Giovanni Giambene

Queuing Theory and Telecommunications

Networks and Applications

Second Edition

EXTRA MATERIALS
extras.springer.com

Springer

Contents (page numbers are not final ones)

Pre-publication version

EXTRACT FROM PART I

Chapter 3

IP-BASED NETWORKS AND FUTURE TRENDS

3.1 Introduction

A growing number of people are using the Internet, the network of the networks; this is also evident from the different bandwidth-intensive applications supported by Internet and by the considerable number of Internet books, video, etc. that have become available during these years. The widespread diffusion of social networks (Facebook, YouTube, etc.), peer-to-peer traffic, and cloud applications have further contributed to the impressive growth in the Internet use. IP traffic has globally grown eight times in the period 2008-2012 (five years) and is expected to increase threefold in the next three years. The annual global IP traffic will surpass the Zettabyte (i.e., 10^{21} bytes) threshold by the end of 2016 [1]. This Chapter focuses on the protocols and the network technologies to support Internet traffic.

3.2 The Internet

J. C. R. Licklider of the Massachusetts Institute of Technology (MIT) proposed a global network of computers in 1962 and moved to the Defense Advanced Research Projects Agency (DARPA) to lead a project to

interconnect Department of Defense (DoD) sites in the United States of America. L. Kleinrock of MIT (and, later, University of California, Los Angeles, UCLA) developed the theory of packet-switching, which is at the basis of Internet traffic. In 1965, L. Roberts of MIT connected a Massachusetts computer with a California computer by means of a dial-up telephone line. He showed the feasibility of wide area networking, but also that the telephone circuit-switching was inadequate for this traffic, thus confirming the importance of the Kleinrock packet-switching theory. These pioneers (as well as other people) are the actual founders of the Internet. The Internet, then known as ARPANET, was brought online in 1969, initially connecting four major sites (computers), under a contract held by the renamed Advanced Research Projects Agency (ARPA).

Once the initial sites were installed, representatives from each site met together to solve the technical problems concerning the interconnection of hosts by means of protocols. A working group, called Network Working Group (NWG), was in charge of defining the first 'rules' (i.e., protocols) of the network. The open approach adopted by the first NWG meeting continued in a more formalized way by using meeting notes, called Request For Comments (RFC). These documents are intended to keep members updated on the status of several things concerning Internet protocol. They were also used to receive responses from researchers.

The Internet was designed to provide a communication network able to work even if some sites are destroyed. The early Internet was used by computer experts, engineers, scientists, and librarians. There were no personal computers and no massive use in those days. Different 'initial' applications and protocols were conceived to exploit ARPANET. E-mail was adopted for ARPANET in 1972. The telnet protocol, allowing us to log on a remote computer, was defined in 1972 [2]. The FTP protocol, enabling file transfers between Internet sites, was published as RFC 354 in 1972 [3],[4] and from then further RFCs were made available to update the characteristic of the FTP protocol. RFCs are today the method used to standardize every aspect of the Internet; they are freely accessible in the ASCII format through the Internet Engineering Task Force (IETF) Web site [5]. RFCs are approved after a very strong review process. IETF is an open, all-volunteer organization (started its activities in 1983), with no formal membership nor membership requirements. It is divided into a large number of working groups, each dealing with a specific Internet issue.

In 1974, a new suite of protocols was proposed and implemented in the ARPANET, based on the Transmission Control Protocol (TCP) for end-to-end communications. In 1978, a new Internet design approach was conceived with the division of tasks between two protocols:

- The new Internet Protocol (IP) for routing packets and device-to-device communications (i.e., host-to-gateway or gateway-to-gateway);
- The TCP protocol for reliable, end-to-end communications.

Since TCP and IP were originally conceived as working in tandem, this protocol suite is commonly denoted as TCP/IP. The original versions of both TCP and IP were written in 1981 [6],[7].

As long as the number of Internet sites was small, it was easy to keep track of the resources of interest that were available. But as more and more universities and organizations connected, the Internet became harder to track. There was the need for tools to index the available resources. Starting from 1989, significant efforts were pursued in this direction. In particular, T. Berners-Lee and others at the European Laboratory for Particle Physics (i.e., CERN) laid the basis to share documents using *browsers* in a multi-platform environment. In particular, three new technologies were incorporated into his proposal: (i) the HyperText Markup Language (HTML) used to write documents (also named 'pages') for the Internet; (ii) the HyperText Transfer Protocol (HTTP), an application layer protocol to transmit documents in HTML format; (iii) a browser client software program to receive and interpret HTML documents and to display the results. His proposal was based on *hypertext*, i.e., a system of embedding links, that is Internet addresses, in the text to refer to other documents Internet documents.

In 1991, the World Wide Web was born because the first really friendly interface to the Internet was developed at the University of Minnesota; it was named 'gopher', after the University of Minnesota mascot, the golden gopher. In 1993, the development of the graphical browser called Mosaic, by M. Andreessen and his team at the National Center For Supercomputing Applications (NCSA), a research institute at the University of Illinois, gave a strong boost to the Web. Starting from this browser, new ones rapidly spread and made the Web a worldwide success. Further developments to the Web were represented by the Web search engines, as already discussed in Chapter 1 (Section 1.1).

3.2.1 Introduction to the Internet protocol suite

The goal of TCP/IP was to interconnect different physical networks to form what appears to the user as a universal network. Such a set of interconnected networks is called an *Internet* [8]-[11]. Communication services are provided by Internet protocols, which operate between the link

layer and the application one. The architecture of the physical networks is hidden to the users.

To be able to interconnect two networks, we need a 'computer' that is attached to both networks and that can forward packets from one network to another and vice versa; this device, called *router*, has two important characteristics:

- From the network standpoint, a router is a normal host.
- From the user standpoint, routers are invisible; the user sees only a larger internetwork.

Each host has an address assigned, the *IP address*, to identify it in the Internet. When a host has multiple network adapters, each adapter has a separate IP address.

3.2.2 TCP/IP protocol architecture

Although there is no universal agreement on how to describe TCP/IP with a layered model, it is generally regarded as being composed of fewer layers than the seven layers of the classical OSI model. Most descriptions of TCP/IP define three to five functional levels in the protocol architecture [12]; a four-layer TCP/IP model is shown in Figure 3.1.

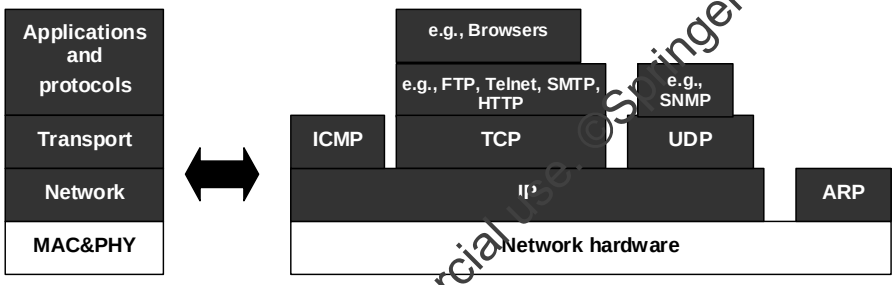

Figure 3-1. Simplified Internet protocol suite. The acronyms in this figure will be described along this Chapter; this figure will be taken as a reference.

As in the OSI model, data are passed down through the stack when they are sent to the network, and passed up through the stack when they are received from the network. Each layer treats the information it receives from the layer above as *data* and adds its own *header* in front of that information

to ensure the proper management of these data. The operation to add the header (containing control information) is called *encapsulation*.

The *network layer* is the lowest layer of the TCP/IP protocol hierarchy. The protocols of this layer provide the means to route data to other network devices. Unlike higher-level protocols, network layer protocols must know the details of the underlying network (its packet structure, addressing, etc.) to correctly format the data being transmitted to comply with local network constraints.

The Internet protocol stack has a layered architecture resembling an *hourglass* (see Figure 3.2): the reason for this denomination of the Internet protocol model is that there are many PHY and MAC layer protocols and there are many application and transport layer protocols, while on the waist of the hourglass at the network layer there are very few protocols, basically the IP protocol. The hourglass model expresses the concept that the IP protocol is the glue, the basic building block of the Internet. The protocols of the waist are those to which we are referring mainly when talking about the Internet 'ossification'; this is seen today mostly as a limit to the flexibility and security, because all information is forced through a small set of mid-layer protocols.

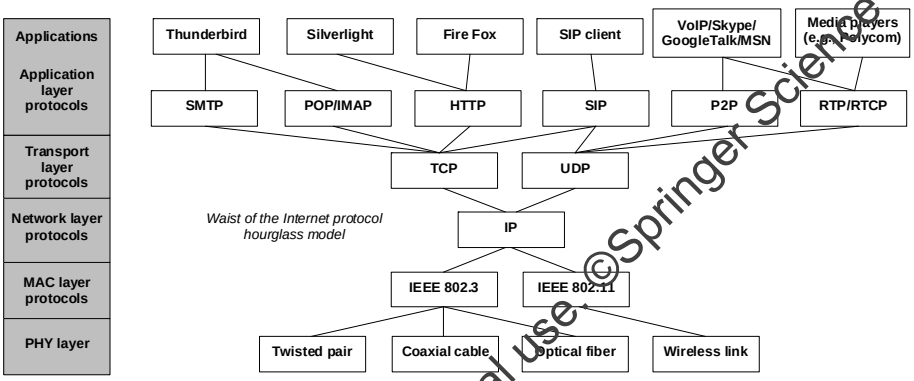

Figure 3-2. The Internet protocol stack and the hourglass model (note that not all the protocols have been shown at the different layers, but just some of them).

The Internet Protocol (IP) originally defined in RFC 791 [6] is the heart of the Internet protocol suite and the most important protocol of the network layer. IP provides the basic packet delivery service for the networks. All the higher-layer protocols (and the related data flows) use IP to deliver data. Its functions include:

- Defining the IP packet (i.e., a datagram, the basic transmission unit in the Internet),
- Defining the Internet addressing scheme,
- Moving data between network and transport layers,
- Routing datagrams to remote hosts,
- Performing fragmentation and re-assembly of datagrams.

IP is an *unreliable protocol*, because it does not perform error detection and recovery for transmitted data. This does not mean that we cannot rely on this protocol. In fact, IP can be relied upon to deliver data accurately to the destination, but it does not check whether data are received correctly or not. Higher-layer protocols of the Internet protocol stack are in charge of providing error detection and recovery, if required.

The protocol layer just above the network one is the *host-to-host transport layer*. This name is commonly shortened to *transport layer*. The two most important protocols at the transport layer are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). TCP provides a reliable, connection-oriented, byte-stream data delivery service; error detection and error recovery (through retransmissions) are end-to-end performed. UDP provides a low-overhead, unreliable, connectionless datagram delivery service. Both protocols exchange data between application and network layers. Applications programmers can choose the service that is most appropriate for their specific needs.

UDP gives application programs direct access to a datagram delivery service, like the delivery service provided by IP. This allows applications to exchange messages over the network with a minimum protocol overhead.

Applications requiring the transport protocol to provide reliable data delivery use TCP, since it verifies that data are accurately delivered across the network and in the right sequence.

The *application layer* is at the top level of the TCP/IP protocol architecture. This layer includes all processes that use transport protocols to deliver data. There are many application layer protocols. Most of them provide user services; new services are constantly being added at this layer. The most popular and implemented application layer protocols are:

- Telnet: The network terminal protocol, which allows us to remotely log on hosts spread in network.
- FTP: The File Transfer Protocol used for file transfer.
- SMTP: The Simple Mail Transfer Protocol, which delivers electronic mail.

- HTTP: The Hypertext Transfer Protocol, delivering Web pages over the network.
- Domain Name System (DNS): This is a service to map IP (numeric) addresses to the names assigned to network devices.
- Network File System (NFS): This protocol permits to share files among various hosts in the network.
- Finally, the Open Shortest Path First (OSPF), which is a layer 3 routing protocol, includes a transfer protocol for the exchange of routing information among routers and as such (even with some debate) can also be considered as an application layer protocol.

3.3 IP (version 4) addressing

IP addresses are used to route datagrams in the network and to allow their correct delivery to destination. An IP version 4 (IPv4) address is formed of 32 bits, written by dividing the bits in groups of 8 and taking the corresponding decimal number. Each of these numbers is written separated by a dot (i.e., dotted-decimal notation) and can range from 0 to 255. For example, 1.160.10.240 could be an IP address. The specification of IP addresses is contained in RFC 1166 [13]. An IP address can be divided in a pair of numbers (the length of these fields depend on the IP address class):

IP address = <network identifier> + <host identifier> .

There are five classes of IP addresses, as described in Figure 3.3. Classes are introduced to divide the space of IP addresses in groups of a limited number of addresses (i.e., that can support a limited number of hosts). This is carried out for an efficient use of IP addresses and takes the name of ‘classful’ IPv4 addressing.

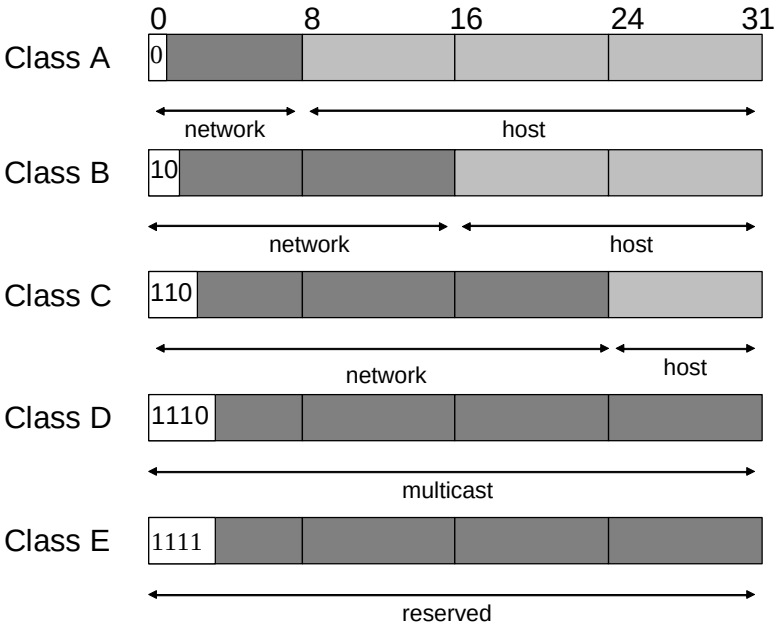

Figure 3-3. IPv4 address classes.

For classes A, B, and C, the address of a network has all the host bits equal to '0'. Whereas, the broadcast address of a network is characterized by all the host bits equal to '1'. The number of hosts addressable in a network is therefore related to the number of available combinations for the bits of the host field minus two addresses for network and multicast purposes.

Class A:

- First bit set to '0' plus 7 network bits and 24 host bits
- Initial byte ranging from 0 to 127
- Totally, 128 (= 2^7) Class A network addresses are available (0 and 127 network addresses are reserved)
- 16,777,214 (= 2^{24}–2) hosts can be addressed in each Class A network

Class B:

- First two bits set to '10' plus 14 network bits and 16 host bits
- Initial byte ranging from 128 to 191
- Totally, 16,384 (= 2^{14}) Class B network addresses
- 65,534 (= 2^{16}–2) hosts can be addressed in each Class B network

Class C:

- First three bits set to '110' plus 21 network bits and 8 host bits
- Initial byte ranging from 192 to 223
- Totally, 2,097,152 ($= 2^{21}$) Class C network addresses
- 254 ($= 2^8 - 2$) hosts can be addressed in each Class C network

Class D:

- First four bits set to '1110' plus 28 multicast address bits
- Initial byte ranging from 224 to 247
- Class D addresses are used for multicast flows

Class E:

- First four bits set to '1111' plus 28 reserved address bits
- Initial byte ranging from 248 to 255
- This address class is reserved for experimental use.

A router receiving an IP packet extracts its IP destination address, which is classified by examining its first bits. Once the IP address class has been determined, the IP address can be broken down into network and host bits. Intermediate routers ignore host bits and only need to match network bits within their routing table to route the IP packet along the correct path in the network. Once a packet reaches its target network, its host field is examined for the final local delivery.

IPv4 addressing space is limited: this is a significant problem because of the continued spread of the Internet. In order to address this issue, possible approaches are: IP subnetting (see Section 3.3.2), the use of private IP addresses (see Section 3.3.3), and the new IP version 6 (see Section 3.3.6).

3.3.1 IPv4 datagram format

Data transmitted over the Internet using IP addresses are organized in variable-length packets, called IP datagrams. Let us consider here the IPv4 datagram format, defined in RFC 791 [6]. An IPv4 datagram is divided into two parts: the header and the payload. The header contains addressing and control fields, while the payload carries the actual data to be sent. Even though IP is a relatively-simple, connectionless, "unreliable" protocol, the IPv4 header carries some control information that makes it quite long. It is minimum 20 byte long and can be even longer with the options. The IP datagram format is shown in Figure 3.4, where each row corresponds to four

bytes (i.e., a word of 32 bits). The meaning of the different header fields is explained below.

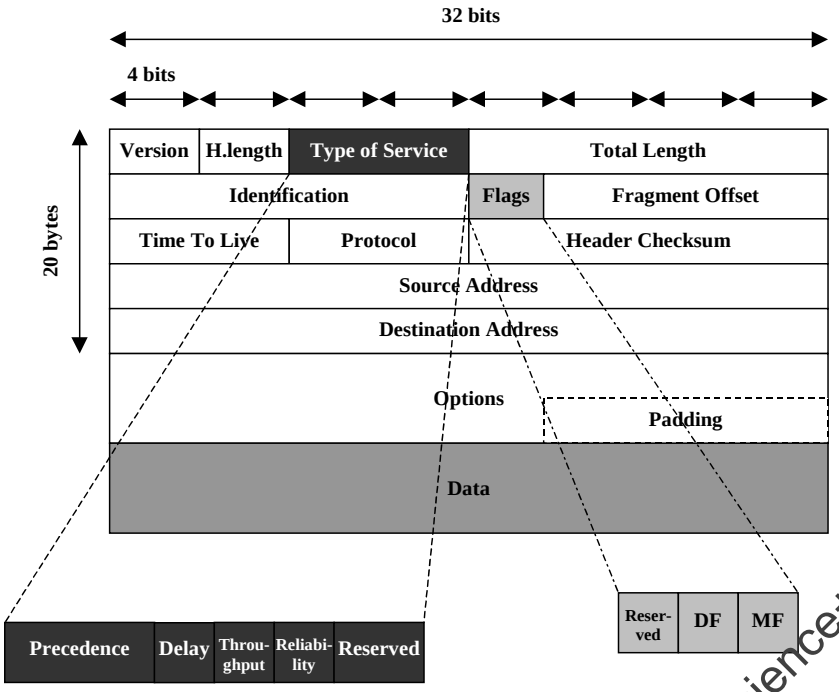

Figure 3-4. IPv4 datagram format.

- Version (4 bits): Identifies the IP version of the datagram. For IPv4, obviously this field contains the number 4. The purpose of this field is to ensure compatibility among different devices, which may be running different IP versions. In general, a device running an older IP version will reject datagrams created by newer implementations.
- IHL, Internet Header Length (4 bits): Specifies the length of the IP header in 32-bit words. This length includes any optional field and padding. The normal value of this field when no options are used is 5 (i.e., 5 words of 32 bits, corresponding to 20 bytes).
- ToS, Type of Service (8 bits): A field carrying information to support quality of service features, such as prioritized delivery of IP datagrams. The ToS byte is divided into four sub-fields, as shown in Figure 3.4:
  - The first three bits are used for the *precedence* field (value of 0 for a normal priority, up to a value of 7 for control messages);

- The *delay* bit specifies whether a low delay is required for the datagram transfer (D = 1) or if the delay is not critical (D = 0);
- The *throughput* bit T = 1 when a high throughput is needed, instead T = 0 if the throughput is not a critical issue;
- The *reliability* bit R = 1 when a high reliability is required, instead R = 0 if reliability is not needed.
- The last two bits are unused.

The ToS byte has never been used as originally defined. A great deal of experimental, research and deployment work has focused on how to use these 8 bits (ToS field), which have been redefined by IETF for use by Differentiated Services (DiffServ) and by Explicit Congestion Notification (ECN); see also the following Section 3.5 and sub-Sections 3.7.8.2, 3.7.8.3.

- TL, Total Length (16 bits): This field specifies the total length of the IP datagram in bytes. Since this field is 16 bits wide, the maximum length of an IP datagram is 65,535 bytes (typically, they are much smaller to avoid fragmentation due to MAC layer constraints). The most common IP packet length is 1500 bytes to be compatible with the maximum Ethernet payload size.
- Identification (16 bits): This field contains a 16-bit value, which is common to each fragment belonging to the same message. It is filled in for originally-unfragmented datagrams, in case they have to be fragmented at an intermediate router along the path. Such field is used by the recipient to reassemble messages in order to avoid an accidental mixing of fragments coming from different messages, since the IP datagrams can be received out of order.
- Flags (3 bits): It contains three control flags, but only two of them are used: Do not Fragment (DF) flag and More Fragments (MF) flag. If DF = 1, the datagram should not be fragmented. MF = 0 denotes the last fragment of a datagram.
- Fragment Offset (13 bits): When a message is fragmented, this field specifies the position of the current data fragment in the overall message. It is specified in units of 8 bytes (64 bits). The first fragment has an offset of 0.
- TTL, Time To Live (8 bits): This field specifies how long a datagram is allowed to "live" in the network in terms of router hops. Each router decrements the TTL value of one, before transmitting the related datagram. If TTL becomes zero, the datagram is not forwarded, but

EXTRACT FROM PART II

Chapter 6

M/G/1 QUEUING THEORY AND APPLICATIONS

6.1 The M/G/1 queuing theory

The M/G/1 theory is a powerful tool, generalizing the solution of Markovian queues to the case of general service time distributions. There are many applications of the M/G/1 theory in the field of telecommunications; for instance, it can be used to study the queuing of fixed-size packets to be transmitted on a given link (i.e., M/D/1 case). Moreover, this theory yields results, which are compatible with the M/M/1 theory, based on birth-death Markov chains.

In the M/G/1 theory, the arrival process is Poisson with mean arrival rate λ, but, in general, the service time is not exponentially distributed. Hence, the service process has a certain memory: if there is a request in service at a given instant, its *residual service time* has a distribution depending on the time elapsed since the beginning of its service. Let us refer to a generic instant t. The system is described by a *two-dimensional* state S(t), characterized as follows:

- Number of requests in the system at instant t, n(t);
- Elapsed time from the beginning of the service of the currently-served request, τ(t). Note that in the Markovian M/M/1 case, the pdf of the residual service time does not depend on τ (t) because of the memoryless property of the exponential distribution.

Hence, $S(t) = \{n(t), \tau(t)\}$. In order to characterize these queues, we study their behaviors at specific time instants ζ_i where we obtain a mono-dimensional simplification of state $S(\zeta_i)$. The M/G/1 queue is studied at specific imbedding instants, where we obtain again a Markovian system; this is a so-called *imbedded Markov chain* [1],[2]. Different alternatives are available to select instants ζ_i. It is not requested that instants ζ_i be equally spaced in time. Typical choices for ζ_i instants are:

1. Service completion instants;
2. Arrival instants {as done for G/M/1 queues to study the waiting time [3]};
3. Regularly-spaced instants for cases with service based on time slots.

It makes a difference how we select the imbedding points: different imbedding options in general do not allow to achieve the same results. In this study, let us refer to the first type of imbedding points: let ζ_i denote the service completion instant of the i-th request arrived at the queue. We have that $\tau(\zeta_i) \equiv 0 \ \forall i$, since at instant ζ_i a request has completed its service and no new request has yet started its service. Hence, at these instants ζ_i the state becomes mono-dimensional: $S(\zeta_i) \equiv n(\zeta_i) = n_i$, where n_i denotes the number of requests in the queue soon after the service completion of the i-th request. Let a_i denote the number of requests arrived at the queue during the service time of the i-th request (ending at instant ζ_i). Note that n_i and a_i random variables are also used with different imbedding points, but the distributions of both n_i and a_i depend on the imbedding instants selected.

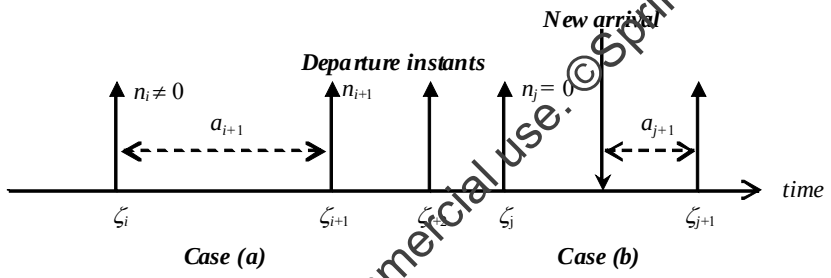

Figure 6-1. Time diagram of service completion events and new arrivals.

Let us refer to the situation depicted in Figure 6.1. If $n_i \neq 0$ [i.e., case (a) in Figure 6.1], the following balance is valid at the next service completion instant: $n_{i+1} = n_i - 1 + a_{i+1}$. Instead, if $n_i = 0$ [i.e., case (b) in Figure 6.1], we

have to wait for the next arrival, which is served immediately, so that at the next completion instant ζ_{i+1} the system contains only the arrivals occurred during the service time of the last request; this number is still represented by variable a_{i+1}. Hence, we have: $n_{i+1} = a_{i+1}$.

Let us recall that the indicator (Heaviside) function is defined as: $I(x) = 1$ for $x > 0$; $I(x) = 0$ for $x \leq 0$. By means of function $I(x)$, we can represent n_{i+1} with an expression, which is valid for both $n_i \neq 0$ and $n_i = 0$, as shown below where we have also provided alternative notations adopted in the literature:

$$n_{i+1} = n_i - I(n_i) + a_{i+1} = \max\{n_i - 1,\ 0\} + a_{i+1} = (n_i - 1)^+ + a_{i+1} \quad (6.1)$$

The difference equation (6.1) describes the behavior of the M/G/1 queue at imbedding instants. Since the variables at the instant ζ_{i+1} depend only on the variables at instant ζ_i , equation (6.1) characterizes the M/G/1 system by means of a discrete-time Markov chain (or, more correctly, an imbedded Markov chain). Note that the method of imbedding instants is quite general and has also been applied to study G/M/1 queues (general iid interarrival times; exponentially-distributed service times; one server). In this case, the chain is imbedded at the arrival instants of the input process [3].

Let G(t) denote the PDF of the service time, X: $G(t) = \text{Prob}\{X \leq t\}$. Let g(t) denote the pdf of the service time: $g(t) = dG(t)/dt$. The mean service time is indicated as E[X].

Let us assume that the M/G/1 queue admits a steady state with P_n denoting the probability (at regime) to have n requests in the queue at imbedding instants:

$$\lim_{i\to\infty} P_{n_{i+1}} = \lim_{i\to\infty} P_{n_i} = P_n$$

Hence, we have:

$$\lim_{i\to\infty} E[n_{i+1}] = \lim_{i\to\infty} E[n_i] = E[n], \text{ where } E[n] \text{ denotes the regime value.}$$

By taking the expected values of both sides of (6.1), we have:

$$E[n_{i+1}] = E[n_i] - E[I(n_i)] + E[a_{i+1}] \quad (6.2)$$

Hence, if we take the limit of both sides for $i \to \infty$, we obtain regime values as:

$$E[n] = E[n] - E[I(n)] + E[a] \Rightarrow E[a] = E[I(n)]$$

We can evaluate E[I(n)] by means of the state probability distribution as:

$$E[I(n)] \overset{def}{=} \sum_{n=0}^{\infty} I(n) P_n = \sum_{n=1}^{\infty} P_n = 1 - P_0 \tag{6.3}$$

By using (6.3) and the expression at regime corresponding to (6.2), we can obtain probability P_0 as:

$$P_0 = 1 - E[a] \tag{6.4}$$

Let us recall that on the basis of the PASTA property P_0 (or $1-P_0$) is the probability that a new arrival finds an empty (or a non-empty) M/G/1 queue.

The mean number of arrivals during the service time of a request, E[a], can be obtained as the mean number of Poisson arrivals conditioned on a given service time X = t, E[a | X = t] = λt, and, then, by removing the conditioning by means of the pdf g(t) of X:

$$E[a] = \int_0^{\infty} E[a \mid X = t] g(t) dt = \lambda \int_0^{\infty} t g(t) dt = \lambda E[X] \tag{6.5}$$

From (6.5) we note that E[a] corresponds to the traffic intensity expressed in Erlangs, ρ. The M/G/1 queue is stable if $P_0 > 0$, or, equivalently on the basis of (6.4) and (6.5), if $\rho < 1$ Erlang.

We focus here on the solution of the difference equation (6.1) in the z domain by means of PGFs. First of all, we consider the equality obtained by taking the exponentiation with base z on both sides of (6.1) for any index i value:

$$z^{n_{i+1}} = z^{n_i - I(n_i) + a_{i+1}}, \qquad \forall i$$

Then, we multiply both sides by the joint distribution Prob{n_{i+1}=h, n_i=k, a_{i+1}=j} and we sum over h, k, j. The summations on k and j can be removed on the left side; moreover the summation on h can be removed on the right side. Details are as follows:

left side :

$$\sum_h \sum_k \sum_j z^{n_{i+1}} P_{n_{i+1},n_i,a_{i+1}} = \sum_h z^{n_{i+1}} \sum_k \sum_j P_{n_{i+1},n_i,a_{i+1}} = \sum_h z^{n_{i+1}} P_{n_{i+1}}$$

right side :

$$\sum_h \sum_k \sum_j z^{n_i - I(n_i) + a_{i+1}} P_{n_{i+1},n_i,a_{i+1}} = \sum_k \sum_j z^{n_i - I(n_i) + a_{i+1}} \sum_h P_{n_{i+1},n_i,a_{i+1}} =$$

$$= \sum_k \sum_j z^{n_i - I(n_i) + a_{i+1}} P_{n_i,a_{i+1}}$$

By equating the two expressions above, we obtain:

$$\sum_h z^{n_{i+1}} P_{n_{i+1}} = \sum_k \sum_j z^{n_i - I(n_i) + a_{i+1}} P_{n_i,a_{i+1}} \tag{6.6}$$

In order to solve the imbedded Markov chain we make the following assumptions:

1. Memoryless arrival process ([1]);
2. Arrival process independent of the number of requests in the queue; n_i and a_{i+1} are independent variables ([2]).

The above assumptions are quite general and can be met by many systems. In particular, they are verified in the special case of Poisson arrivals and general service time, which are both independent of the queue state.

Under the previous assumption #2, Prob$\{n_i=k, a_{i+1}=j\}$ = Prob$\{n_i=k\}$ × Prob$\{a_{i+1}=j\}$. Therefore, the left side in (6.6) can be rewritten as:

$$\sum_h z^{n_{i+1}} P_{n_{i+1}} = \sum_k z^{n_i - I(n_i)} P_{n_i} \sum_j z^{a_{i+1}} P_{a_{i+1}} \tag{6.7}$$

[1] In the case of continuous-time processes, we have to consider Poisson (or compound Poisson) processes. Instead, in the case of discrete-time processes, we have to consider Bernoulli or Binomial arrival processes on a slot basis (in this respect, symbol M used to denote the arrival process at the queue has to be considered in a wider sense and as such it will be substituted by 'M').

[2] Note that it is also possible to solve (6.6) by removing such assumption: we obtain a recursive formula to determine the state probabilities P_n at imbedding instants. More details are provided in the following Section 6.5.

Let P(z) denote the PGF at regime of the state probability distribution at the imbedding instants. Let A(z) denote the PGF at regime of the number of arrivals during the service time of a request. Moreover, note that:

$$\sum_{k=0}^{\infty} z^{n_i - I(n_i)} P_{n_i} = P_{0i} + \sum_{k=1}^{\infty} z^{n_i - 1} P_{n_i} = P_{0i} + z^{-1} \sum_{k=1}^{\infty} z^{n_i} P_{n_i} = \\ = P_{0i} + z^{-1} \left\{ \sum_{k=0}^{\infty} z^{n_i} P_{n_i} - P_{0i} \right\} \tag{6.8}$$

By considering the situation at regime (i.e., for $i \to \infty$), we can eliminate subscript i in equations (6.7) and (6.8). Then, we substitute (6.8) in equation (6.7) where we use the PGFs P(z) and A(z):

$$P(z) = \left\{ P_0 + z^{-1} \left[P(z) - P_0 \right] \right\} A(z) \tag{6.9}$$

Finally, we can solve P(z) in (6.9):

$$P(z)\left[z - A(z)\right] = P_0 (z-1) A(z) \Rightarrow P(z) = P_0 \frac{(z-1)A(z)}{z - A(z)} \tag{6.10}$$

The PGF of the state probability distribution in (6.10) represents a quite general formula, which can be applied to all the imbedded Markov chains fulfilling (6.1) and the previous assumptions #1 and #2. In particular, the PGF in (6.10) is valid for any service policy, provided that the conditions of the insensitivity property are fulfilled (see Section 5.5).

Since P_0 is determined from (6.4), the PGF of the state probability distribution depends only on the PGF A(z), which, in turn, depends on both the arrival process and the imbedding instants. The state probability distribution can be obtained by inverting (6.10). This is not an easy task, since there may not be a closed form solution: the PGF in (6.10) typically does not correspond to a classical distribution. A possible approach to invert P(z) is to adopt the method of the Taylor series expansion centered at z = 0, as show in Section 4.3.1: the coefficients of the expansion represent the state probability distribution. This approach requires a numerical method based on the Matlab® symbolic toolbox. Another method to invert (6.10) is described in Section 6.5.

By means of (6.4), the *stability condition* $P_0 > 0$, can be expressed as follows, noticing that E[*a*] = A'(*z*=1):

$$P_0 = 1 - A'(1) > 0 \quad \Rightarrow \quad A'(1) < 1 \ \left[\text{Erlang}\right]$$

Under the assumption of Poisson arrivals and imbedding at the service completion instants, A(z) can be derived considering the PGF of the number of arrivals in a given interval X = t, $A(z \mid X = t) = e^{\lambda t(z-1)}$ and then removing the conditioning by means of the general pdf of the service time X, g(t):

$$A(z) = \int_0^{+\infty} e^{\lambda t(z-1)} g(t) dt = \Gamma\left[s = -\lambda(z-1)\right] \tag{6.11}$$

where Γ(s) denotes the Laplace transform of the pdf g(t). On the basis of the expression of A(z) in (6.11) we can evaluate A'(1) and A''(1) as follows:

$$\left.\frac{dA(z)}{dz}\right|_{z=1} = -\lambda\Gamma'\left[-\lambda(z-1)\right]_{z=1} = \lambda\left[-\Gamma'(0)\right] = \lambda E\left[X\right] \tag{6.12}$$

$$\begin{aligned}\left.\frac{d^2A(z)}{dz^2}\right|_{z=1} &= \left.\frac{d}{dz}\left\{-\lambda\Gamma'\left[-\lambda(z-1)\right]\right\}\right|_{z=1} = \lambda^2\Gamma''\left[-\lambda(z-1)\right]_{z=1} \\ &= \lambda^2\Gamma''(0) = \lambda^2 E\left[X^2\right]\end{aligned} \tag{6.13}$$

Note that (6.12) is equivalent to (6.5).

The PGF in (6.10) has a singularity at z = 1 (a removable singularity according to the Abel theorem), which causes some problems for both the normalization condition and the derivation of the moments of the distribution. Of course, we can use the Hôpital theorem to prove that P(z = 1) = 1 (normalization). Moreover, the moments of the state probability distribution can be obtained by taking subsequent derivatives on both sides of the leftmost expression in (6.10). With the first derivative, we have:

$$P'(z)\left[z - A(z)\right] + P(z)\left[1 - A'(z)\right] = P_0 A(z) + P_0(z-1)A'(z) \tag{6.14}$$

If we evaluate (6.14) at z = 1, we obtain: $P_0 = 1 - A'(1)$; this is the same expression as in (6.4).

If we derive again (6.14) on both sides with respect to z we obtain:

$$P''(z)[z - A(z)] + 2P'(z)[1 - A'(z)] + P(z)[-A''(z)] = \\ = 2P_0 A'(z) + P_0(z-1)A''(z) \tag{6.15}$$

If we evaluate (6.15) at z = 1 and we use (6.4) for P_0, we have:

$$2P'(1)[1 - A'(1)] - A''(z) = 2P_0 A'(1) \\ \Rightarrow N = P'(1) = A'(1) + \frac{A''(z)}{2[1 - A'(1)]} \tag{6.16}$$

The mean number of requests in the queue at imbedding instants, N, depends on the first two derivatives of A(z) computed at z = 1. Let us recall that the stability condition is met if 1 – A'(1) > 0, i.e., traffic intensity is lower than 1 Erlang. Note that (6.16) is a general expression, which could also be applied to memoryless arrival processes different from the Poisson one provided that the imbedded system is characterized by (6.1). If we refer to Poisson arrivals (i.e., the classical M/G/1 queue) and imbedding points at service completion instants, we can substitute (6.12) and (6.13) in (6.16), thus yielding:

$$N = \lambda E[X] + \frac{\lambda^2 E[X^2]}{2[1 - \lambda E[X]]} \tag{6.17}$$

We can derive the mean delay to cross the queuing system, T, by applying the Little theorem to (6.16) for the more general case or to (6.17) for the Poisson arrival case. In particular, referring to (6.17), we obtain the well-known Pollaczek-Khinchin formula for the mean queuing delay [1],[2],[4]:

$$T = \frac{N}{\lambda} = E[X] + \frac{\lambda E[X^2]}{2[1 - \lambda E[X]]} \tag{6.18}$$

Note that in (6.18) the first contribution to the mean delay is E[X], i.e., the *mean service time*; instead, the second contribution $\lambda E[X^2]/\{2[1-\lambda E[X]]\}$ represents the *mean waiting time*. The mean queuing delay is related to the second moment of the service time distribution. In particular, the mean waiting time increases with the variance of the service time, considering a certain fixed mean service time. If the traffic intensity of the input arrival process, $\lambda E[X]$, tends to 1 Erlang (stability limit), the mean delay tends to infinity.

In the case of exponentially-distributed service times (mean rate μ), the above formulas (6.17) and (6.18) yield the same expressions of the M/M/1 queue as shown in Chapter 5. In this case, we have $\Gamma(s) = \mu/(\mu+s)$, $E[X] = 1/\mu$ and $E[X^2] = 2/\mu^2$. As shown in [1],[2], this result permits to conjecture that the state probability distribution obtained for an M/G/1 system at the imbedding instants is also valid in general for the continuous-time chain. These considerations can be supported more formally introducing the Kleinrock principle [1]: for queuing systems where the state changes at most by +1 or −1 (we refer here to actual changes in the number of requests in the queue and not to what happens only at imbedding instants), the state distribution as seen by an arriving customer is the same as that seen by a departing customer. Hence, the state probability distribution at departure instants is equal to the state probability distribution at arrival instants. Moreover, by applying the PASTA property (in the Poisson arrival case), the state probability distribution at arrival instants is also valid at generic instants (random observer). Hence, by means of both the Kleinrock principle and the PASTA property, we can conclude that the state probability distribution at service completion instants coincides with the distribution of the continuous-time system (random observer). As for discrete-time (Markov) systems, the equivalent BASTA property can be adopted to determine the probability that an arrival finds the queue in a certain state by means of the corresponding state probability.

6.1.1 The M/D/1 case

In this system the requests have a fixed, constant service time, x. This is for instance the case of the transmission of packets of a given size on a link with constant capacity. Therefore, the pdf of the service time becomes $g(t) = \delta(t-x)$, where $\delta(.)$ denotes the Dirac Delta function. The corresponding Laplace transform is $\Gamma(s) = e^{-xs}$. By using (6.11), we have: $A(z) = \Gamma(s)|_{s=-\lambda(z-1)} = e^{x\lambda(z-1)}$. Note that λx is the intensity of the input traffic in Erlangs. By substituting this expression of A(z) in (6.10), we obtain P(z) with imbedding points at the service completion instants as:

$$P(z) = (1-\lambda x)\frac{(z-1)e^{\lambda x(z-1)}}{z - e^{\lambda x(z-1)}} \tag{6.19}$$

Note that the PGF of an M/D/1 queue in (6.19) cannot be anti-transformed in closed form, so that numerical methods (as those discussed in Section 4.3.1) are needed to obtain the state probability distribution.

Finally, the mean number of requests in the queue N can be expressed according to (6.17) as:

$$N = \lambda x + \frac{\lambda^2 x^2}{2[1-\lambda x]} = \frac{\lambda x}{1-\lambda x} - \frac{\lambda^2 x^2}{2[1-\lambda x]} \tag{6.20}$$

Hence, N has [rightmost term in (6.20)] a contribution corresponding to that of an M/M/1 queue (with the same mean arrival rate and the same mean service time) minus a positive term. Hence, the congestion of an M/D/1 queue is lower than that of the corresponding M/M/1 queue. The same relation holds for the mean system delay given by (6.18). This is consistent with the fact that for the same mean service time the exponential distribution has a mean square value two times larger than that of a deterministic distribution.

6.1.2 The $M^{[comp]}/G^{[b]}/1$ queue with bulk arrivals or bulk service

The queue with bulk (compound) arrivals (as defined in Section 5.1.1) and imbedding instants at the end of the service of each object entails a modification in (6.1) when $n_i = 0$: when the service is completed for the first object of a group arrived at an empty system, the remaining objects in the queue are not only those arrived during the service time of this object, but also the remaining objects belonging to the same group arrived at an empty system. Let *m* denote the random variable of the length of a group in objects. Then, the difference equation in (6.1) for $n_i = 0$ becomes: $n_{i+1} = a_{i+1}^*$, where $a_{i+1}^* = m - 1 + a_{i+1}$; this model corresponds to the differentiated service time case detailed in Section 6.10. According to the previous notations, this queuing system can be denoted as $M^{[comp]}/G/1$.

In the bulk service case, b arrivals (= objects) can be serviced together in the imbedding interval. This is for instance the case of TDM(A) transmissions with a frame-based allocation of packets having fixed service time: the imbedding points are here at the end of each frame. With bulk service, the difference equation (6.1) has to be modified when $n_i \neq 0$, thus obtaining the following expression: $n_{i+1} = \max(n_i - b, 0) + a_{i+1}$. According to the previous notations, this queuing system can be denoted as $M/G^{[b]}/1$; in the TDM(A) case we have actually an $M/D^{[b]}/1$ queue.

More details on the solution of these cases (including the consideration of different imbedding options) will be provided in Section 6.7.

6.2 M/G/1 system delay distribution in the FIFO case

This Section provides an extension of the study made in Section 5.11.1 to the case of general service times. As long as possible, we keep the same notations as those used in Section 5.11.1. Let us refer to a queue with FIFO discipline, Poisson arrivals, general service time, and system imbedded at service completion instants. The n requests left in the system at the service completion instant are those arrived during the system delay T_D experienced by the request just served; see Figure 6.2.

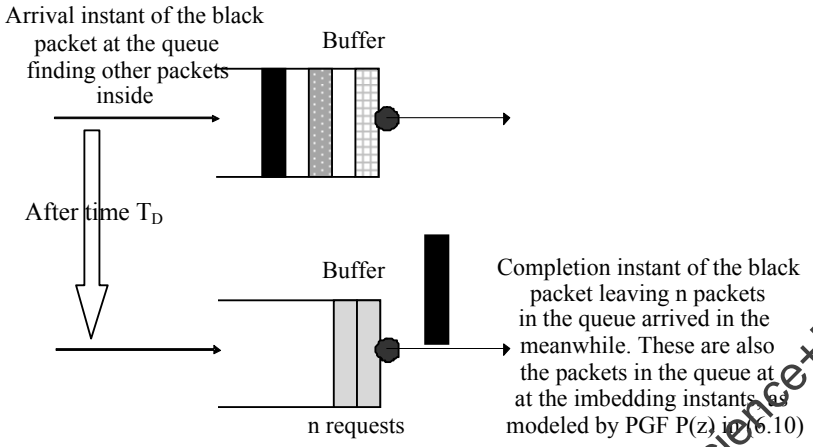

Figure 6-2. Relation between random variable T_D of the queuing delay and the PGF P(z) of the number of requests n in the queue at imbedding instants.

The probability distribution for the n requests in the system at the service completion instants coincides with the state probability distribution with PGF P(z) in (6.10). This PGF of random variable n can also be obtained referring to the fact that these n requests are the arrivals at the system during the system delay T_D, with corresponding pdf $f_D(t)$ [note that $f_D(t)$ is the unknown distribution that we need to characterize]. Let us first condition our study on a given system delay $T_D = t$, so that the PGF of the number of Poisson arrivals in this interval is: $P(z \mid T_D = t) = e^{\lambda t(z-1)}$. Then, we remove the conditioning by means of the pdf $f_D(t)$ as:

$$P(z) = \int_0^{+\infty} e^{\lambda t(z-1)} f_D(t) dt = T_D[s = -\lambda(z-1)] \tag{6.21}$$

where $T_D(s)$ is the Laplace transform of the pdf $f_D(t)$.