Chapter 1
Introduction

An organization refers to structuring of resources to produce and provide goods and/or services. The basic goal of an organization is to produce goods and services and make them available. Entire organization is viewed as a collection of systems. Hospitals, banks, government agencies and online book stores are some examples of organizations. Each organization comprises of a set of systems such as production, marketing, sales, human resources, accounts and stores. Each system is divided into major processes. These processes are subdivided into smaller processes. For example production might be broken into processes such as manufacturing of the product, verification, packaging and shipment of the product. Processes are essential to understand how system operates. Processes play an important role in the design and realization of systems. For successful operation of an organization, these processes must be integrated and work in coordination.

A business process consists of a set of activities to provide a specified service. An Activity is automated action that indicates what is to be done at a particular step in the process. For example, consider the book procurement process of a book store, where a customer needs to order book(s) from a book store (provider). The service provider processes the order and delivers book(s) to the customer. Once the order is processed, payment is made by the customer using credit card or any other mode of payment. To carry out this process, various activities such as purchase order receiving, book availability checking, credit card validation checking, payment and delivery of books are involved.
With the advancement of Internet, organizations express their functionalities in terms of services provided by them. A service requires individual and autonomous unit of activity to be performed. The size and scope of the functionality represented by a service varies. A process is composed of one or more services. Services are performed by a single organization or may interact with services performed by other organizations. For example, book availability checking is performed by book store, while credit card validation is conducted by a credit card company.

Business to business (B2B) interaction provides the connectivity and aggregation of organizations. Business processes interact with each other. For instance, the business processes of a reseller interact with buyer processes. In our book purchase example, if book is not available then book store will order books from resellers as follows.

- The book store sends order to the book reseller.
- The book reseller receives order and starts order processing.
- The book reseller sends an invoice.
- The book store receives the invoice and sends payments. Finally, the book store receives the ordered books.
To carry out this process, various parties such as customers, sellers, suppliers, transporters, couriers and various services are involved. The state of the services should be maintained in order to provide the status of a process. For example, customer can check the status of the purchase order and add or delete an item. The overall process also includes various types of policies, service level agreements and events among partners.

Organizations are becoming more collaborative, distributed, and heterogeneous with the advancement of Information and Communication Technology (ICT). As a result, business processes require integration of distributed heterogeneous services, customers, and providers. When more than one service is required, multiple services can be combined as a single composite service [1]. Business processes are event-driven [2] and events affect the execution of a business process. An event is the specification of a significant occurrence that has a location in time and space. An event changes the state of a service during its execution. Business processes are dynamic in nature because interaction policies, strategies, services, providers and consumers change at a runtime. Identification, selection and composition of services at the runtime of a business process are defined as dynamic composition of a process. Complete execution of a process requires dynamic composition and integration of heterogeneous services according to events. Long running and dynamic business processes require support for state monitoring, transaction management, negotiation, and event notifications. Transaction over a composite service is represented as a set of long running business activities. Transaction management is required to coordinate interactions between consumers and providers, to achieve atomicity and to resolve the conflicts occurring during the execution of a business process. Semantic interoperability among services is desirable when a process spans across the boundaries of multiple business organizations, where vocabulary is different. Because of heterogeneity, dynamic nature, and lack of common vocabulary among business partners, a scalable software architecture is essential. Such an architecture should be capable of providing seamless interoperable integration, automation, execution monitoring, state, transaction, and notification management.

This Thesis aims to focus on dynamic nature of business process and propose event-driven service-oriented architecture (EDSOA) based on the convergence of Web services, Semantic
web, and grid computing. Web Services\(^1\) (WS) fulfills the functionality of a business process by integrating distributed heterogeneous services. Dynamic nature of a business process is modeled by capturing events using event calculus and composition schema is generated dynamically by using Event-Condition-Action (ECA) rules. Semantic web provides required support for common vocabulary, interoperability, and automation. Grid provides middleware support for execution of a business process with state, transaction, notification, and life-cycle management of a process.

This chapter is organized as follows. In Section 1.1, outline of Service-oriented computing is given. Section 1.2 provides the overview of service composition, its phases and approaches. Section 1.3 outlines the research issues and challenges of services composition. Section 1.4 presents the proposed approach to resolve the issues of composition and list the contributions of this research work. In Section 1.5 the structure of the thesis is given.

### 1.1 Service-Oriented Computing

The idea behind the Service-oriented Computing (SOC) is to provide platform and language independent development of software components and applications. The development and acceptance of open standard, framework and languages are the key concepts behind the success of SOC. Services are the key component elements to provide loosely coupled service centric computing. Emerging service-oriented concepts and technologies like Web services, grid computing and Semantic web are seen as efficient and relevant paradigms to integrate computational resources together to provide access to information and processing capability anytime anywhere [3].

Web services has emerged as the next generation of web technology to publish, discover, and invoke the software component as services. It is standardized by World Wide Web Consortium (W3C). As per W3C, Web services is described as: “a software application identified by a Universal Resource Identifier (URI), whose interfaces and bindings are capable of being defined, described, and discovered as XML artifacts. A Web service supports direct interactions with other software agents using XML-based messages exchanged via Internet-based protocols” [4]. It

\(^1\) In this thesis, the term Web services is referred as a singular term.
provides loosely coupled, interoperable integration of web hosted services and access to wide range of computing devices. Web services is based on three core protocols: Web Services Description Language (WSDL) [5] to describe the service, Simple Object Access Protocol (SOAP) [6] is a communication protocol to access the service, and Universal Description, Discovery, and Integration (UDDI) [7] to register and discover services from the registry. Web services and Service-Oriented Architecture (SOA) are promising paradigms for development of enterprise applications [8]. SOA is software architecture to enable loosely coupled integration and interoperation of distributed heterogeneous systems by using services as component elements.

Grid computing becomes known for wide-area scientific and enterprise applications. It allows runtime selection, integration, and coordination of distributed resources to accommodate dynamic business requirements [9]. It gives scalability and flexibility by following open standards, protocols, and technology like Web services. Modern grid [10] is based on Open Grid services Architecture (OGSA) and Web services Resource Framework (WSRF) [11]. WSRF bridges the gap between Grid services and Web services. This new development makes grid suitable for various kinds of applications like collaborative computing, ubiquitous computing, multimedia applications, and enterprise applications [8, 12, 13].

Semantic web comes as an answer to provide semantic interoperability and automated machine processable system. It is as an extension of current web to provide well defined meaning to the information [14]. It uses ontology to define the concepts of a domain of interest. Semantic approach helps in search, discovery, selection, composition and integration of Web services and also in automation of invocation, composition and execution of services. Service-oriented computing utilizes the Semantic web to provide interoperability and automation benefits to enterprise applications.

1.2 Service Composition

Service composition is a very important service-oriented principle and is known as service assemblies. Composition provides new functionalities by creating new services from existing and distributed services. From a business perspective, business service represents distinct business logic. Service composition is comprised of a set of independent business services. Service itself
composes other services and can call other services to accomplish its work. Therefore, each service that participates in composition performs its individual role. Service-orientated principles promote composability. Therefore, services should be designed by keeping composition in mind, so that they can be used for future service compositions.

Service composition contains various phases and it is a complex and challenging task. It is rapidly gaining attention and many solutions and different approaches have been proposed.

1.2.1 Phases of Composition

Following phases are involved in development of services and service composition.

**Service Creation:** First of all, the service provider should create the service and publish the interface with the description of a service. The description should provide the information about functional and non-functional requirements of a service. It should provide information about methods, inputs, outputs, exceptions, data types and transport mechanisms. Non-functional parameters are related to quality of service, response time, and cost. These parameters are required to evaluate the service.

**Service Discovery:** Provider should advertise or publish the capabilities of each service, so that later on service consumers can discover any of these services as per their requirements and then communicate with service provider to access discovered services. As number of services increase, discovery and selection becomes complex and essential phase. Directories and registries should be able to manage different versions of services and help to discover the services based on the specified criteria.

**Service Composition:** When requirement of a requester is not satisfied by a single service, composite service can be created with a set of atomic services. The functionality of atomic services combines by defining the control and data flow and mapping input and output of services. Each service should be designed with proper granularity, so that it can participate in composition.

**Service Selection:** Similar or identical functionality may be provided by many service providers. A huge set of services, providing identical functionality, may exist. The best available services
should be discovered and selected to generate a composite service. Selection is influenced by quality, functional and non-functional requirements of the service.

**Execution and Monitoring of Composite Service:** After the selection of the best composite service, deploy the components and services and configure the middleware for the execution. Services will be executed in the sequence as specified in the composition schema by passing messages and data from output of a service as input to the next service. Monitoring is required to keep the track of execution of a process, usage of services and exception handling.

### 1.2.2 Approaches of Services composition

Service composition approaches can be classified in two categories: time based and human intervention based. Based on the time, it can be categorized into two types: static (design time) composition and dynamic (run time) composition [15-17]. Human intervention based composition can be categorized into two types: manual (human driven) and automated (machine driven) [15].

**Static composition** is an approach, where business processes, business partners and services are known at design time and do not change frequently. Application designer will manually generate the composition schema by selecting and integrating services at design time. Static composition is useful to provide complex interaction pattern among known components. Leading commercial products such as Oracle BPEL Process Manager, IBM WebSphere Business Modeler, BEA Weblogic, Microsoft Biztalk are supporting static composition.

**Dynamic composition** is an approach, where business partners, consumers, and services are changing at runtime. New and better services may become available and partner policies are likely to change dynamically. Business process should be flexible and adaptable and should provide service selection based on user requirements and context. The service composition schema is generated dynamically and it does not require human intervention for composition. Therefore, dynamic service composition is useful for applications where components, services and users are dynamic, such as mobile computing, grid computing and ubiquitous computing. SWORD [18], eFlow [19], and StarWSCoP [20] are few examples of dynamic service composition systems.
Manual composition is an approach, where designer (human) can manually design or model the workflow and interactions among components for generating the composition schema. Business Process Execution Language (BPEL) [21] is an example of manual composition approach. It is a lower level process modeling and execution language where designer design the flow using control constructs like if-then, switch case, fork, while-loops etc. It is a widely accepted standard for manual composition of services.

Automated composition is a semantic based approach, where it processes the data and generates the composition schema. Semantic web gives well defined meaning of information and makes it machine processable. It uses ontology to provide description of functional and non-functional properties of services, to carry out automated discovery, selection and composition. Several research efforts both in academia and in industry have proposed a number of automated planning and semantic based techniques for automated composition (SHOP2 [22], Medjahed et al. [23], Berardi et al. [24]). OWL-S [25] and WSDL-S [26] are known standards to provide semantic in Web services composition.

Model driven service composition is Unified Modeling Language (UML) [27] based approach. Due to lack of dynamic and adaptive composition support in BPEL, model driven approach is proposed and it is similar to business rule driven service composition. It uses UML to provide higher level of abstraction and Object Constraint Language (OCL) based business rules to describe process flow. This approach can be static or dynamic. Orriens et al [28] and Zhang et. al [29] have introduced model driven dynamic web services composition.

Adaptive service composition is a dynamic composition approach, where composite services need to be adaptive. It should be capable to adjust according to the changes in the environment, requirements and the context of a user. It is an advanced service composition approach, where service selection and composition is done dynamically based on user context, constraints and preferences. Complex applications also support negotiations to provide optimal solution. eFlow and Ardagna et. al [30] are based on adaptive composition approach.
1.3 Research Issues and Challenges of Service Composition

Web services standards, specifications and languages do not support all the workflow patterns. Core Web services is stateless and does not have notation for event and notification. Standards and specifications of Web services are syntax oriented and lack semantic support to make the business processes machine processable.

Services composition is a very important aspect to realize enterprise integration and business process. For efficient composition of services, descriptions of functional and non-functional properties of services are to be defined. Business process modeling language to model the business process and to generate the composition schema as per the events is required. Runtime orchestration of the services should be based on the events and the requirements of a process under execution. Dynamic generation of composition schema, dynamic selection of services and runtime life-cycle management of a business process is needed. To achieve state, notification and execution monitoring during the execution of composite services, middleware support is required. Issues related to Web services and service composition are identified as under:

- **Service Description and Discovery:** For efficient composition, discovery, and selection of services, description of functional and non-functional properties of services are required. Interoperability, incompatible vocabularies and semantic interoperability among service providers and consumers should be resolved.

- **Modeling of Composite Service:** Business process modeling deals with design and execution of a business process. Modeling language needs to model the business process as per the events. Language needs to specify the flow of a process, services to be combined and the order in which services are to be executed. It also specifies the parameters, conditions, and events required for invoking a service. For dynamic business process, modeling language should provide support for dynamic schema generation, dynamic service selection and runtime life-cycle management of a business process.

- **Adaptive Composition:** Composite service should be adaptive to the changes (state, event and execution) likely to occur during the execution. It should be flexible to adjust to the dynamic enterprise environment. Enterprises are changing constantly, new service provider with better Quality of Service (QoS) may become available at any time, old or
previous version of services may be removed, existing services may withdraw execution or throw an exception during the execution. Business process should have the ability to manage such changes. The required support should be available in the form of middleware.

- **Dynamic service composition:** B2B interaction with dynamic and automated business processes is quite challenging. Most of the real world business processes keep changing. Dynamic business processes require dynamic discovery of services, dynamic service selection, and dynamic schema generation. Business processes behave according to various rules and policies. For generating composition schema at runtime, rules, and policies need to be considered. Once the composition schema is generated, services should be selected on the basis of various criteria such as functional and non-functional requirements, QoS, and consumer preferences. Services are distributed and require constant monitoring for exception handling, state, event and execution management.

- **State Management:** Web services are fundamentally stateless. Composite service is a series of services, where result of one service depends on prior service(s) and/or prepares for a subsequent service. A service acts upon stateful resources based on messages it receives and sends. It uses messages to determine the processing behavior of a process. Business process involves the transaction, which again constantly initiates the changes in the state of a process under execution. Business processes require runtime coordination, asynchronous integration, notification mechanism and state and transaction management. Core Web services standards lack the notion of state, stateful interactions, resource lifecycle management and notification of state changes. Web services themselves are not capable enough to provide required functionalities and various specifications have been proposed to achieve missing functionalities within Web services. Flexible and reliable ACID (Atomicity, Consistency, Isolation, Durability) transaction in a long running process and loosely coupled environment is a challenging issue [31, 32].

- **Execution Monitoring:** Execution of a business process requires scalable software architecture with workflow execution monitoring support. Web services are published and maintained by respective organizations. For the complete execution of a business
process, a common controlling mechanism to monitor the execution and lifecycle of a process is required. Again conflicting standards and specifications and lack of middleware support have raised challenges to resolve these issues.

- **Event and Notification**: Business services are event driven. Different specifications and mechanisms have been proposed to achieve eventing and notification in SOA. Two major specifications exist: WS-Notification and WS-Eventing. Microsoft has published WS-Eventing [33] and IBM and HP have published WS-Notification as a collection of specifications to address the same problem. These specifications use different approaches and terminology to address the same issues. These two specifications provide overlapping features. Apart from these specifications, Web Services stack [34] is flooded with numerous specifications related to routing, addressing, reliable messaging, transaction, orchestration etc. While developing a real life application, the challenge is to carefully select these specifications and to ensure their coherence after the implementation.

- Apart from these broader issues, issues related to coordination, transaction, security and performance are likely to arise due to dynamic and heterogeneous nature of composite service. Challenges such as inability to ensure scalability, robustness, and QoS related issues of Web services make it unfit for mission-critical and certain kinds of business applications [35-38].

### 1.4 Proposed Approach

Semantic web has originated from an Artificial Intelligence (AI) domain to provide knowledge-centric computing environment [39]. Grid computing provides support for data and computation intensive large-scale distributed computing system. Web Services and SOA aims to provide language and machine independent, loosely coupled services and architecture for integration of distributed heterogeneous components. Researchers and developer found relationships among these technologies to achieve collaboration, cooperation among scattered components with flexible and scalable open standard based global scale architecture.
1.4.1 Convergence of Web services, Semantic web and Grid computing

With the advancement of Web services and Semantic web, recent research merges these two paradigms as Semantic Web Services (SWS). Semantic Web Services aims to automate discovery, composition, invocation, and execution of Web services and to make Web services machine processable and interoperable [40]. Various semantic markup languages and standards have been proposed for the annotation of Web services. Among important standards; OWL for Services (OWL-S) [25], Web Service Modeling Ontology (WSMO) [41] and WSDL-S [26] are the known standards.

Global Grid Forum's (GGF) proposed OGSA as a convergence of SOA and grid. OGSA uses Web services as a core component to expose its core functionalities and combine the SOA and grid computing for business and scientific applications. With the replacement of Open Grid Service Infrastructure (OGSI) with WSRF, OGSA is now based on Web services standards such as WSRF, WS-Notification [42], and Web Services Distributed Management (WSDM) [43]. WSRF is proposed as a standard to converge Web services and Grid service [3].

The Semantic Grid [44] is described as an extension of grid computing where metadata is used to describe resource, information and services of the grid. Semantic helps in discovery, sharing and collaboration of resources. It also helps to achieve automation, to make services and process machine processable, and to enable cooperation between man and machine. Semantic provides knowledge in a grid environment, where intensive data and information integration are involved. At present Semantic grid is evolving and it is at experimental level. It lacks support in terms of framework, standards and architecture.

We plan convergence of Web Services, Semantic web and grid computing to resolve the issues of enterprise applications as shown in figure-1. Web services provides loosely coupled integration of scattered services. Semantic provides the knowledge and vocabulary of a domain and rules to design composition of Web services. Grid provides middleware support to achieve state, transaction, notification, execution, monitoring, and scalable enterprise architecture. We propose EDSOA to facilitate dynamic composition, negotiation, state, transaction, notification and middleware support for the automation of business processes as shown in figure-15.
1.5 Contributions

There exist various approaches for service composition but the aim of this thesis is to propose EDSOA to achieve event-driven dynamic Web services composition with the convergence of three emerging paradigms: Web services, Semantic web and grid computing.

More precisely, the major contributions of this thesis are as under:

- **Event-driven Service-oriented Architecture (EDSOA) [45-47]:** As an extension of SOA we have proposed EDSOA to capture the requirements of event-driven dynamic business process and to achieve event-driven dynamic composition of a business process.

- **Realization of EDSOA [47]:** We have proposed convergence of Web Services, Semantic Web, and Grid computing to achieve EDSOA. Web services provides loosely coupled integration of information from distributed sources. It also provides orchestration of heterogeneous services [48, 49]. Semantic provide interoperability for orchestration of a business process. Ontology and rules are used for event correlation, automated negotiation and discovery, and delivery of personalized context and location based recommendation [50-52]. Grid provides middleware for workflow execution and to
achieve state, transaction, notification, execution, monitoring and scalable enterprise architecture [45, 53].

- Modeling and Composition [46, 54]: We have proposed Event Calculus based approach to model the event-driven business process. Dynamic composition schema is generated based on events using ontology, rules, backward and forward chain algorithm.

- Grid Business Process [55]: A business process is likely to span across various distributed services. Development, deployment and execution of integrated services come with the challenge of its inherent heterogeneity. The software and hardware infrastructures are also heterogeneous. Proposed standards and specifications are conflicting, not yet matured and face many difficult challenges. We have proposed Grid Business Process to fulfill dynamic business process requirements.

- Service Grouping and Group Notification [56]: Business processes require integration with distributed heterogeneous services. Business processes are running in parallel and interacting with multiple services, partners and customers as per the requirement and policy. There is a need to aggregate information from multiple resources or services, to provide better query, search and group notification. We have proposed event-driven service grouping and group notification of stateful services.

- Policy-driven Grid Business Process [57]: There are a number of factors that both service provider and service consumer should consider before they interact with each other. In Web service selection phase both functional and non-functional requirements need to be considered. Composition of services requires dynamic services discovery and dynamic service selection. Service discovery and selection are depending on business services metadata, policy and event associated with the services. There is a need of dynamic services selection based on runtime environment such as content (semantics), context (event) and contract (policy). We have proposed event-driven dynamic services selection based on event, policy and semantic. Dynamic service selection will help in dynamic composition of business process and to deliver the efficient services to consumer as per their business context and request.
• For implementation of end-to-end EDSOA, we have evaluated different strategies of SOA. We have tested the capabilities of Web service, Semantic web and Grid computing standards and specifications. We have shown the effective integration of different tools and techniques for the development of enterprise application.

1.6 Organization of the Thesis

This thesis proposes an EDSOA and event-driven dynamic composition methods to provide state and notification management during the execution of business processes. An outline of the organization of this thesis is shown as follows:

Chapter 2 provides an overview of concepts of Service-oriented computing, standards and specifications related to Web services, Semantic web, Grid computing and Web services composition.

Chapter 3 discusses the proposed event-driven SOA lifecycle, EDSOA and realization of architecture using Agro-produce marketing use case scenario.

Chapter 4 presents a generic approach of event calculus based event-driven modeling and ECA rules based approach for event-driven dynamic composition and automatic schema generation.

Chapter 5 describes the concepts and implementation of our approach for grid business process. It presents the usage of grid to achieve the required middleware support for the execution of workflow and to achieve state and notifications during the execution of a business process. It also discusses the implementation of prototype system and development of grid business services using existing standards, protocols, and tools.

Chapter 6 shows the deployment and execution of grid business process prototype and discusses experiments and results.

Chapter 7 summarizes our work and discusses the future direction of this research work.