Abstraction is one of the means to cope with system complexity, but abstractions could drive us away

from physical reality. We should not forget that our objective is to build systems that store, transform,

and transport information and that the physical properties of such systems must be well understood.

Three fundamental abstractions are sufficient to describe all possible elements of computing and

communication systems: storage, interpreter, and communication channel. An interpreter is an active

element of a system that transforms information; we distinguish hardware interpreters, such as a processor

or a disk controller, from software interpreters, such as scripting languages and text-processing

systems. In all the possible embodiments of this abstraction, the interpreter, we recognize three elements

[312]: (i) an action reference, which tells the interpreter where to find the next action; (ii) a repertoire,

which defines the set of actions the interpreter is able to perform; and (iii) an environment reference,

which points to the environment and the state in which the interpreter should be when executing the next

action. The physical properties of actual embodiments of an interpreter span a very large spectrum; for

example, when we design an application for a mobile device, a major concern in selecting the processor

should be its power consumption, whereas the main concern for the browser should be the size of the

screen.

Information has a physical support, and each one of these three processes acts on a property of that

physical system. For example, to transport information from a sender to a receiver, we can encode it in the amplitude, the frequency, or the phase of electromagnetic waves, and we have to well understand

the interaction of the electromagnetic waves with the communication media. Indeed, if we want to

have error-free communication channels we have to determine the error rate and use error-correcting or

error-detecting code designed to cope with the error rate.

We have to maintain a delicate balance between the abstractions that are critical to the development

of algorithms and the physical characteristics of the systems used to store, transform, and transport

information. This balance has shifted in recent years due to the widespread use of cyberphysical systems.

In such systems there is a tight relationship between the computational elements and the physical

elements. Embedded systems are ubiquitous, and sensors allow them to react to the properties of the

physical systems they control and, at the same time, to monitor their own properties and adapt to changes

in the environment.

One dimension of the design of modern complex systems is related to their operational environment.

The recently introduced concept of sociotechnical systems (see, for example, [329]) captures the fact

that the design of modern systems should consider not only the technological aspect but also the

environment – more precisely, the humans who use a system.

From this informal discussion of several attributes of complex systems we move to quantitative

measures of system complexity, the topic of the next section.