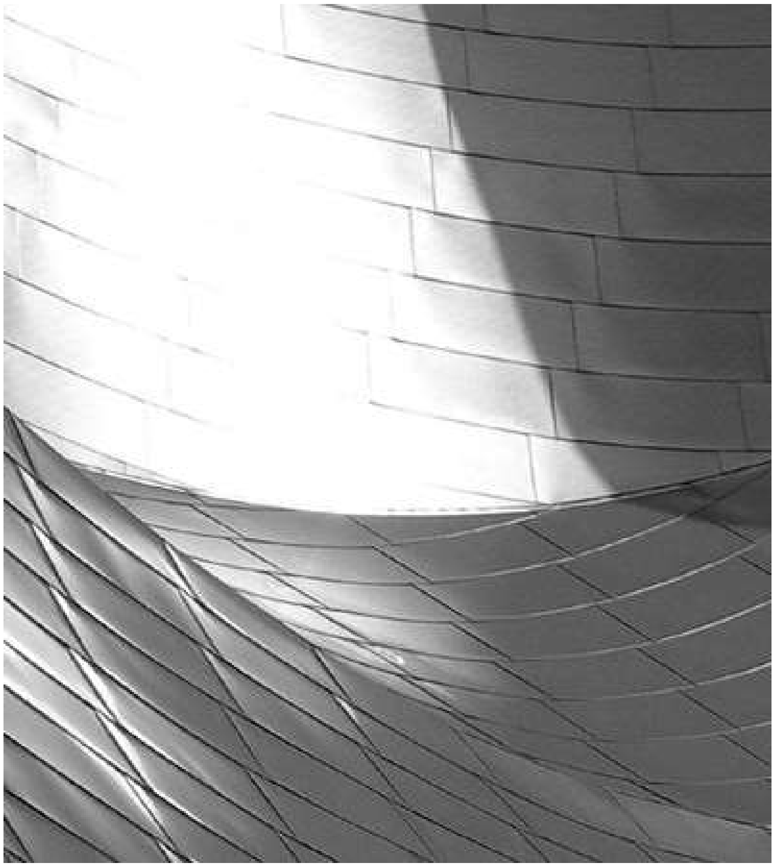
Appendix E. Emerging Technologies



This appendix presents two primary emerging technologies relevant to cloud computing.

E.1. Autonomic Computing

Autonomic computing refers to the ability of a computer system to self-manage, which includes the following capabilities:

- *Self-Configuration* – ability to accommodate varying and possibly unpredictable conditions
- *Self-Healing* – ability to remain functioning when problems arise
- *Self-Protection* – ability to detect threats and take appropriate actions
- *Self-Optimization* – constant monitoring for optimal operation

An autonomic computing system maintains comprehensive knowledge of its components and the operating environment (self-knowledge) so that it can self-react to external inputs (self-adaptation). In order to be able to react automatically, these systems have built-in sensors that monitor the environmental conditions and external inputs (sensory capacity) in order to determine and execute the appropriate response actions.

Self-adaptation is triggered by changing conditions and driven by system objectives. System objectives can be specified as high-level policies and rules that are interpreted by decision-making logic designed to maintain the system in an operational state that remains compliant with pre-specified rules in the face of changing external conditions.

Autonomic systems are commonly modeled as closed-loop control systems where sensors monitor the external conditions and feed the collected data back to the decision logic. This optimizes the system configuration towards the defined system purpose. Actuation is automatic and does not require any human intervention. The aim is to have systems that can self-run while adapting to increasing system complexity, without the need for any user input. These systems can have high levels of built-in artificial intelligence that remain hidden from the users. Autonomic computing is one of the building blocks of pervasive computing, which is an anticipated future computing model.

Autonomic computing supports several cloud computing characteristics, including:

- *Elasticity* – Autonomic systems can monitor usage conditions and leverage cloud-based IT resources to automatically acquire and free IT resources as needed for the purpose of maintaining required service levels.
- *Resiliency* – Autonomic systems can automatically detect unavailable IT resources and self-respond to allocate alternative IT resources as required.

Each cloud-based IT resource is accompanied by an automated scaling listener that is connected to a recipient IT resource on the cloud consumer side. The link between the automated scaling listener and recipient enables the cloud service consumer to automatically react to changes. For example, if one cloud service becomes unavailable, an alternative cloud service in another cloud is further scaled out to handle the increased workload.

E.2. Grid Computing

Grid computing is a specialized form of distributed computing where multiple IT resources, also called grid nodes, collaboratively provide large computational capacity. The distinctive characteristic of grid computing, as opposed to other types of high-performance systems like cluster computing, is that computing IT resources are more loosely coupled and pertain to multiple administrative domains. These IT resources are usually heterogeneous and geographically dispersed as well.

IT resources used for grid computing are connected through a communications network that includes the Internet, and can be either privately or publicly owned and administered. These IT resources use common distribution and coordination middleware to distribute the workload among processing nodes and coordinate the processing results.

As a consequence of their loosely coupled relationship, the workload is distributed to individual nodes that carry out a given processing task by itself, without communication with the other nodes in the system. Grid nodes typically also have autonomy in relation to the grid computing system so that nodes can join in and disappear independently. The middleware that coordinates the workload processing responds to grid node failure by dispatching incomplete computing tasks to other available nodes.

A second implication of loose coupling is that communications capabilities are limited since grid nodes do not share the high-speed network. Applications need to be specifically designed to operate in grids, usually by dividing a large computing task into smaller tasks that can be processed in parallel by different computing systems that are not required to have direct inter-communication capabilities.

Grid-enabled applications can address various computing requirements, ranging from large datasets that need to be processed independently (data partitioning) and individual data bits that require manageable computing capacities (genome and proteome analysis), to computation that needs to be broken down into smaller, more manageable workloads (computing workload partitioning).

The parallelization of computing tasks is programmed during application design-time and relies on human expertise, with a complexity that can limit the applicability of the computing model.

The origin of grid computing emerged from the opportunity to build high-performance computing systems using the idle capacity of existing IT resources, as well as the goal of building high-performance computing systems with low-cost commodity hardware. IT resources can be added to the grid voluntarily, or as a result of the corporate policies and agreements with partner organizations.

Comprised of IT resources that have different ownerships and originate from different administrative domains, grid computing systems face prominent security issues. On one hand, grid node owners need to trust distribution and coordination middleware managers not to execute any unsecure or malicious code in the client computers. Conversely, middleware managers need to trust grid node owners not to intentionally produce erroneous results for the computing tasks they are assigned.

Common characteristics are shared between grid and cloud computing, since both are based on the use of networked access, shared, scalable, and resilient IT resources that can be provided by third parties. As such, grid and cloud computing providers encounter similar challenges when establishing computing platforms.

Service Grids

Service grid platforms can be viewed as an extension of infrastructure that provides horizontally scalable processing and caching. A service grid can span many physical servers, resulting in a computing platform that can provide built-in load balancing and failover support. The use of service grids can dramatically increase the scalability and reliability of cloud services and cloud-based IT resources.

A service grid can be constructed in different ways, such as using a grid computing system to produce virtualized grid servers that span numerous physical servers (Figure E.1). This produces a computing platform that offers a number of advantages, such as:

- horizontally scalable processing and caching
- built-in load balancing of cloud-based IT resources
- failover support and availability
- scalability and reliability of cloud-based services and IT resources

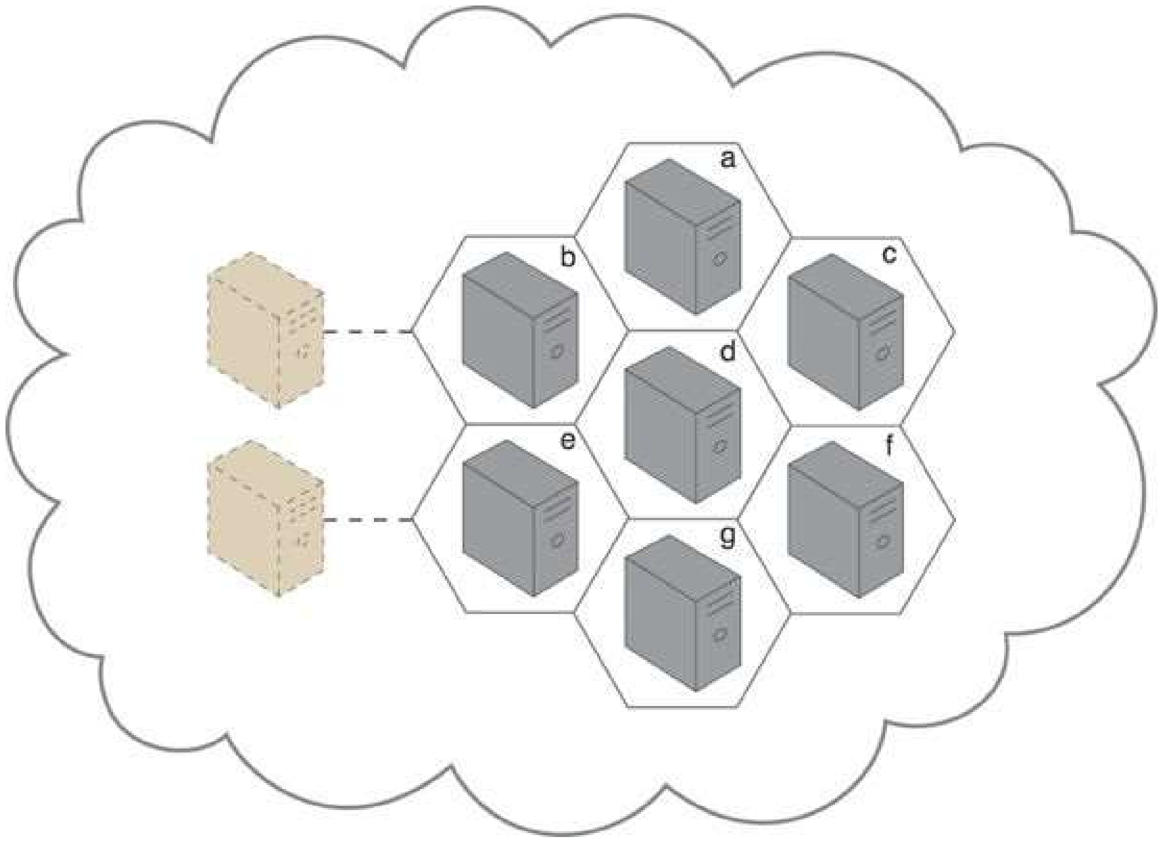


Figure E.1. A grid computing system is exposed as a virtual grid server.

Cloud-based IT resources can be used in the construction and scaling of service grids. Cloud consumers that use cloud-based IT resources to develop service grids only have to pay for the IT resources that they actually consume (Figure E.2).

The service grids that are illustrated in Figures E.1 and E.2 can effectively support a number of cloud computing characteristics, including:

- *Multitenancy and Resource Pooling* – sharing of the same underlying grid computing system by multiple cloud consumers
- *Elasticity* – scaling out of virtual servers
- *Resiliency* – built-in failover support of grid systems

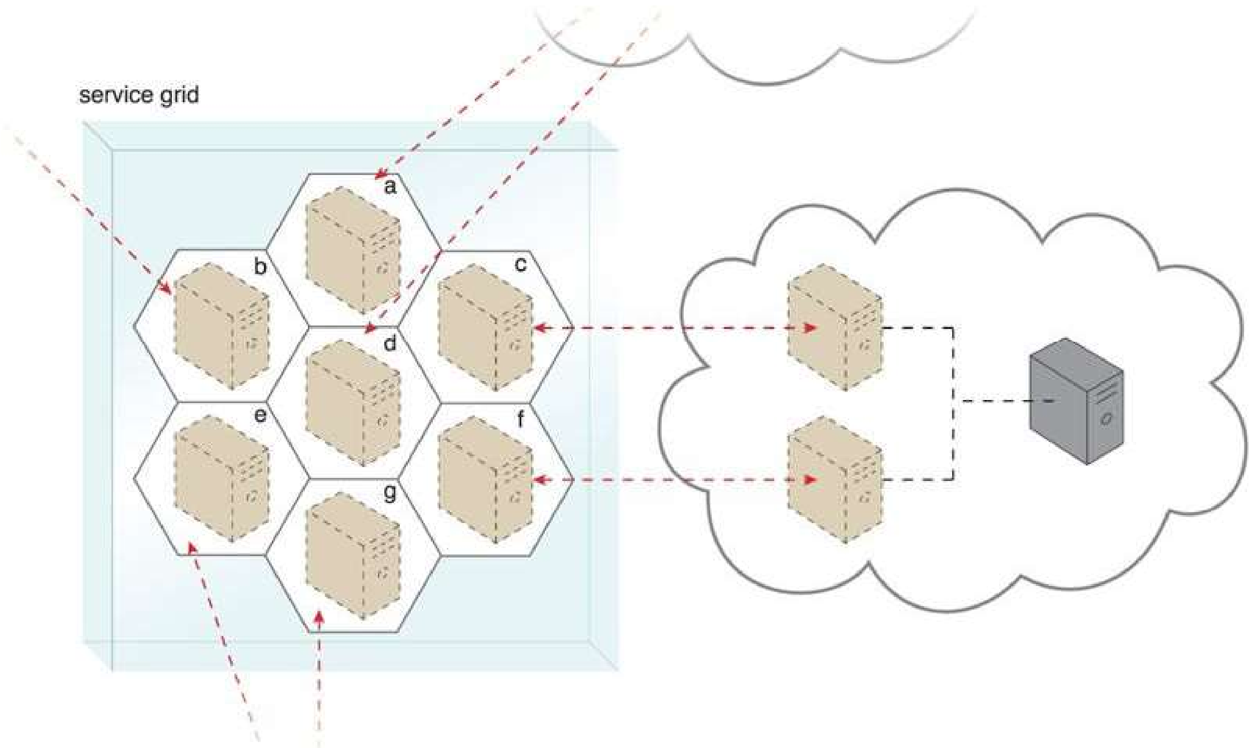


Figure E.2. Cloud-based IT resources that belong to different clouds collectively form a grid computing system.