CONCLUSION AND FUTURE WORK
A characteristic feature of the software industry that makes it distinct from most of the other industries is the constant evolution of technologies. The new technologies are willfully and swiftly embraced by the stakeholders for several reasons such as – they provide solution to some real problems; they offer tangible benefits in the form of increased productivity, reduced time to market, increased Return On Investment (ROI), ease of use etc; also, as the new technologies evolve the vendors stop supporting the old ones forcing the users to change to new technologies. As a consequence, the investments in previous technologies lose their value. With technology transition, the existing software has to be either ported to the new technology or to the latest version of an existing technology. In cases where the software utilizes the older technology and is not changed, the existing, now legacy software needs to interoperate with new systems built using new technology. Thus, portability and interoperability of the existing systems becomes a significant issue in the wake of changing technologies. Besides changing technologies, there has been a major shift from building huge monolithic systems to building component-based systems. The individual components of these systems are built using the best technology available for the job. But, for the system to function as a single, coherent unit these components must be able to interact with each other. This again makes interoperability a vital issue in large, complex systems. In this backdrop a software development approach that could address these issues is desired and required.

Model Driven Architecture (MDA), an initiative by OMG, is an open, vendor-neutral approach to software development. In this architectural framework, the models are the
prime artifacts which drive the entire process of software development starting from requirements gathering through analysis, design, implementation, testing, deployment and maintenance. These models are formal in nature and therefore can be understood and processed by computers. In MDA, the models are specified at different levels of abstraction to represent the various aspects of the system. The different models at the heart of MDA are: Computation Independent Model (CIM), Platform Independent Model (PIM), Platform Specific Model (PSM) and Implementation Model. The CIM describes the problem domain and the requirements of the system in a vocabulary familiar to domain experts. The PIM captures the information regarding the data and processing for a system, independent of any platform or technology used. The PSM specifies the system in terms of constructs specific to the technology on which the system would be implemented. A single PIM may be transformed into several PSMs each targeting a different platform. As the PSM fits its technology rather closely, the transformation of PSM into code is straightforward. The key concept in MDA is the use of automated transformation tools to perform model-to-model and model-to-code transformations. The transformation definitions defined for the purpose are executed by these tools. The obvious benefits of MDA include enhanced productivity, improved software quality, portability, interoperability, reusability, increased ROI, reduced cost, reduced time to market and easy and simple maintenance and documentation. Thus, in MDA the PIM of the system is generated only once and can then be transformed into PSMs targeted on different technology platform – old or new. The deliberations for the MDA approach are highlighted in chapter 1.
Conclusion 1 Model Driven Architecture framework is an effective software development approach that mitigates the undesirable effects of technology change.

As the technologies advance, the computing models also evolve correspondingly. Cloud computing is the most recent computing model which represents the fifth generation of computing paradigm following mainframe computing, personal computing, client-server computing and web computing. Cloud computing is characterized by the presence of a cloud which hosts a variety of hardware, software and other infrastructure resources. These resources are offered to the consumers as services over high bandwidth linkages. These resources are owned, controlled and managed by the service providers. The consumers are only required to pay for the usage of these resources. The resources are made available to the users on demand and can be dynamically acquired from and released to a shared pool of configurable resources with great elasticity, in response to the changing needs of the consumer. An important characteristic of Cloud computing is that a service can be accessed across an interface as simple as a browser, using a variety of devices ranging from PCs, laptops, PDAs or even the mobile phones. In the recent years more and more big vendors have gradually joined the fray of cloud service providers.

A cloud SaaS in the Cloud computing paradigm refers to a service model whereby the software applications running in the cloud are provided as services to the consumers. As a cloud SaaS is a software component, the MDA approach may be enhanced to develop these software services in the cloud. A PIM of a cloud SaaS need to be developed only once, and may then be automatically transformed into one or more PSMs targeted on different platforms. Subsequently, the PSMs may be implemented as Cloud SaaS in a more direct and straightforward manner. Cloud computing provides several benefits to
its users by – i) eliminating the cost and complexity of buying, configuring and managing the resources needed to build and deploy applications which in turn are delivered as a service, ii) facilitating achievement of supercomputing capabilities from minimal resources, iii) enabling the clients to access data and applications from anywhere and at anytime, iv) eliminating the need to purchase software licenses, sign agreements or hire trained professionals for the job, etc.

Chapter 1 elaborates the concept of Cloud computing. Chapters 2 and 3 specify the CIM, PIM and PSMs of a cloud SaaS and the corresponding transformation definitions.

**Conclusion 2 Leveraging the Model Driven Architecture**

approach to develop cloud SaaS facilitates reaping the benefits of both, MDA and Cloud computing.

Cloud computing has evolved from a variety of technologies that include Service Oriented Architecture (SOA) and Web Services. SOA is an architectural framework in which business logic of a system is decomposed into distinct, smaller units of logic called services. Each service has a well-defined interface; these interfaces enable the services to communicate with each other. The Web services framework is currently providing the required technology to realize SOA. Web services framework comprises of XML-based industry standards that include Web Service Description Language (WSDL), Simple Object Access Protocol (SOAP) and Universal Description, Discovery and Integration (UDDI). The WSDL component of Web Services is responsible for describing the service. The service description specifies operations or functionalities provided by a service, the input and output messages (data) that are required or generated by the various operations exposed by the service, the endpoint or the port at which the service is available, the binding of the service to specific protocol etc. The service
description thus establishes a service contract – a set of conditions that must be met and accepted by a potential service requestor to enable successful communication. SOAP is the transport and communication protocol for the exchange of messages. UDDI is a standard for structuring service registries that keep track of service descriptions.

The essential characteristics possessed by a service in SOA are loose-coupling, autonomy, statelessness, abstraction, reusability, composability, discoverability and a formal contract that can be shared with other services. A cloud architecture based on service-oriented paradigm will have cloud SaaS (application) that inherently possess the underlying characteristics of the services in SOA. Chapter 1 in this thesis discusses the basic concepts of SOA and Web Services. Chapter 4 provides a detailed description on SOA and Web Services and also stresses on the need to develop SOA-based clouds.

**Conclusion 3** *A cloud in Cloud computing must not be considered as a different architecture. Instead, a service-oriented approach to cloud architecture will enable development of cloud SaaS using mechanisms that make them work well together, inside and outside of the enterprise.*

The software applications whether in an enterprise or in the cloud do not exist in isolation. They must be able to interact with each other. An SOA-based cloud has services that are self-controlled and self-managed; besides the services do not depend on other services for their execution. This enables each service to evolve independently. Any change in the functionality of the service does not affect its interaction with other services, so long as the service description remains the same. The interaction between the cloud services is facilitated by formal, well-defined interfaces which expose the
functionalities offered by a service to other services. Chapter 4 in this thesis describes the interoperability between two cloud SaaS, taken as example, in an SOA-based cloud.

**Conclusion 4** In an SOA-based cloud, as the Web services framework provides the necessary technology for the services to interoperate, the invocation of service remains unaffected by any change in the underlying logic of service functionality.

As mentioned in this thesis, SOA fosters intrinsic interoperability. The services in an SOA-based cloud would be inherently interoperable. It has been stressed throughout the thesis that the software development based on MDA approach would help alleviate the adverse effects of technology change. Since a service interface is a software component, the MDA approach may be leveraged to develop not only the cloud SaaS (application), but also the interface provided by it. As observed, the interface implementation technologies have also evolved over the years from CORBA-IDL to COM/DCOM and now the WSDL. In order to keep pace with evolving technologies, based on MDA approach, a PIM of the service interface may be defined. This interface PIM may then be transformed into one or more PSMs targeted on different implementation technologies. Chapter 5 in this thesis presents a PIM for a cloud SaaS interface which is then transformed into a PSM targeted on WSDL. The transformation is automated using a transformation tool developed by the authors. This tool executes the transformation definition specified for the purpose.

**Conclusion 5** The MDA approach may be enhanced to develop not only the software applications running as
services in the cloud but also the interfaces of these services.

An essential requirement of MDA is that the models defining the various aspects of the system must conform to their respective metamodels. These metamodels provide the schema for the semantic data that is to be exchanged, stored or processed. Within the MDA framework, meta-modeling provides a mechanism to define modeling languages. Besides, in MDA, the transformation rules, which comprise the transformation definition, use metamodels to describe how source model constructs can be transformed into target model constructs.

**Conclusion 6** The source and target models must conform to their respective metamodels, as these metamodels are used by the transformation rules to define the transformations.

Chapter 5 in this thesis specifies a metamodel for WSDL, a language that is used for describing the cloud service interface.

In this thesis, the authors present a model-driven approach to ensure interoperability among the cloud SaaS in spite of changing hardware and software technologies. Also, it is ascertained that the MDA approach does not eclipse the various technologies. Rather, it works with them synergistically and enhances their efficiency.

**Future Work**

One of the underlying principles of SOA is composability. According to this principle, services may compose other services. This principle ensures that services are designed such that they can participate as effective members of other service compositions if ever
required, irrespective of whether the service itself composes others to accomplish its work. In addition, composability is simply another form of reuse.

In cloud architecture, very often a service consumer’s requirement is such that the required functionality can be achieved by composing the service from one or more other services. In an SOA-based cloud, the Model Driven Approach may be leveraged to address this aspect of Cloud computing. This will promote reusability of cloud services and also creation of abstraction layers.