mechanism by which behavior is triggered in component-based systems.

Metamodel: A metamodel is a model used to describe the constructs or elements of another model.

Method of Maximum Likelihood: The estimation method of maximizing the likelihood function defined as the probability for observing the actual data.

Middleware: Middleware is software that is used to facilitate and link interactions between various software components across distributed computing environments. Middleware typically utilizes adapters to enable interoperability between wide varieties of components. Also, middleware often provides higher-level and more user-friendly access to the components it manages.

Mobile (Software) Architecture: A dynamic architecture of software with component or service mobility features.

Mobile Device: A handheld device that offers mobile application functionality to its user and can host Web services.

Mobile Process: A process that contains tasks executed on mobile devices.

Model-Driven Engineering: A software development methodology that focuses on creating and exploiting domain models rather than on the computing or algorithmic concepts.

Model Transformation: A process of converting the source model into target model based on a specific set of rules.

Mogramming: Mogramming is programming or modeling or both.

Multi-Agent System: A set of communicating and collaborating agents where each agent senses its environment changes and reacts accordingly.

Neural Network: The machine learning technique using the input-output rules learned from the data sets.

Non-Homogeneous Poisson Process (NHPP): The stochastic process model for describing the stochastic behavior of the fault-counting process.

Ontology: A description of a concept with a common understanding in a particular domain.

Ontology: Set of representational primitives consisting of machine encoding of terms, concepts, and relations among them to model a domain of knowledge.

Ontology: Specification of a conceptualization or a formal description of concept in a domain.

Open Source Software: The software based on the concept of the open source code where source code is non-proprietary and freely licensed.

PaaS: Platform as a service level of Cloud computing.

Performability: The rate of operation of software system.

Perturbation: Involves altering the internal state of an application without modifying its source code. Perturbation is a technique used in testing service-driven systems where request messages
are modified and resent to the Web service to analyze the change in the response message with respect to the original response message.

**Petri Nets:** A formal and graphical technique for the specification and analysis of concurrent, discrete-event-based dynamic systems.

**Platform as a Service (PaaS):** A level of the Cloud-service model in which the user requests a computing platform without caring about the machines and network on which it stands on.

**Platform as a Service:** Platform as a Service (PaaS) is the extension of IaaS to provide a development environment as a framework.

**Platform as Service (PaaS):** Platform as a service is an application development solution offered by Cloud providers that includes operating systems and development tools that IT staff can use to manage infrastructure applications.

**Policy:** A policy is a security requirement that a consumer wants to apply to a service execution.

**Portal Technology:** Portal technology refers to the use of centralized portals to expose underlying functionality of a system. A portal acts as a doorway, allowing the underlying functionality (which is often quite complex) to be presented and used in a much more user-friendly manner. A portal also provides a single point of entry so that it is easier for users to find what they are looking for.

**Private Cloud:** Private Cloud is a proprietary Cloud infrastructure that is wholly operated for an organization to offer secure computing services either on the organization’s premises or in a third-party vendor offsite location.

**Prolog:** A logical programming language used in artificial intelligence for building of expert systems.

**Provider:** A role performed by a resource B in an interaction with another A, which involves waiting for a request from A, honoring it, and typically sending a response to A.

**Public Cloud:** Public Clouds are network infrastructure designed to provide software access through Web based portals to general public for access to hardware, networking, and software.

The Cloud infrastructure is shared by multiple organizations and offer specific non-customizable services.

**Public Cloud Infrastructure:** Cloud infrastructure pertaining to a different institution from that using the services. Regularly, these services are charged by usage time.

**QoS (Quality of Service):** Constraints placed on Web services that describe the minimum service levels (performance, security, scale) that the provider can guarantee.

**Reconfiguration of Architecture:** A modification of the configuration of a system’s architecture.

**Reification:** Is a process of moving static or otherwise inaccessible information into user-supplied dynamic data and using it to affect the execution of a system.

**Representational State Transfer (REST):** An architecture style for designing networked applications. Rather than using complex mechanisms such as CORBA, RPC, or SOAP to connect between machines, simple HTTP is used to make calls between machines. RESTful applications use HTTP requests to post data (create and/or update), read data (e.g., make queries), and delete data, thus using HTTP for all four CRUD (Create/Read/Update/Delete) operations.

**Reproducibility:** Full replication refers to independently repeating a piece of research from scratch in order to obtain the same results. Where replication is not possible or impractical, research artifacts (e.g., data, process records, and code) can be made available as a means to reproduce the same results when the artifacts are analyzed again. Hence, reproducibility of scientific results offers a stepping stone towards full replication, where the more artifacts are made available, the closer we can get towards full replication.

**Resource:** A logical or physical entity required for the running of a service.

**Resource:** An entity of any nature (material, virtual, conceptual, noun, action, and so on) that embodies a meaningful, complete, and discrete
concept, makes sense by itself, and can be distinguished from although able to interact with other entities.

**Resource Awareness:** The capability that has more knowledge about the relationship between the amount of resources and the quality of service composition.

**Resources:** The key data and code abstractions that serve as the basis of all RESTful Web service interactions.

**Resource Scaling:** In the Cloud resource provisioning process, resource scaling refers to the amount and type of resources that need to be acquired.

**Resource Scheduling:** Resource scheduling refers to the second step in the resource provisioning process, i.e., allocating the acquired resources to submitted jobs. The two steps, resource scaling and resource scheduling are dependent on each other.

**REST:** An architectural style for implementing distributed systems based on stateless design, client-server interaction model, layering, uniform interfaces, and hypermedia driven state transitions.

**REST Services:** Representation State Transfer (REST) is an architectural style to define Web-based services using a set of HTTP operations GET, POST, PUT, and DELETE. It is highly flexible and offers the use of free data structures.

**Runtime Reconfiguration of Architecture:** An ability of a software system to perform reconfiguration of its architecture at runtime, e.g., to create, destroy, and update the services, and to establish and destroy their interconnections dynamically at the runtime, on demand, and according to various aspects to move the services into different contexts and to different providers.

**Runtime Testing:** The ability of a system or component to execute tests while operating in a production environment.

**Scientific Process:** A scientific process in data-intensive science consists of the sequence of activities performed by scientists in order to extract knowledge from data. A typical scientific process would cover a wide range of activities including data acquisition, data manipulation, and publication of analysis results.

**Secure Web Service:** A Web service whose behavior does not violate the security policy.

**Security-by-Contract-with-Trust:** A security methodology that substitutes the evidence checking functionality of security-by-contract with a trust management mechanism, and thus removes the assumption that verification may be performed and relies on how much a user trusts the correctness of the published contract.

**Security-by-Contract:** A security methodology that implements automatic checking of the formal correspondence between the code and the contract, i.e., provide evidence checking.

**Self-Adaptive Services:** Services that reconfigure themselves in response to changes in business requirements and operating environments. They reconfigure by running variants of the initial configuration through substitution of services to adapt to the changing ecosystem.

**Self-Composition of Services:** Spontaneous composition of services, either through chaining or parallel execution of services, arising in response to a query for a value or for a computation.

**Self-Managing Services:** Services that through the dynamic adaptation of parameters or the switching of algorithms ensure non-functional requirements, such as bandwidth optimization, accuracy, or availability.

**Self-Organizing Mechanisms:** Rules that agents employ to coordinate their behavior, usually following information gathered from their local environment. Typical mechanisms inspired by nature are spreading, stigmergy, digital pheromone, evaporation, gossip, etc.

**Self-Organizing Services:** Services which implement self-organizing mechanisms that the higher-level applications can reuse.

**Semantically Enabled Service-oriented Architectures (SESA):** A comprehensive frame-
work that augments existing SOA frameworks to incorporate semantic solutions to address the SOA semantic gap.

**Semantic Discovery:** Attempts to use semantics to meet user requirements against offered services.

**Semantic Mashups:** Mashups, which are composites based on RESTful services, and which have their APIs annotated by a semantic layer for dynamic selection and composition.

**Semantic Tuplespace:** Refers to an extended version of the Linda model, where information is annotated with appropriate semantics and provided API is enhanced so as to accommodate and exploit the underlying semantics.

**Semantic Web:** A framework based on the Resource Description Framework (RDF) allowing data to be shared and reused across application, enterprise, and community boundaries.

**Semantic Web Services (SWS):** A technology that combines Web services and Semantic Web technologies in order to allow for a higher level of automation when dealing with Web services.

**Semantic Web Services:** Web services annotated with semantics. Semantic Web services extend the existing Web services with semantic capability.

**Service-Driven Approach:** Architectural approach based on software services.

**Service-Driven Computing:** Computing methodology for implementing enterprise applications and integration solutions to support the dynamics of the organization by composing business and scientific services from granular application services through the mechanisms of Service Mediation and Service Composition using architectural practices such as SOA and REST.

**Service-Oriented Architecture (SOA):** A logical way of designing software architectures to provide composed services via published and discoverable interfaces.

**Service-Oriented Architecture:** An architectural model for building software applications that use services available in a network such as the Web.

**Service-Oriented Architectures (SOA):** A methodology for reorganizing and building software applications and infrastructure into a set of meaningful interacting services.

**Service-Oriented Process:** Process models implemented in terms of services.

**Service-Oriented Testbed:** Indicates the testbed to support diverse media-centric service composition experiments with programmable and virtualized resources having computing and networking capabilities.

**Service:** A distinct logic of unit addressing a specific business concern.

**Service:** A service is the work performed in which a service provider (one that serves) exerts acquired abilities to execute a computation.

**Service:** Implementation of well-defined business functionality.

**Service:** The set of operations supported by a resource and that together define its behavior (the set of reactions to messages that the resource exhibits).

**Service Adaptation:** A behavioral change of a service in order to become better suited to the user.

**Service Availability:** Represents the ability of services to be accessible as needed, whenever and wherever they are required.

**Service Composition:** It is the process of synthesizing a new composite service using a set of available Web services in order to satisfy a user request that cannot be satisfied by any individual Web services.

**Service Composition:** The assembly of distributed composable (component) services into a composite service to complete the desired application.
**Service Composition Process:** Represents the execution order of relevant tasks to integrate composable tasks such as discovery, matchmaking, placement, stitching, monitoring, and tuning.

**Service Contract:** A service contract is a formal description of the service behavior. Contracts typically describe the service in terms of interaction protocols such as input and output channels, message syntax, parameter types, encryption algorithms, etc.; and service computation such as message semantics, service state transitions, and resources usage.

**Service Delivery Platform:** A software platform which provides automatic support for service related tasks including discovery, ranking, invocation, and monitoring.

**Service Description:** A service description captures the functional and non-functional characteristics of a service in a processable format.

**Service Discovery:** Locating services hosted by mobile devices that are available for immediate use.

**Service Federation:** A service federation is a runtime collection of all service providers needed to actualize service collaboration expressed by an exertion.

**Service Level Agreement (SLA):** A formal definition of the expected quality of a service that will be delivered, performance and deadline are typical elements used when defining SLAs.

**Service Ontology:** A service ontology is a knowledge representation format capturing service information in a rich conceptual modeling framework that enables description, manipulation, and reasoning.

**Service Oriented Architecture (SOA):** A technical software architecture that allows client applications to request services from service provider type applications in a host system.

**Service Oriented Architecture (SOA):** The architectural paradigm in which new applications are developed by reusing and combining existing Web services.

**Service Oriented Architecture:** A technological approach and methodology for building business applications. These business applications implement business processes or services through a set of loosely coupled, black-box components orchestrated to deliver a well-defined level of service.

**Service Proxy:** A service proxy, in its most general form, is a service functioning as an interface to another service. The proxy could interface to anything: a network connection, a large object in memory, a file, or a service provider that is expensive or impossible to duplicate like a server or federation of servers.

**Service Reusability:** It is the degree to which the service can be used in more than one business process or service applications without the need for reconfiguration or recreation.

**Services Adaptation:** Refers to the capability of the software service to adapt its behavior in order to satisfy new requirements and to fit new situations dictated by the environment.

**Service Technology:** Service technology refers to the use of services for software development, where a service is an autonomous, platform agnostic software component that operates within an ecosystem of services. The ecosystem is governed by a service-oriented architecture, which relies on the composition of loosely coupled services to achieve complex functionality.

**Simple Object Access Protocol:** An XML-based communication protocol to access the Web services over the network using transport protocol such as HTTP. SOAP is language and platform independent.

**SOA:** A Service-oriented Architecture (SOA) is a software architecture using loosely coupled service providers that introduces a service registry, the third component to client-server architecture. The registry allows finding service providers in the network.
SOA: A Service Oriented Architecture requires software components to publish logically coherent groups of system functionality.

SOA: Service Oriented Architecture is an architectural style that promotes integration of services available in a network to build enterprise applications.

SOAP: A set of standards that define an architecture for developing, hosting, and managing Web services based on remote procedure calls for computation and messages encoded using XML for communication.

Software Agent: A piece of software capable of performing the specified tasks on behalf of the user.

Software as a Service (SaaS): A level of the Cloud-service model in which the user requests the use of a given application without caring about machines, networks, or software dependencies.

Software as a Service (SaaS): Software as a service is an application delivery concept that provides pay-as-you go pricing structure for software licenses and access for users instead of traditional installation of software on local computers or through client server network model.

Software Component: A software unit of functionality that manages a single abstraction.

Software Component: Modern software applications and systems are most often developed as assemblies of many smaller parts. The idea of software components formalizes the definition of these “smaller parts”: A software component is basically a software unit with a well-defined interface and explicitly specified dependencies. A software component can be as small as a block of reusable code, or it can be as big as an entire application.

Software Reliability: The probability that a particular software will not cause the failure of a system for a specified time under specified conditions as detailed in IEEE Standard 610.12, 1990.

Software Reliability Growth Model (SRGM): The stochastic model for measuring and assessing the reliability of a software system.

Software Testing: A form of software validation that is most commonly used in the software industry. It is the process of operating software under specified conditions, observing or recording the result, and making an evaluation of some aspect of the software.

Software Validation: The process of ensuring that a software system meets the needs and expectations of its clients and/or customers.

SOOA: A Service-object-oriented Architecture is SOA with the communication based on remote message passing with the ability to pass data using any wire protocol that can be chosen by a remote object (provider) to satisfy efficient communication with its requestors. In SOOA a proxy object used by the requestor, for example in Jini ERI, is created, registered, and owned by the provider.

SORCER: A Service-ORiented Comput-ing EnviRonment (SORCER) is a SOOA based computing platform, which provides federated service-oriented environment within its operating system that runs front-end services (exertions) and dynamically manages corresponding federations of local and remote service providers.

Spatial Services: Services that exploit or spread data spatially distributed over stationary or mobile devices in a given geographic area. Spatial services use self-organizing services to retrieve, spread, or aggregate data.

Spatial Structure: Geographically situated structure made of data disseminated in mobile or stationary devices. A typical example is given by a gradient disseminated across multiple devices.

SPOA: A Service-protocol-oriented Architecture is SOA with a wire protocol between a service proxy used by the requestor and its service provider that is predefined and final, for example IIOP, JRMP, or SOAP protocol.

Static (Software) Architecture: A software architecture without ability to be modified during
runtime. After initialization of the system, there are no new connections between the system’s components and existing connections cannot be destroyed.

**Static Service Composition:** The process of assembling individual services at design time to compose a static composite service.

**Subsumption:** It is a hyponym-hypernym relationship.

**Target Application:** The application distributed by CloudScale.

**Technology Adoption:** Adoption of technology theory examines individuals or business leaders’ decisions to accept or not to accept a given technology for integration of that technology into the entire organization.

**Template Manipulation Calculus:** Templates are specifications that need to be manipulated to allow their adaptation to specific situations. The manipulation operators include single template changes such as renaming, restriction, and refinement, and combination operators such as union, intersection, and composition.

**The Semantic Web:** A Web of data that can be processed by machines.

**Traceability:** is a property that represents the ability to trace a compliance requirement throughout the lifecycle of a business process or its artifacts.

**Tuple Space:** Blackboard container of data provided in the form of tuples.

**Ubiquitous Computing:** A new domain of computing where the user is surrounded by several networked devices that provide services continuously.

**Universal Description, Discovery, and Integration:** A Web-based distributed directory that enables businesses to list themselves on the Internet and discover each other, similar to a traditional phone book’s yellow and white pages.

**User List Task:** The user list task constitutes an element of the delegation service. To be more precise, this task will be used to compute the user lists of the mobile task and its delegation tasks before they will be activated.

**Utility Computing:** A metered service on remote servers paid on demand. It allows users to lease access to a virtualized resource from Cloud service providers that places the users’ data under the control of Cloud providers.

**Validation Task:** The validation task constitutes another element of the delegation service. It allows confirming the execution of the backup task, i.e., the backup task may only be confirmed if it provides same information as the failed mobile task. Note that the mobile user responsible for the failed (mobile) task is also responsible for confirming the task.

**Violation:** A violation happens when a service behaves in a way that is not allowed by the policies.

**Virtualization:** A technology that provides the basis for Cloud Computing where virtual resources such as operating systems, application servers, data stores, and message queues are provisioned and de-provisioned dynamically.

**VM Startup Time:** VM startup time measures the latency between the times the user initializes the VM acquisition request until the time a VM is ready to use. The VM startup time varies by VM type, OS image, data center locations, and Cloud providers. It is an important factor that Cloud resource provisioning mechanisms need to consider, especially for time critical applications.

**Web API:** The network-accessible interface of a Web service through which remote users can interact with the service.

**Web Ontology Language for Services:** A high level language (XML-based) used for describing Web service properties. It consists of three parts: service profile, process model, and grounding. Service profile includes general information and is used to describe what the service will do. A process model describes how the service will perform its functionally and the grounding
describes links with industry standards. Its main goal is to enable users to automatically discover, invoke, compose, and execute Web services under certain conditions.

**Web Service:** A computational entity that is accessible over the Internet, using Web service standards and protocols.

**Web Service:** A modular application that can be described, located, and invoked on the Internet. A Web service is designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine processable format, specifically WSDL.

**Web Service:** A service available on the Web that is platform-independent, self-describing, discoverable, and accessible using a standard communications framework.

**Web Service:** A software component with a well defined interface that can be accessed over a network and which supports interoperable machine-to-machine interaction using open standards and protocols.

**Web Service:** A software entity provided over the Web that can expose a number of functionalities.

**Web Service:** A software system identified by a URL and whose public interfaces and bindings are defined and described using XML is called a Web service. Its architecture is loosely coupled. The Web Service Description Language WSDL uses the XML format to describe the methods provided by a Web service, including input and output parameters, data types and the transport protocol, which is typically HTTP. Web Services use a registry to publish details about services and provide an opportunity for service consumers to find service providers.

**Web Service:** URL-addressable set of functions exposed over a network to serve as a building block for creating distributed applications.

**Web Service Composition:** Combination and coordination of a set of services with the purpose of achieving functionality that cannot be realized through existing services.

**Web Service Description Language (WSDL):** An XML format for describing network services as a set of endpoints operating on messages containing either document oriented or procedure oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint.

**Web Service Description Language:** An XML-formatted language used to describe a Web service’s capabilities as collections of communication endpoints capable of exchanging messages. It provides a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns. It thus serves a purpose that corresponds roughly to that of a method signature in a programming language.

**Web Service Modeling Ontology (WSMO):** A conceptual model which defines the basic concepts of Semantic Web Services.

**Web Service Modeling Ontology Lite (WSMO-Lite):** A lightweight service ontology inspired by WSMO, which builds on the newest W3C standards and allows bottom-up, lightweight semantic annotation of services.

**Workflow:** A workflow is composed of connected tasks, which is normally described as a directed acyclic graph (DAG). Tasks belonging to the same workflow need to follow the precedence execution constraints.

**Workflow Technology:** Workflow technology aims to automate common scientific activities as steps in a workflow model, which explicitly defines how a sequence of activities (the work) “flows” from one to the next. A workflow application is often used to “run” or execute workflow models. The application also manages the lifecycle of activities in the workflow model and interactions with data and other resources required to complete the activities.

**WSDL:** An XML-based language used for describing Web services.