Chapter 1
Content Delivery Networks: State of the Art, Insights, and Imperatives

Mukaddim Pathan, Rajkumar Buyya and Athena Vakali

1.1 Introduction

Over the last decades, users have witnessed the growth and maturity of the Internet which has caused enormous growth in network traffic, driven by the rapid acceptance of broadband access, the increases in systems complexity, and the content richness. The over-evolving nature of the Internet brings new challenges in managing and delivering content to users, since for example, popular Web services often suffer congestion and bottlenecks due to the large demands posed on their services. Such a sudden spike in Web content requests (e.g. the one occurred during the 9/11 incident in USA) is often termed as flash crowds [14] or SlashDot [11] effects. It may cause heavy workload on particular Web server(s), and as a result a "hotspot" [14] can be generated. Coping with such unexpected demand causes significant strain on a Web server and eventually the Web servers are totally overwhelmed with the sudden increase in traffic, and the Web site holding the content becomes temporarily unavailable.

A Content Delivery Network (CDN) [47, 51, 54, 61, 63] is a collaborative collection of network elements spanning the Internet, where content is replicated over several mirrored Web servers in order to perform transparent and effective delivery of content to the end users. Collaboration among distributed CDN components can occur over nodes in both homogeneous and heterogeneous environments. CDNs have evolved to overcome the inherent limitations of the Internet in terms of user perceived Quality of Service (QoS) when accessing Web content. They provide services that improve network performance by maximizing bandwidth, improving

Mukaddim Pathan
GRIDS Lab, Department of CSSE, The University of Melbourne, Australia,
e-mail: apathan@csse.unimelb.edu.au

Rajkumar Buyya
GRIDS Lab, Department of CSSE, The University of Melbourne, Australia,
e-mail: raj@csse.unimelb.edu.au

Athena Vakali
Department of Informatics, Aristotle University of Thessaloniki, Greece,
e-mail: avakali@csd.auth.gr

R. Buyya et al. (eds.), *Content Delivery Networks*,

accessibility, and maintaining correctness through content replication. The typical functionalities of a CDN include:

- *Request redirection and content delivery services,* to direct a request to the closest suitable CDN cache server using mechanisms to bypass congestion, thus overcoming flash crowds [14] or SlashDot [11] effects.
- *Content outsourcing and distribution services,* to replicate and/or cache content from the origin server to distributed Web servers.
- *Content negotiation services,* to meet specific needs of each individual user (or group of users).
- *Management services,* to manage the network components, to handle accounting, and to monitor and report on content usage.

The major application domains of CDNs are public content networking services, enterprise content networks, and edge services. As CDNs being a thriving research field, advances, solutions, and new capabilities are being introduced constantly. Therefore, in this chapter, we capture a "snapshot" of the state of the art at the time of writing this book. However, it can be expected that the core information and principles presented in this chapter will remain relevant and useful for the readers.

The remainder of this chapter is structured as follows: we start with providing an overview of CDNs. Next we describe the background highlighting the evolution of CDNs and identify uniqueness of CDNs from other related distributed computing paradigms. In Sect. 1.4 we provide insights for CDNs. The state of the art in CDN landscape is presented in Sect. 1.5. Our visions about future technological evolutions in CDNs domain follows next, along with a research roadmap in Sect. 1.7 by exploring future research directions. Finally, Sect. 1.8 concludes the chapter.

1.2 Overview

Figure 1.1 shows the model of a CDN where the replicated Web server clusters spanning the globe are located at the edge of the network to which end users are connected. A CDN distributes content to a set of Web servers, scattered over the globe, for delivering content to end users in a reliable and timely manner. The content is replicated either on-demand when users request for it, or it can be replicated beforehand, by pushing the content to the distributed Web servers. A user is served with the content from the nearby replicated Web server. Thus, the user ends up unknowingly communicating with a replicated CDN server close to it and retrieves files from that server.

1.2.1 Terminologies

In the context of CDNs, *content delivery* describes an action of servicing content based on end user requests. *Content* refers to any digital data resources and

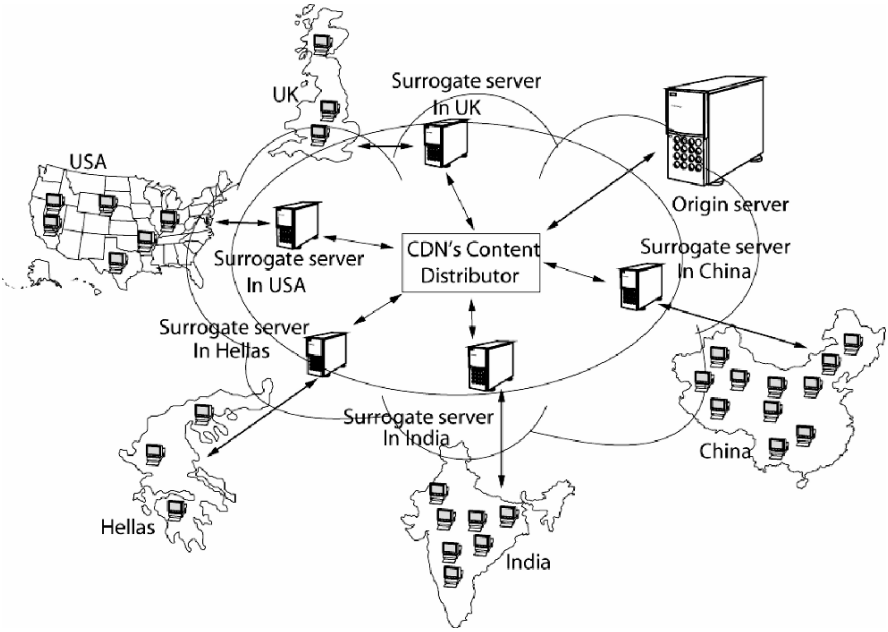

Fig. 1.1 Model of a CDN

it consists of two main parts: the *encoded media* and *metadata* [53]. The encoded media includes static, dynamic, and continuous media data (e.g. audio, video, documents, images and Web pages). Metadata is the content description that allows identification, discovery, and management of multimedia data, and facilitates its interpretation. Content can be pre-recorded or retrieved from live sources; it can be persistent or transient data within the system [53]. CDNs can be seen as a new virtual overlay to the Open Systems Interconnection (OSI) network reference model [32]. This layer provides overlay network services relying on application layer protocols such as Hyper Text Transfer Protocol (HTTP) or Real Time Streaming Protocol (RTSP) for transport [26].

The three main entities in a CDN system are the following: *content provider*, *CDN provider*, and *end users*. A *content provider* or *customer* is one who delegates the Uniform Resource Locator (URL) name space of the Web objects to be distributed. The *origin server* of the content provider holds those objects. A *CDN provider* is a proprietary organization or company that provides infrastructure facilities to content providers in order to deliver content in a timely and reliable manner. *End users* or *clients* are the entities who access content from the content provider's Web site.

CDN providers use *caching* and/or *replica servers* located in different geographical locations to replicate content. CDN cache servers are also called *edge servers* or *surrogates*. The edge servers of a CDN are called *Web cluster* as a whole. CDNs distribute content to the edge servers in such a way that all of them share the same

content and URL. Client requests are redirected to the nearby optimal edge server and it delivers requested content to the end users. Thus, transparency for users is achieved. Additionally, edge servers send accounting information for the delivered content to the accounting system of the CDN for traffic reporting and billing purposes.

1.2.2 CDN Components

Figure 1.2 shows the general architecture of a CDN system which involves four main components:

- The content-delivery component which consists of the origin server and a set of replica servers that deliver copies of content to the end users;
- The request-routing component which is responsible for directing client requests to appropriate edge servers and for interacting with the distribution component to keep an up-to-date view of the content stored in the CDN caches;
- The distribution component which moves content from the origin server to the CDN edge servers and ensures consistency of content in the caches; and
- The accounting component which maintains logs of client accesses and records the usage of the CDN servers. This information is used for traffic reporting

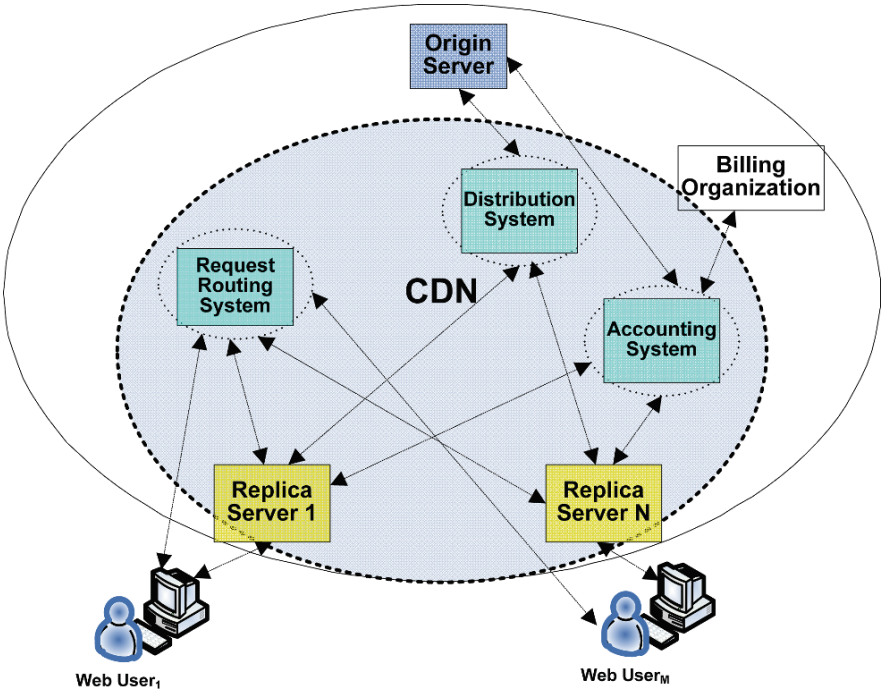

Fig. 1.2 Architectural components of a CDN

and usage-based billing by the content provider itself or by a third-party billing organization.

A CDN focuses on building its network infrastructure to provide the following services and functionalities: storage and management of content; distribution of content among edge servers; cache management; delivery of static, dynamic, and streaming content; backup and disaster recovery solutions; and monitoring, performance measurement, and reporting.

A content provider (i.e. customer) can sign up with a CDN provider for service and have its content placed on the cache servers. In practice, CDNs typically host third-party content including static content (e.g. static HTML pages, images, documents, software patches), streaming media (e.g. audio, real time video), User Generated Videos (UGV), and varying content services (e.g. directory service, e-commerce service, file transfer service). The sources of content include large enterprises, Web service providers, media companies, and news broadcasters. Typical customers of a CDN are media and Internet advertisement companies, data centers, Internet Service Providers (ISPs), online music retailers, mobile operators, consumer electronics manufacturers, and other carrier companies. Each of these customers wants to publish and deliver their content to the end users on the Internet in a reliable and timely manner. End users can interact with the CDN by specifying the content/service request through cell phone, smart phone/PDA, laptop and desktop. Figure 1.3 depicts the different content/services served by a CDN provider to end users.

CDN providers charge their customers according to the content delivered (i.e. traffic) to the end users by their edge servers. CDNs support an accounting mechanism that collects and tracks client usage information related to request-routing, distribution, and delivery [26]. This mechanism gathers information in real time

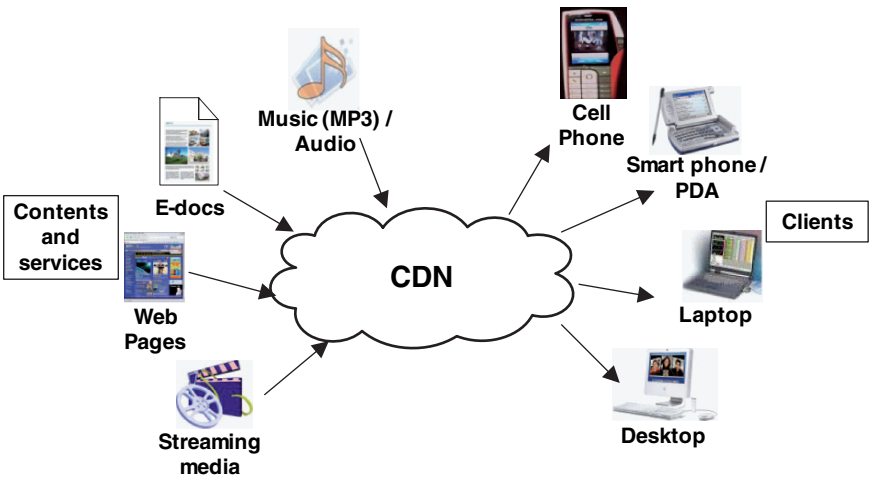

Fig. 1.3 Content/services provided by a CDN

and collects it for each CDN component. This information can be used in CDNs for accounting, billing, and maintenance purposes. The average cost of charging of CDN services is quite high [35], often out of reach for many small to medium enterprises (SME) or not-for-profit organizations. The most influencing factors [47] affecting the price of CDN services include:

- Bandwidth usage which is measured by the content provider to charge (per Mbps) customers typically on a monthly basis;
- Variation of traffic distribution which characterizes pricing under different situations of congestion and bursty traffic;
- Size of the content replicated over edge servers which is a critical criterion for posing charges (e.g. price per GB) on customer audiences;
- Number of edge servers which capture the ability of a CDN provider to offer content at charges that will not overcome the typical caching scenarios; and
- Reliability and stability of the whole system and security issues of outsourcing content delivery also inhibit a cost of sharing confidential data which varies over different content providers on the basis of the type of the protected content.

1.3 Background and Related Systems

Content providers view the Web as a vehicle to bring rich content to their users since decreases in services quality, along with high access delays (mainly caused by long download times) leaves users in frustration. Companies earn significant financial incentives from Web-based e-business and they are concerned to improve the service quality experienced by the users while accessing their Web sites. As such, the past few years have seen an evolution of technologies that aim to improve content delivery and service provisioning over the Web. When used together, the infrastructures supporting these technologies form a new type of network, which is often referred to as "content network" [26].

1.3.1 The Evolution of CDNs

Several content networks attempt to address the performance problem by using different mechanisms to improve QoS:

- An initial approach is to modify the traditional Web architecture by improving the Web server hardware adding a high-speed processor, more memory and disk space, or maybe even a multi-processor system. This approach is not flexible, since small enhancements are not possible and at some point, the complete server system might have to be replaced [31].
- Caching proxy deployment by an ISP can be beneficial for the narrow bandwidth users accessing the Internet, since to improve performance and reduce bandwidth

utilization, caching proxies are deployed close to the users. Caching proxies may also be equipped with technologies to detect a server failure and maximize efficient use of caching proxy resources. Users often configure their browsers to send their Web request through these caches rather than sending directly to origin servers. When this configuration is properly done, the user's entire browsing session goes through a specific caching proxy. Thus, the caches contain most popular content viewed by all the users of the caching proxies.

- A provider may also deploy different levels of local, regional, international caches at geographically distributed locations. Such arrangement is referred to as *hierarchical caching*. This may provide additional performance improvements and bandwidth savings [17]. The establishment of server farms is a more scalable solution which has been in widespread use for several years. A server farm is comprised multiple Web servers, each of them sharing the burden of answering requests for the same Web site [31]. It also makes use of a Layer 4-7 switch (intelligent switching based on information such as URL requested, content type, and username, which can be found in layers 4-7 of the OSI stack of the request packet), Web switch or content switch that examines content requests and dispatches them among the group of servers. A server farm can also be constructed with surrogates instead of a switch [24]. This approach is more flexible and shows better scalability. Moreover, it provides the inherent benefit of fault tolerance. Deployment and growth of server farms progresses with the upgrade of network links that connects the Web sites to the Internet.
- Although server farms and hierarchical caching through caching proxies are useful techniques to address the Web performance problem, they have limitations. In the first case, since servers are deployed near the origin server, they do little to improve the network performance due to network congestion. Caching proxies may be beneficial in this case. But they cache objects based on client demands. This may force the content providers with a popular content source to invest in large server farms, load balancing, and high bandwidth connections to keep up with the demand. To address these limitations, another type of content network has been deployed in late 1990s. This is termed as *Content Distribution Network* or *Content Delivery Network*, which is a system of computers networked together across the Internet to cooperate transparently for delivering content to end users.

With the introduction of CDN, content providers started putting their Web sites on a CDN. Soon they realized its usefulness through receiving increased reliability and scalability without the need to maintain expensive infrastructure. Hence, several initiatives kicked off for developing infrastructure for CDNs. As a consequence, Akamai Technologies [1, 27] evolved out of an MIT research effort aimed at solving the flash crowd problem and scientists developed a set of breakthrough algorithms for intelligently routing and replicating content over a large network of distributed servers spanning the globe. Within a couple of years, several companies became specialists in providing fast and reliable delivery of content, and CDNs became a huge market for generating large revenues. The flash crowd events [14, 34] like the 9/11 incident in USA [10], resulted in serious caching problems for some sites. This influenced the providers to invest more in CDN infrastructure development, since

CDNs provide desired level of protection to Web sites against flash crowds. First generation CDNs mostly focused on static or Dynamic Web documents [36, 61]. On the other hand, for second generation of CDNs the focus has shifted to Video-on-Demand (VoD), news on-demand, audio and video streaming with high user interactivity. The CDNs of this generation may also be dedicated to deliver content to mobile devices. However, most of the research efforts on this type of CDNs are still in research phase and have not yet exclusively reached the market. We anticipate that the third generation CDNs would be community-based CDNs, i.e. it would be mainly driven by the common "people" or the average end users. More information on such community-based CDNs can be found in Chap. 15 of this book. Figure 1.4 shows the evolutions of CDNs over time with a prediction of their evolution in the upcoming years.

With the booming of the CDN business, several standardization activities also emerged since vendors started organizing themselves. The Internet Engineering Task Force (IETF) as an official body has taken several initiatives through releasing Request For Comments (RFCs) [15, 16, 24, 26] in relation to many research initiatives in this domain. Other than IETF, several other organizations such as Broadband Services Forum (BSF) [3], ICAP forum [6], Internet Streaming Media Alliance [7] have taken initiatives to develop standards for delivering broadband content, streaming rich media content – video, audio, and associated data – over the Internet. In the same breath, by 2002, large-scale ISPs started building their own CDN functionality, providing customized services.

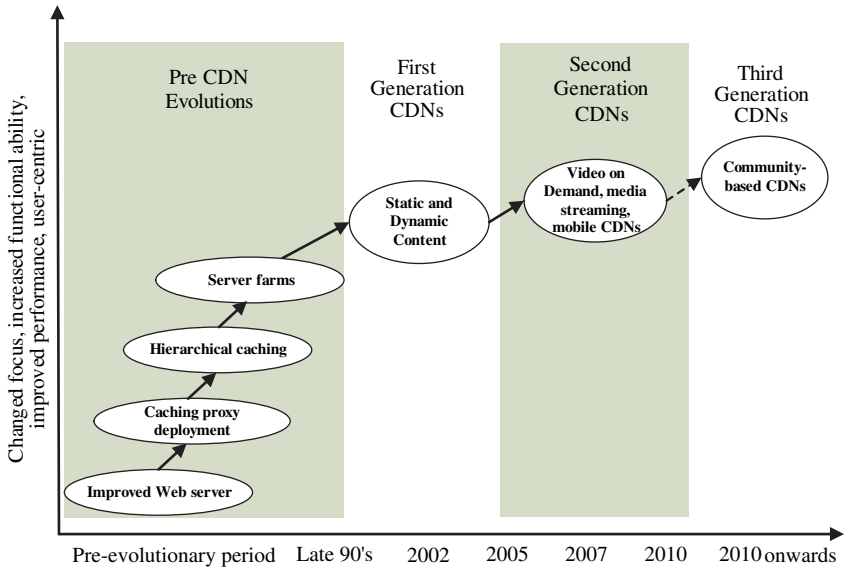

Fig. 1.4 CDN evolutions

1.3.2 Related Systems

Data grids, distributed databases, and peer-to-peer (P2P) networks are three distributed systems that have some characteristics in common with CDNs. These three systems have been described here in terms of requirements, functionalities, and characteristics. Table 1.1 presents the comparison between CDNs and these three related systems based on their unique characteristics/features.

1.3.2.1 Data Grids

A data grid [43, 62] is a data intensive computing environment that provides services to the users in different locations to discover, transfer, and manipulate large datasets stored in distributed repositories. At the minimum, a data grid provides two basic functionalities: a high-performance, reliable data transfer mechanism, and a scalable replica discovery and management mechanism [22]. A data grid consists of computational and storage resources in different locations connected by high-speed networks. They are especially targeted to large scientific applications such as high energy physics experiments at the Large Hadron Collidor [37], astronomy projects – Virtual Observatories [59], and protein simulation – BioGrid [2] that require analyzing a huge amount of data. The data generated from an instrument, experiment, or a network of sensors is stored at a principle storage site and is transferred to other storage sites around the world on request through the data replication mechanism. Users query the local replica catalog to locate the datasets that they require. With proper rights and permissions, the required dataset is fetched from the local repository if it is present there; otherwise it is fetched from a remote repository. The data may be transmitted to a computational unit for processing. After processing, the results may be sent to a visualization facility, a shared repository, or to individual users' desktops. Data grids promote an environment for the users to analyze data, share the results with the collaborators, and maintain state information about the data seamlessly across organizational and regional boundaries. Resources in a data grid are heterogeneous and are spread over multiple administrative domains. Presence of large datasets, sharing of distributed data collections, having the same logical namespace, and restricted distribution of data can be considered as the unique set of characteristics for data grids. Data grids also contain some application specific characteristics. The overall goal of data grids is to bring together existing distributed resources to obtain performance gain through data distribution. Data grids are created by institutions who come together to share resources on some shared goal(s) by forming a Virtual Organization (VO). On the other hand, the main goal of CDNs is to perform caching of data to enable faster access by end users. Moreover, all the commercial CDNs are proprietary in nature – individual companies own and operate them.

Table 1.1 Comparison between CDNs and related systems

Features	CDNs	Data Grids	Distributed Databases	P2P Networks
Category	A collection of networked computers spanning the Internet	Data intensive computing environment	Locally organized collection of data distributed across multiple physical locations	Information retrieval network formed by ad-hoc aggregation of resources
Constitution	Distribution of cache servers to the edge of the Internet	Formation of a VO of participating institutions.	Federation or splitting of existing database(s)	Collaboration among peers
Main goal	Reducing Web latency during content delivery	Performance gain through data distribution by pre-staging, optimal source selection, and high speed data movement	Integration of existing databases and replication of database fragments in a transparent manner	File sharing among peers
Integrity	Integrity between caches	Integrity between data grid replicas	Integrity between multiple DBs	N/A
Consistency	Strong cache consistency between replicated content	Weak consistency between data grid replicas	Strong database consistency between distributed DBs	Weak consistency between cached content
Autonomy	None	Autonomous participants	Autonomous DDB sites	Autonomous peers
Operational activities	Content caching	Seamless analysis, collaboration, and maintenance of data across organizational and regional boundaries	Query processing, optimization, and management	Locating or caching content, encrypting, retrieving, decrypting, and verifying content
Administration	Individual companies. Proprietary in nature	Institutions who cooperate on some shared goals	Single authoritative entity	Self-interested end users/peers

1.3.2.2 Distributed Databases

A Distributed Database (DDB) [21, 45] is a logically organized collection of data distributed across multiple physical locations. It may be stored in multiple computers located in the same physical location, or may be dispersed over a network of interconnected computers. Each computer in a distributed database system is a node. A node in a distributed database system acts as a client, server, or both depending on the situation. Each site has a degree of autonomy, is capable of executing a local query, and participates in the execution of a global query. A distributed database can be formed by splitting a single database or by federating multiple existing databases. The distribution of such a system is transparent to the users as they interact with the system as a single logical system. The transactions in a distributed database are transparent and each transaction must maintain integrity across multiple databases. Distributed databases have evolved to serve the need of large organizations that need to replace existing centralized database systems, interconnect existing databases, and to add new databases as new organizational units are added. Applications provided by DDB include distributed transaction processing, distributed query optimization, and efficient management of resources. DDBs are dedicated to integrate existing diverse databases to provide a uniform, consisting interface for query processing with increased reliability and throughput. Integration of databases in DDBs is performed by a single organization. Like DDBs, the entire network in CDNs is managed by a single authoritative entity. However, CDNs differ from DDBs in the fact that CDN cache servers do not have the autonomic property as in DDB sites. Moreover, the purpose of CDNs is content caching, while DDBs are used for query processing, optimization, and management.

1.3.2.3 P2P Networks

P2P networks [13, 44] are designed for the direct sharing of computer resources rather than requiring any intermediate and/or central authority. They are characterized as information retrieval networks that are formed by ad-hoc aggregation of resources to form a fully or partially decentralized system. Within a P2P system, each peer is autonomous and relies on other peers for resources, information, and forwarding requests. Ideally there is no central point of control in a P2P network. Therefore, the participating entities collaborate to perform tasks such as searching for other nodes, locating or caching content, routing requests, encrypting, retrieving, decrypting, and verifying content. P2P systems are more fault-tolerant and scalable than the conventional centralized system, as they have no single point of failure. An entity in a P2P network can join or leave anytime. P2P networks are more suited to the individual content providers who are not able to access or afford the common CDN. An example of such system is BitTorrent [33], which is a popular P2P file sharing application. Content and file sharing P2P networks are mainly focused on creating efficient strategies to locate particular files within a group of peers, to provide reliable transfers of such files in case of high volatility, and to

manage heavy traffic (i.e. flash crowds) caused by the demand for highly popular files. This is in contrast to CDNs where the main goal lies in respecting client's performance requirements rather than efficiently sharing file/content among peers. Moreover, CDNs differ from the P2P networks because the number of nodes joining and leaving the network per unit time is negligible in CDNs, whereas the rate is important in P2P networks.

1.4 Insights for CDNs

From the above discussion it is clear that a CDN is essentially aimed at content providers or customers who want to ensure QoS to the end users when accessing their Web content. The analysis of present day CDNs reveals that, at the minimum, a CDN focuses on the following business goals: scalability, security, reliability, responsiveness and performance.

1.4.1 Scalability

Scalability refers to the ability of the system to expand in order to handle new and large amounts of data, users, and transactions without any significant decline in performance. To expand to a global scale, CDN providers need to invest time and costs in provisioning additional network connections and infrastructures. It includes provisioning resources dynamically to address flash crowds and varying traffic. A CDN should act as a shock absorber for traffic by automatically providing capacity-on-demand to meet the requirements of flash crowds. This capability allows a CDN to avoid costly over-provisioning of resources and to provide high performance to every user.

Within the structure of present day CDN business model, content providers pay the CDN providers to maximize the impact of their content. However, current trends reveal that the type of applications that will be supported by CDNs in future, will transform the current business model [53]. In future, the content providers as well as the end users will also pay to receive high quality content. In this context, scalability will be an issue to deliver high quality content, maintaining low operational costs.

1.4.2 Security

One of the major concerns of a CDN is to provide potential security solutions for confidential and high-value content [19]. Security is the protection of content against unauthorized access and modification. Without proper security control, a CDN platform is subject to cyber fraud, Distributed Denial-of-Service (DDoS) attacks,

viruses, and other unwanted intrusions that can cripple business. A CDN aims at meeting the stringent requirements of physical, network, software, data, and procedural security. Once the security requirements are met, a CDN can eliminate the need for costly hardware and dedicated component to protect content and transactions. In accordance to the security issues, a CDN provider combat against any other potential risk concerns including DDoS attacks or other malicious activity that may interrupt business.

1.4.3 Reliability, Responsiveness, and Performance

Reliability refers to when a service is available and what are the bounds on service outages that may be expected. A CDN provider can improve client access to specialized content through delivering it from multiple locations. For this a fault-tolerant network with appropriate load balancing mechanisms is to be implemented [42]. Responsiveness implies, while in the face of possible outages, how soon a service would start performing the normal course of operation. Performance of a CDN is typically characterized by the response time (i.e. latency) perceived by end users. Slow response time is the single greatest contributor to customers' abandoning Web sites and processes. The reliability and performance of a CDN is affected by the distributed content location and routing mechanism, as well as by data replication and caching strategies. Hence, a CDN employs caching and streaming to enhance performance especially for delivery of media content [57]. A CDN hosting a Web site also focuses on providing fast and reliable service since it reinforces the message that the company is reliable and customer-focused.

1.5 Existing CDNs: State of the Art

In this section, we provide the state of art in current CDN landscape. We also describe the different services and technologies of existing CDNs. First, we provide a brief description on commercial CDNs (e.g. Akamai, EdgeStream, Limelight Networks, and Mirror Image) which exist in the content distribution space. Then we present a snapshot on academic CDNs (e.g. CoDeeN, Coral, and Globule) which gives a picture of what the CDN technologies are at this moment.

1.5.1 Commercial CDNs

Most or all of the operational CDNs are developed by commercial companies which are subject to consolidation over time due to acquisition and/or mergers. Hence, in the section, we focus on studying only those commercial CDNs that have been

Table 1.2 Summary of the existing commercial CDNs

CDN Name	Service Type	Coverage	Products/Solutions (If any)
Akamai www.akamai.com	Provides CDN service, including streaming	Covers 85% of the market. 25,000 servers in 900 networks in 69 countries. It handles 20% of total Internet traffic today	Edge Platform for handling static as well as dynamic content, Edge Control for managing applications, and Network Operations Control Center (NOCC)
EdgeStream www.edgestream.com	Provides disrupted video streaming applications over the public Internet	Provides video streaming over consumer cable or ADSL modem connections around the globe, even over paths that have 20 router hops between server and end user	EdgeStream video on-demand and IPTV Streaming software for video streaming
Limelight Networks www.limelightnetworks.com	Provides distributed on-demand and live delivery of video, music, games and download	Edge servers located in 72 locations around the world	Limelight ContentEdge for distributed content delivery via HTTP, Limelight MediaEdge Streaming for distributed video and music delivery via streaming, and Limelight Custom CDN for custom distributed delivery solutions
Mirror Image www.mirror-image.com	Provides content delivery, streaming media, Web computing and reporting services	Edge servers located in 22 countries	Global Content Caching, Extensible Rules Engine (XRE), Video On-Demand, and Live Webcasting

in stable operation for a significant period of time. Table 1.2 shows a list of four commercial CDNs and presents a brief summary of each of them. An updated listing of most of the existing commercial CDNs can be found in the research directories of Davison [25] and Pathan [49].

The proprietary nature of commercial CDNs makes it difficult to reveal detailed information about the technical and business strategies used by them. However, in the presented state-of-the-art survey of commercial CDNs, we provide information to significant details. In this context, it is worth mentioning that many CDN-specific information such as fees charged by CDNs, existing customers of CDNs are ignored since they are highly likely to change quickly over time. Therefore, the information provided in this section is expected to be stable and up-to-date. However, for readers' understanding on how a CDN charges its customers (i.e. CDN pricing strategies); we refer to Chap. 8 of the book, which outlines the pricing policies used for CDN services.

1.5.1.1 Akamai

Akamai technologies [1, 27] was founded in 1998 at Massachusetts, USA. It evolved out of an MIT research effort aimed at solving the flash crowd problem. Akamai is the market leader in providing content delivery services. It owns more than 25,000 servers over 900 networks in 69 countries [1]. Akamai's approach is based on the observation that serving Web content from a single location can present serious problems for site scalability, reliability and performance. Hence, a system is devised to serve requests from a variable number of cache servers at the network edge. Akamai servers deliver static (e.g. HTML pages, embedded images, executables, and PDF documents), dynamic content (e.g. animations, scripts, and DHTML), and streaming audio and video.

Akamai's infrastructure handles flash crowds by allocating more servers to sites experiencing high load, while serving all clients from nearby servers. The system directs client requests to the nearest available server likely to have the requested content. Akamai provides automatic network control through the mapping technique (i.e. the direction of request to content servers), which uses a dynamic, fault-tolerant DNS system. The mapping system resolves a hostname based on the service requested, user location, and network status. It also uses DNS for network load-balancing. Akamai name servers resolve hostnames to IP addresses by mapping requests to a server. Akamai agents communicate with certain border routers as peers; the mapping system uses BGP (Border Gateway Protocol) [56] information to determine network topology. The mapping system in Akamai combines the network topology information with live network statistics – such as traceroute data [39] – to provide a detailed, dynamic view of network structure, and quality measures for different mappings.

Akamai's DNS-based load balancing system continuously monitors the state of services and their servers and networks. To monitor the entire system's health end-to-end, Akamai uses agents that simulate the end user behavior by downloading Web objects and measuring their failure rates and download times. Akamai uses this information to monitor overall system performance and to automatically detect and suspend problematic data centers or servers. Each of the content servers frequently reports its load to a monitoring application, which aggregates and publishes

load reports to the local DNS server. That DNS server then determines which IP addresses (two or more) to return when resolving DNS names. If a certain server's load exceeds a certain threshold, the DNS server simultaneously assigns some of the server's allocated content to additional servers. If the server's load exceeds another threshold, the server's IP address is no longer available to clients. The server can thus shed a fraction of its load when it experiences moderate to high load. The monitoring system in Akamai also transmits data center load to the top-level DNS resolver to direct traffic away from overloaded data centers. In addition to load balancing, Akamai's monitoring system provides centralized reporting on content service for each customer and content server. This information is useful for network operational and diagnostic purposes.

Akamai delivers static and dynamic content over HTTP and HTTPS. Akamai content servers apply lifetime and other features (e.g. ability to serve secure content over HTTPS protocol, support alternate content, transfer encodings, and handle cookies) to the static content based on its type. Based on these attributes the edge server ensures the consistency of the content. On the other hand, Akamai handle dynamic content on the edge servers with the use of Edge Side Includes (ESI) [4] technology. The use of ESI enables the content providers to break their dynamic content into fragments with independent cacheability properties. These fragments can be maintained as separate objects in Akamai's edge servers and are dynamically assembled to a dynamic Web page in response to the end user requests.

Akamai supports Microsoft Windows Media, Real, and Apple's QuickTime format for delivering streaming services (live and on-demand media). A live stream is captured and encoded by the content provider and sent to the entry point server of a set of Akamai edge servers, which in turn serve content to the end users. In order to avoid all single points of failure, backups are maintained for the entry point server. Moreover, the entry point server sends data on multiple redundant paths to the edge servers through using information dispersal techniques.

More information on Akamai and its overlay routing, including its performance, availability benefits, different uses in live streaming, application, and IP acceleration can be found in Chap. 10 of this book.

1.5.1.2 EdgeStream

EdgeStream [23] was founded in 2000 at California, USA. It is a provider of video streaming applications over the public Internet. It provides video on-demand and IPTV streaming software to enable transportation of high bit rate video over Internet. It uses HTTP streaming for content delivery. EdgeStream supports different compression formats for delivering content. It has developed Continuous Route Optimization Software (CROS), Internet Congestion Tunnel Through (ICTT) and Real Time Performance Monitoring Service (RPMS) technologies, which together assist to address the latency, packet loss, and congestion bottlenecks. Embedded applications in Consumer Electronics Devices, wireless handheld devices, IP set top boxes, and advanced digital TV's can use the EdgeStream software for video streaming.

EdgeStream platform is made of client and server software modules. The server software consists of Content Management and Online Reporting (CMOR) Server Software Module, EdgeStream Control Server Software Module, EdgeStream Database System Module, and EdgeStream Streaming Server Module. All server modules may be combined to run on a single server, or run separately. CMOR module manages accounts, content, and all other servers in the system. It also generates Web-based real time reports for viewing statistics and transactions from a SQL database. The control module provides necessary authority to obtain the content location information along with streaming delivery management and logging functions. The database module maintains logs for accounting and billing purpose. It uses the Microsoft SQL 2000 Standard or Enterprise server software. The streaming server module is designed for load balancing and for running on standard low cost server platforms. When running on a dual processor server, streaming capacity can excess 500 Mbps with terabyte storage capacity.

EdgeStream client software provides a plug-in interface to the Windows Media and Real players. It can also be used to measure the Internet connection quality on a second by second basis. The PC client software is available for standard Windows platform and it is a 600 KB download. The Firmware client is a 300 KB (or smaller) download and can be either embedded in Windows XP or used with Windows CE.

1.5.1.3 Limelight Networks

Limelight Networks [30] was founded in 2001 at Tempe, Arizona, USA. Its content delivery services include HTTP/Web distribution of digital media files such as video, music, games, software and social media. It delivers content to media companies, including businesses operating in television, music, radio, newspaper, magazine, movie, video game, and software industries.

Content providers upload content either directly to the Limelight CDN's servers or to their own servers, which are connected directly to Limelight's network. Upon request from an end user, Limelight distributes that content to one or more Web server clusters which feed the specially configured servers at each content delivery location around the world. The content is then delivered directly to the end users either through ISPs or over the public Internet if appropriate. Like other commercial CDNs, it uses DNS redirection to reroute client requests to local clusters of machines, having built detailed maps of the Internet through a combination of BGP feeds and their own measurements, such as traceroutes from numerous vantage points.

Limelight Networks support Adobe Flash, MP3 audio, Microsoft Windows Media, Real, and Apple's QuickTime format for delivering on-demand streaming services. Limelight Networks proprietary software include Limelight ContentEdge for distributed content delivery via HTTP, Limelight MediaEdge Streaming for distributed video and music delivery via streaming, Limelight StorageEdge for storing customer's content library within Limelight's CDN architecture, and Limelight Custom CDN for custom distributed delivery solutions. Content providers using

Limelight's streaming services use Limelight User Exchange (LUX), which is a Web-based management and reporting console for tracking the end users' activity with real time reporting. All these software together assist in managing the content delivery system.

1.5.1.4 Mirror Image

Mirror Image [40] was founded in 1999 at Massachusetts, USA. It is a provider of online content, application, streaming media, Web computing, reporting, and transaction delivery services to the end users. It follows a Concentrated "Superstore" architecture, where content is placed in large Web server clusters in central locations close to densely populated user regions. Mirror Image exploits a global Content Access Point (CAP) infrastructure on top of the Internet to provide content providers, service providers, and enterprises with a platform for content delivery.

When a user request for content comes from a Mirror Image provisioned Web site, it is automatically routed to a global load balancer on the CAP network. The load balance uses DNS routing to determine the CAP location with fastest response time. Upon reception of the request at the selected CAP location, the caches and then the core databases are checked for the requested content. If the content is found, it is delivered to the user. On cache miss, the CAP network automatically returns a redirection status code "302" to the origin server's URL. Then the requested content is delivered to the user from the origin server and the CAP network retrieves (or pull) the content from the origin server and stores it for future subsequent requests.

Mirror Image provides content delivery, streaming media, and Web computing solutions, including Global Content Caching solution to offload traffic spikes while serving static content; Digital Asset Download solution to manage the storage and download of digital content; Video On-Demand solution for streaming delivery of digital content; Extensible Rules Engine (XRE) to give customers control over the delivery process; and Webcasting solution to allow users to send "one-to-many" messages for training, marketing, and distance learning outlets.

1.5.2 Academic CDNs

Unlike commercial CDNs, the use of P2P technologies is mostly common in academic CDNs. Thus, the content delivery follows a decentralized approach and request load is spread across all the participating hosts, and thus the system can handle node failures and sudden load surges. Academic CDNs built using P2P techniques are effective for static content only and therefore, are unable to handle dynamically generated content due to the uncachable nature of dynamic content. In this section, we present three representative academic CDNs, namely CoDeeN, Coral, and Globule. Table 1.3 provides a brief summary of these academic CDNs. Two other academic CDNs – FCAN (adaptive CDN for alleviating flash crowds) and COMODIN

Table 1.3 Summary of the existing academic CDNs

CDN Name	Description	Service Type	Implementation and Testing	Availability
CoDeeN www.codeen.cs. princeton. edu	CoDeeN is an academic testbed CDN built on top of PlanetLab	Provides caching of content and redirection of HTTP requests	Implemented in C/C++ and tested on Linux(2.4/2.6) and MacOS(10.2/10.3)	N/A
Coral www.coralcdn.org	Coral is a free P2P CDN. It is hosted on PlanetLab	Provides content replication in proportion to the content's popularity	Implemented in C++ and tested on Linux, OpenBSD, FreeBSD, and Mac OS X	No official release yet. Coral is a Free software, licensed under GPLv2 (GNU General Public License)
Globule www.globule.org	Globule is an open source collaborative CDN	Provides replication of content, monitoring of servers and redirecting client requests to available replicas	Implemented using PHP scripting, C/C++ and tested on Unix/Linux and Windows	Globule is open source, subject to a BSD-style license and the Apache software license for the packaged Apache HTTP server

(streaming CDN for collaborative media streaming services), are presented respectively in Chap. 11 and Chap. 12 of this book.

1.5.2.1 CoDeeN

CoDeeN [46, 64] is a P2P-based proxy server system developed at Princeton University, USA. It is an HTTP-based CDN, which gives *participating* users better performance to *most* Web sites. CoDeeN provides caching of Web content and redirection of HTTP requests. It is built on top of PlanetLab [9], consisting of a network of high performance proxy servers. CoDeeN nodes are deployed as "open" proxies in order to allow access from outside the hosting organization. Each CoDeeN node is capable of acting as a forward proxy, a reverse proxy, and a redirector. CoDeeN operates in the following way: (1) users set their internet caches to a nearby high bandwidth proxy that participates in the CoDeeN system; (2) the CoDeeN node acts as a forward proxy and tries to satisfy the request locally. On cache miss, the redirector logic built in the CoDeeN node determines where the request should be sent. For most requests the redirector take into account request locality, system load, reliability, and proximity information to forward the requests to other CoDeeN nodes, which act as a reverse proxy for the forwarded requests. Requests which are still not satisfied at this stage are sent to the origin server.

CoDeeN has the local monitoring ability that examines the service's primary resources, such as free file descriptors/sockets, CPU cycles, and DNS resolver service. It gathers information about the CoDeeN instance's state and its host environment. This information assists in assessing resource connection as well as external service availability. To monitor the health and status of the peers, each CoDeeN node employs two mechanisms – a lightweight UDP-based heartbeat and a "heavier" HTTP/TCP-level "fetch" helper [64]. In the first case, each proxy sends a heartbeat message once per second to one of its peers, which then responds (heartbeat acknowledgement or ACK) with piggybacked load information including peer's average load, system time, file descriptor availability, proxy and node uptimes, average hourly traffic, and DNS timing/failure statistics. By coupling the history of ACKs with their piggybacked local status information, each CoDeeN instance independently assesses the health of other nodes. In the later case, each CoDeeN node is employed with a toll to specify what fails when it can not retrieve a page within the allotted time. A history of failed fetches for each peer is maintained, which in combination with UDP-level heartbeats assists in determining if a node is viable for request redirection.

A number of projects are related to CoDeeN – CoBlitz (a scalable Web-based distribution system for large files), CoDeploy (an efficient synchronization tool for PlanetLab slices), CoDNS (a fast and reliable name lookup service), CoTop (a command line activity monitoring tool for PlanetLab), CoMon (a Web-based slice monitor that monitors most PlanetLab nodes), and CoTest (a login debugging tool).

A significant application service running on top of CoDeeN is CoBlitz [48]. It is a file transfer service which distributes large files without requiring any modifications

to standard Web servers and clients, since all the necessary support is located on CoDeeN itself. One of the motivations for building CoBlitz on top of CoDeeN is to ensure long duration caching so that client requested content can be served quickly even after demand for it drops. CoBlitz is publicly accessible which allows the clients to prepend the original URL with "`http://coblitz.codeen.org:3125`" and fetch it like any other URL. A customized DNS server maps the name coblitz.codeen.org to a nearby PlanetLab node. To deploy CoBlitz, the HTTP CDN, CoDeeN is made amenable to handling large files. This approach includes modifying large file handling to efficiently support them on CoDeeN and modifying its request-routing to enable more swarm-like behavior under heavy load. In CoBlitz, a large file is considered as a set of small files (chunks) that can be spread across the CDN. CoBlitz works if the chunks are fully cached, partially cached, or not at all cached, fetching any missing chunks from the origin as needed. Thus, while transferring large files over CoBlitz, no assumptions are made about the existence of the file on the peers.

1.5.2.2 Coral

Coral [28] is a free P2P content distribution network. It was developed by the New York University's Secure Computer Systems group during their visit to Stanford University, USA. It is designed to mirror Web content and its goal is to give most *users* better performance to *participating* Web sites. It uses the bandwidth of volunteers to avoid flash crowd and to reduce the load on Web sites and other Web content providers in general. CoralCDN is deployed on PlanetLab, instead of third party volunteer systems. To use CoralCDN, a content publisher, end-host client, or someone posting a link to a high-traffic portal has to append "`.nyud.net:8090`" to the hostname in a URL. Clients are redirected to the nearby Coral Web caches transparently through DNS redirection. Coral Web caches cooperate to transfer data from nearby peers whenever possible, minimizing both the load on the origin Web server and the latency perceived by the user. CoralCDN is built on top of the Coral key-value indexing layer. It allows nodes to access nearby cached objects without redundantly querying more distant nodes. It also prevents the creation of hotspots in the indexing infrastructure, even under degenerate loads.

CoralCDN is comprised of three main parts: a network of cooperative HTTP proxies for handling client requests; a network of DNS nameservers for ".nyud.net" that map clients to nearby CoralCDN HTTP proxy; and an underlying indexing infrastructure and clustering machinery on which the first two applications rely.

Coral uses an indexing abstraction called Distributed Sloppy Hash Table (DSHT), which is a variant of Distributed Hash Tables (DHTs) for building key value indexes. DSHTs are designed for applications storing soft-state key-value pairs, where multiple values may be stored under the same key. A DSHT caches key-value pairs at nodes whose identifiers are increasingly close to the key being referenced, as an inverse function of load. It has a "sloppy" storage technique that leverages cross-layer interaction between the routing and storage layers.

The CoralHTTP proxy satisfies HTTP requests for Coralized URLs. To minimize the load on the origin servers, a CoralHTTP proxy fetch Web pages from other proxies whenever possible. Each proxy keeps a local cache to fulfill requests immediately. If a CoralHTTP proxy discovers the requested content in one or more other proxies, it establishes parallel TCP connections to them (multiple other proxies) and issues an HTTP request to the first proxy to which it successfully connects. Once the neighboring proxy begins to send valid content, all other established TCP connections are closed. When a client requests content from a non-resident URL, CoralHTTP proxy first attempts to locate a cached copy. If the Coral indexing layer does not provide any referral or any of its referrals return the requested content, CoralHTTP proxy fetches the content directly from the origin server. In the face of a flash crowd, all CoralHTTP proxies naturally form a kind of "multicast tree" for retrieving the Web page, instead of making simultaneous requests to the origin server and data flows from the proxy that initially fetch the content from the origin server to those arriving later. Such behavior in CoralCDN is provided by combining optimistic references and cut-through routing.

The CoralDNS server maps the IP addresses to the hostnames of Coralized URLs and returns it to CoralHTTP proxies. Coral's architecture is based on clusters of well-connected machines. Clusters are exposed in the interface to higher-level software, and in fact form a crucial part of the DNS redirection mechanism. In order to improve locality, when a DNS resolver contacts a nearby CoralDNS server instance, the CoralDNS server returns the proxies within an appropriate cluster and ensures that future DNS requests form this client does not leave the cluster. A CoralDNS server only returns the CoralHTTP proxy addresses which is has recently verified first hand in order to check a proxy's liveness status synchronously prior to replying to a DNS query.

1.5.2.3 Globule

Globule [52] is an open-source collaborative CDN developed at the Vrije Universiteit in Amsterdam, the Netherlands. It aims to allow Web content providers to organize together and operate their own world-wide hosting platform. In particular, it is an overlay network composed of end user nodes that operate in a P2P fashion across a wide-area network, where participating members offer resources such as storage capacity, bandwidth, and processing power. It provides replication of content, monitoring of servers and redirection of client requests to available replicas.

In Globule, a site is defined as a collection of documents that belong to one specific user (the site's owner) and a server is a process running on a machine connected to a network, which executes an instance of the Globule software. Each site is composed of the origin, backup, replica, and redirector servers. The origin server(s) has the authority to contain all documents of the site and has the responsibility to distribute content among other involved servers. The backup servers maintain a full up-to-date copy of the hosted site. Other than backup servers, a number of replica servers can be used to host a site. While backup servers just maintain a copy, replica

servers aim to maximize performance based on the request load and QoS requirements. A replica server for a site is typically operated by a different user than the origin and a replica server typically contain a partial copy of the hosted site. One can view the replica server as a caching proxy which fetches the content from the origin server on a local cache miss. A redirector server is responsible for redirecting client requests to the optimal replica server for serving a given request. Redirectors in Globule can use either HTTP or DNS-based redirection. A redirector also implements a policy for client redirection. The default policy redirects clients to the closest replica in terms of estimated latency. Redirectors also monitor the availability of other servers to ensure effective redirection of requests. Depending on the role a server can perform as origin, replica, backup and/or redirector servers.

Globule takes inter-node latency as the proximity measure. This metric is used to optimally place replicas to the clients, and to redirect the clients to an appropriate replica server. Globule is implemented as a third-party module for the Apache HTTP Server that allows any given server to replicate its documents to other Globule servers. To replicate content, content providers only need to compile an extra module into their Apache server and edit a simple configuration file.

1.6 Visionary Thoughts for Practitioners

We envision the following technological evolutions to be realized in the coming years in CDN industry related research.

1.6.1 A Unified Content Network

To make content transformations and processing and infrastructure service accessible by the user, vendors have implemented Content Service Networks (CSN) [38], which act as another network infrastructure layer built upon CDNs and provide next generation of CDN services. CSN appears to be a variation of the conventional CDN. Network resources provided by a CSN is used as a "service" distribution channel for value added service providers in order to make their applications as an infrastructure service. This logical separation between content and services under the "Content Delivery/Distribution" and "Content Services" domain, is undesirable considering the on-going trend in content networking. Hence, a unified content network, which supports the coordinated composition and delivery of content and services, is highly desirable.

1.6.2 Dynamic Content

Dynamic content refers to the content which is generated on-demand using Web applications based on user requests. Such content generation is customized depending

on a given user profile and characteristics. A large amount of Web content is generated dynamically. Dynamic content includes scripts, animations, DHTML or XML pages that are generated on the fly based on user specification. The dynamic generation of Web pages can be performed with the use of scalable Web application hosting techniques such as edge computing [55], context-aware data caching [20, 58], data replication [58], and content blind data caching [58]. Instead of replicating the dynamic pages generated by a Web server, these techniques aim to replicate the means of generating pages over multiple edge servers [58]. Commercial CDN providers have their own proprietary solutions and application server platform to handle dynamic content. EdgeSuite content distribution from Akamai and IBM WebSphere edge services [5] are examples of systems to provide usage-based application and (dynamic) content delivery. In order to manage dynamic content, a CDN provider may use such scalable techniques to accelerate the dynamic generation of Web pages. The choice of the appropriate strategy may vary depending on the characteristics of Web applications.

1.6.3 Web Services

Nowadays, a few commercial CDNs host Web services. For instance, Akamai has deployed .NET services on its network. Mirror Image has also developed an Application Delivery Network (ADN) that hosts both .NET and J2EE applications at its edge servers. Several studies [29, 60] have shown that the performance of Web services is relatively poor because of the requirements for processing and special hosting capability. Some solutions can be found in literature, which can be used to address the problem of effective replication of Web services to the CDN edge servers. Geng et al. [29] propose a sharable and tradable cache infrastructure between several ISPs and networks. This solution is characterized by a capacity provisioning network (CPN) for trading cache capacities. A CPN is operated by a trading hub rather than being operated by a particular CDN. Such a solution can be followed by a CDN to acquire (through trading) necessary capacity to meet the demand for Web service caching. Takase et al. [60] present caching using XML messages, improvements by caching event sequences of the XML parser. They also propose caching of application objects using Java serialization, reflection copy, and clone copy.

1.6.4 Service-Oriented Architecture

Future trends in content networking domain are expected to allow services to be composed of other services by building on standard protocols and invocation mechanisms. Thus, content networks should be capable of exploiting an SOA. High-level transparency within SOA is required, which could have impact on all the constituent technologies. Content management in such an SOA-based CDN is expected to be

highly motivated by user preferences. Hence, a comprehensive model for managing the distributed contents and services in future CDN would be crucial to avail end user's preferences. To address this issue, contents can be personalized to meet specific user's (or a group of users) preferences. Like Web personalization [41], user preferences can be automatically learned from content request and usage data by using data mining techniques. Data mining over content network can exploit significant performance improvement through dealing with proper management of traffic, pricing and accounting/billing in SOA-based CDNs.

1.7 Future Research Directions

In this section, we provide a roadmap to the academic CDN researchers by exploring possibilities and research challenges that are expected to drive innovations within this domain.

1.7.1 Load Balancing and Content replication in Cooperative Domain

The issue of effective replication and caching of content is critical to the success of traditional as well as cooperative arrangement of CDNs. The concept of caching "hot" content is not new, but in the context of a cooperative content delivery, there will be significant competing considerations. Future research should lead to the outcome of dynamic, scalable, and efficient replication mechanisms that cache content on demand with respect to the locality of requests, focusing on regions where specific content is needed most. Moreover, innovative solutions integrating replication and caching are expected in the management of dynamic and personalized content in the cooperative domain. Chapter 3 provides more information on such innovative content replication techniques. Detailed information on the integrated use of caching and replication as well as cache consistency mechanisms can be found in Chap. 4 and Chap. 5 of this book.

1.7.2 Deployment of Market Mechanisms

An economic model can exploit the dynamism of the CDN market and makes the system more manageable through analyzing the emergent marketplace behavior. This also provides benefits to the CDNs to offer their resources and to open up many interesting new services. Deployment of the market mechanisms can be done based on an SOA. In addition, replication, resource sharing, and load balancing polices need to be guided by profit-driven utility functions that satisfy QoS requirements of end users. More information on economics-informed design of CDNs and pricing of CDNs can be found in Chap. 7 and Chap. 8 respectively.

1.7.3 An Adaptive CDN for Media Streaming

Hosting of on-demand media streaming service is challenging because of the enormous network and bandwidth required to simultaneously deliver large amount of content to end users. To avoid network congestion and to improve performance, P2P techniques can be used to build an adaptive CDN. In such a system, content storage and workload from streaming server, network, and storage resources are offloaded to the end users' workstations. The fundamental idea is to allow multiple subscriber peers to serve streams of the same video content simultaneously to a consuming peer rather than the traditional single-server-to-client streaming model, while allowing each peer to store only a small portion of the content. Such a solution for cost-effective media streaming using a P2P approach has been reported in the design of the Decentralized Media Streaming Infrastructure (DeMSI) [65]. Another work on open and adaptive streaming CDN through collaborative control on media streaming services is described in Chap. 12 of this book.

1.7.4 A Mobile Dynamic CDN

Mobile networks are becoming increasing popular for distributing information to a large number of highly dynamic users. In comparison to wired networks, mobile networks are distinguished by potentially much higher variability in demand due to user mobility. Content delivery techniques for mobile networks must take into account potentially very high spatial and temporal demand variations to dynamically reconfigure the system, and to minimize the total traffic over the network backbone. A model for mobile dynamic CDN should be designed to allow the access of accurate and up-to-date information and enterprise applications. Such a mobile dynamic CDN model for enterprise networks and related content management policies are presented by Aioffi et al. [12]. Example of a commercial mobile CDN provider is Ortiva Wireless [8], which delivers audio, video, and multimedia content to mobile users. More information on information dissemination in mobile CDNs can be found in Chap. 14 of this book.

1.7.5 Content Distribution Through Internetworking/Peering/Brokering

Present trends in content networks and content networking capabilities give rise to the interest in interconnecting content networks. High quality service could be achieved by permitting CDNs to cooperate and thereby providing a means for CDNs to redistribute content delivery between themselves. Such cooperation could reach to a large client population that one CDN cannot achieve otherwise. Therefore, future research will heavily focus on the innovation of technologies for

internetworking, brokering or peering arrangements between CDNs [18, 26, 50]. More information on CDN internetworking can be found in Chap. 16 of this book.

1.8 Conclusion

In this chapter, we present a state-of-the-art survey of the existing CDNs and give an insight into the underlying technologies that are currently in use in the content-distribution space. After analyzing the ongoing content networking trend, we can anticipate the integrated uses of existing emerging as well as stable technologies (e.g. agent, P2P, grid, data mining) to augment the effectiveness and boost the efficiency of future CDN infrastructures. We also perceive that there is a possible shift change in the CDN industry as CDN internetworking, adaptive CDNs, mobile CDNs, and to the full, community-based CDNs are evolving. Therefore, this chapter can be used as a basis to provide an in-depth analysis and complete understanding of the current and future trends in the content distribution landscape.

References

1. Akamai Technologies, 2007. www.akamai.com
2. BioGrid Project, Japan, 2005. http://www.biogrid.jp
3. Broadband Service Forum, 2007. http://broadbandservicesforum.org
4. ESI Developer Resources, 2007. http://www.akamai.com/html/support/esi.html
5. IBM WebSphere Application Server, 2007. http://www-306.ibm.com/software/webservers/appserv/was/
6. ICAP Forum, 2007. http://www.i-cap.org/
7. Internet Streaming Media Alliance, 2007. http://www.isma.tv/
8. Ortiva Wireless, 2007. http://www.ortivawireless.com/
9. PlanetLab Consortium, 2007. http://www.planet-lab.org/
10. Wikipedia. September 11, 2001 attacks. http://en.wikipedia.org/wiki/September_11,_2001_attack
11. Adler, S. The slashdot effect: an analysis of three Internet publications. *Linux Gazette Issue*, 38, 1999.
12. Aioffi, W. M., Mateus, G. R., de Almeida, J. M., and Loureiro, A. A. F. Dynamic content distribution for mobile enterprise networks. *IEEE Journal on Selected Areas in Communications*, 23(10), pp. 2022–2031, 2005.
13. Androutsellis-Theotokis, S. and Spinellis, D. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys*, 36(4), ACM Press, NY, USA, pp. 335–371, 2004.
14. Arlitt, M. and Jin, T. A workload characterization study of 1998 world cup Web site. *IEEE Network*, pp. 30–37, 2000.
15. Barbir, A., Batuner, O., Beck, A., Chan, T., and Orman, H. Policy, authorization, and enforcement requirements of the open pluggable edge services (OPES). Internet Engineering Task Force RFC 3838, 2004. www.ietf.org/rfc/rfc3838.txt
16. Barbir, A., Penno, R., Chen, R., Hofmann, H., and Orman, H. An architecture for open pluggable edge services (OPES). Internet Engineering Task Force RFC 3835, 2004. www.ietf.org/rfc/rfc3835.txt

17. Bartolini, N., Casalicchio, E., and Tucci, S. A walk through content delivery networks. In *Proc. of the 11th Annual Meeting of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, LNCS Vol. 2965/2004, pp. 1–25, April 2004.
18. Biliris, A., Cranor, C., Douglis, F., Rabinovich, M., Sibal, S., Spatscheck, O., and Sturm, W. CDN brokering. *Computer Communications*, 25(4), pp. 393–402, 2002.
19. Brussee, R., Eertink, H., Huijsen, W., Hulsebosch, B., Rougoor, M., Teeuw, W., Wibbels, M., and Zandbelt, H. Content distribution network state of the art," Telematica Instituut, 2001.
20. Buchholz, T., Hochstatter, I., and Linnhoff-Popien, C. A profit maximizing distribution strategy for context-aware services. In *Proc. of 2nd International Workshop on Mobile Commerce and Services (WMCS'05)*, pp. 144–153, 2005.
21. Ceri, S. and Pelagatti, G. *Distributed Databases: Principles and Systems*, McGraw-Hill, NY, 1984.
22. Chervenak, A., Foster, I., Kesselman, C., Salisbury, C., and Tuecke, S. The data grid: towards an architecture for the distributed management and analysis of large scientific datasets. *Journal of Network and Computer Applications*, 23, pp. 187–200, 2001.
23. Chung, R. Network latency and its effect on video streaming. EdgeStream, 2004. www.edgestream.com
24. Cooper, I., Melve, I., and Tomlinson, G. Internet Web replication and caching taxonomy. Internet Engineering Task Force RFC 3040, 2001.
25. Davison, B. D. Web caching and content delivery resources. http://www.web-caching.com, 2007.
26. Day, M., Cain, B., Tomlinson, G., and Rzewski, P. A model for content internetworking (CDI). Internet Engineering Task Force RFC 3466, 2003.
27. Dilley, J., Maggs, B., Parikh, J., Prokop, H., Sitaraman, R., and Weihl, B. Globally distributed content delivery. *IEEE Internet Computing*, pp. 50–58, 2002.
28. Freedman, M. J., Freudenthal, E., and Mazières, D. Democratizing content publication with Coral. In *Proc. of 1st USENIX/ACM Symposium on Networked Systems Design and Implementation*, San Francisco, CA, USA, 2004.
29. Geng, X., Gopal, R. D., Ramesh, R., and Whinston, A. B. Scaling web services with capacity provision networks. *IEEE Computer*, 36(11), pp. 64–72, 2003.
30. Gordon, M. The Internet streaming media boom: a powerful trend that represents fundamental change. Limelight Networks, 2007. www.limelightnetworks.com
31. Hofmann, M. and Beaumont, L. R. *Content Networking: Architecture, Protocols, and Practice*. Morgan Kaufmann Publishers, San Francisco, CA, USA, pp. 129–134, 2005.
32. International Standards Organization (ISO), Open systems interconnection–basic reference model. ISO 7498, 1989.
33. Izal, M., Urvoy-Keller, G., Biersack, E. W., Felber, P., Hamra, A. A., and Garces-Erice, L. Dissecting bittorrent: five months in a torrent's lifetime. In *Proc. of 5th Annual Passive and Active Measurement Workshop (PAM'2004)*, Antibes Juan-Les-Pins, France, 2004.
34. Jung, J., Krishnamurthy, B., and Rabinovich, M. Flash crowds and denial of service attacks: characterization and implications for CDNs and Web sites. In *Proc. of the International World Wide Web Conference*, pp. 252–262, 2002.
35. Kangasharju, J., Roberts, J., and Ross, K. W. Object replication strategies in content distribution networks. *Computer Communications*, 25(4), pp. 367–383, 2002.
36. Lazar, I. and Terrill, W. Exploring content delivery networking. *IT Professional*, 3(4), pp. 47–49, 2001.
37. Lebrun, P. The large hadron collider, a megascience project. In *Proc. of the 38th INFN Eloisatron Project Workshop on Superconducting Materials for High Energy Colliders*, Erice, Italy, 1999.
38. Ma, W. Y., Shen, B., and Brassil, J. T. Content services network: architecture and protocols. In *Proc. of 6th International Workshop on Web Caching and Content Distribution (IWCW6)*, 2001.
39. Malkin, G. Traceroute using an IP option. Internet Engineering Task Force RFC 1393, 1993.

40. Mirror Image Internet. Content Delivery and the Mirror Image Adaptive CAP Network, 2007. www.mirror-image.com
41. Mobasher, B., Cooley, R., and Srivastava, J. Automatic personalization based on Web usage mining. *Communications of the ACM*, 43(8), pp. 142–151, 2000.
42. Molina, B., Palau, C. E., and Esteve, M. Modeling content delivery networks and their performance. *Computer Communications*, 27(15), pp. 1401–1411, 2004.
43. Moore, R., Prince, T. A., and Ellisman, M. Data intensive computing and digital libraries. *Communications of the ACM*, 41(11), ACM Press, NY, USA, pp. 56–62, 1998.
44. Oram, A. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*, O'Reilly & Associates, Inc., Sebastopol, CA, 2001.
45. Ozsu, M. T. and Valduriez, P. *Principles of Distributed Database Systems*, Prentice-Hall, Inc., Upper Saddle River, NJ, 1999.
46. Pai, V. S., Wang, L., Park, K. S., Pang, R., and Peterson, L. The dark side of the web: an open proxy's view. In *Proc. of the Second Workshop on Hot Topics in Networking (HotNets-II)*, Cambridge, MA, USA, 2003.
47. Pallis, G. and Vakali, A. Insight and perspectives for content delivery networks. *Communications of the ACM*, 49(1), ACM Press, NY, USA, pp. 101–106, 2006.
48. Park, K. S. and Pai, V. S. Scale and performance in the CoBlitz large-file distribution service. In *Proc. of the 3rd Symposium on Networked Systems Design and Implementation (NSDI 2006)*, San Jose, CA, USA, 2006.
49. Pathan, M. Content delivery networks (CDNs) research directory, 2007. http://www.gridbus.org/cdn/CDNs.html
50. Pathan, M., Broberg, J., Bubendorfer, K., Kim, K. H., and Buyya, R. An Architecture for Virtual Organization (VO)-Based Effective Peering of Content Delivery Networks, UPGRADE-CN'07. In *Proc. of the 16th IEEE International Symposium on High Performance Distributed Computing (HPDC)*, CA, USA, 2007.
51. Peng, G. CDN: Content distribution network. Technical Report TR-125, Experimental Computer Systems Lab, Department of Computer Science, State University of New York, Stony Brook, NY, 2003. http://citeseer.ist.psu.edu/peng03cdn.html
52. Pierre, G. and van Steen, M. Globule: a collaborative content delivery network. *IEEE Communications*, 44(8), 2006.
53. Plagemann, T., Goebel, V., Mauthe, A., Mathy, L., Turletti, T., and Urvoy-Keller, G. From content distribution to content networks – issues and challenges. *Computer Communications*, 29(5), pp. 551–562, 2006.
54. Rabinovich, M. and Spatscheck, O. *Web Caching and Replication*, Addison Wesley, USA, 2002.
55. Rabinovich, M., Xiao, Z., Douglis, F., and Kalmanek, C. Moving edge side includes to the real edge – the clients. In *Proc. of USENIX Symposium on Internet Technologies and Systems*, Sealttle, Washington, USA, 2003.
56. Rekhter, Y. and Li, T. A border gateway protocol 4. Internet Engineering Task Force RFC 1771, 1995.
57. Saroiu, S., Gummadi, K. P., Dunn, R. J., Gribble, S. D., and Levy, H. M. An analysis of Internet content delivery systems. *ACM SIGOPS Operating Systems Review*, 36, pp. 315–328, 2002.
58. Sivasubramanian, S., Pierre, G., Van Steen, M., and Alonso, G. Analysis of caching and replication strategies for Web applications. *IEEE Internet Computing*, 11(1), pp. 60–66, 2007.
59. Szalay, A. and Gray, J. The world-wide telescope. *Science* 293(5537), pp. 2037–2040, 2001.
60. Takase, T. and Tatsubori, M. Efficient Web services response caching by selecting optimal data representation. In *Proc. of 24th International Conference on Distributed Computing Systems (ICDCS 2004)*, pp. 188–197, 2004.
61. Vakali, A. and Pallis, G. Content delivery networks: status and trends. *IEEE Internet Computing*, 7(6), IEEE Computer Society, pp. 68–74, 2003.
62. Venugopal, S., Buyya, R., and Ramamohanarao, K. A taxonomy of data grids for distributed data sharing, management, and processing. *ACM Computing Surveys*, 38(1), ACM Press, NY, USA, 2006.

63. Verma, D. C. *Content Distribution Networks: An Engineering Approach*, John Wiley & Sons, Inc., New York, USA, 2002.
64. Wang, L., Park, K. S., Pang, R., Pai, V. S., and Peterson, L. Reliability and security in CoDeeN content distribution network. In *Proc. of the USENIX 2004 Annual Technical Conference*, Boston, MA, USA, 2004.
65. Yim, A. and Buyya, R. Decentralized media streaming infrastructure (DeMSI): an adaptive and high-performance peer-to-peer content delivery network. *Journal of Systems Architecture*, 52(12), Elsevier, The Netherlands, pp. 737–772, 2006.