Chapter 1
Introduction to Modelling and Validation

System development and engineering is a comprehensive discipline involving a multitude of activities such as requirements engineering, design and specification, implementation, testing, and deployment. An increasing number of system development projects are concerned with concurrent systems. There are numerous examples of this, ranging from large-scale systems, in the areas of telecommunication and applications based on Internet technology, to medium- or small-scale systems, in the area of embedded systems.

Section 1.1 introduces the basic ideas and motivation for modelling in system development, and Sect. 1.2 gives a high-level overview of the CPN modelling language. Section 1.3 discusses the role of abstraction and visualisation when one is constructing models of concurrent systems. Section 1.4 presents the benefits of formal modelling languages and verification. Section 1.5 gives an overview of the main features of CPN Tools. Finally, Sect. 1.6 provides an overview of the results from four industrial projects using CP-nets. A more detailed description of the projects will be given in Chap. 14.

1.1 Modelling and System Development

The development of concurrent systems is particularly challenging. A major reason is that these systems possess concurrency and non-determinism which means that the execution of such systems may proceed in many different ways, for example, depending on whether messages are lost during transmission, the scheduling of processes, and the time at which input is received from the environment. Hence, such systems have an astronomical number of possible executions. It is extremely easy for a human designer to miss some important interaction patterns when designing such a system, leading to gaps or malfunctions in the system design. As a result, concurrent systems are, by nature, complex and difficult to design, test, and debug. Furthermore, for many concurrent systems such as those integrated into nuclear power plants, aircraft control systems, and hospital life support equipment, it is essential that the system works correctly from the very beginning. To cope with the
complexity of modern concurrent systems, it is therefore crucial to provide methods that enable the debugging and testing of central parts of system designs prior to implementation and deployment.

One way to approach the challenge of developing concurrent systems is to build a model of the system. Modelling is a universal technique that can be used across many of the activities in system development. Many modelling languages have been suggested, and many are being used for system development. One prominent example is the Unified Modeling Language (UML) [94] which is becoming the de-facto standard modelling language of the software industry and which supports modelling of the structure and behaviour of systems. The focus of this textbook is on executable models that can be used to simulate the behaviour of systems.

The act of constructing a model of a system to be developed is typically done in the early phases of system development, and is also known from other disciplines, such as when engineers construct bridges and architects design buildings. For example, architects make architectural drawings and may build three-dimensional models in cardboard, plastic, or plywood, or use computerised 3D animation to visualise a building. The purpose of these different models is to get a better impression of the building. This allows the architect and the intended owners and users of the building to imagine what the building will look like and how it will function, for example, whether some corridors are too narrow or some doors so close to each other that they may create dangerous situations. The main motivation behind such models is that it is obviously preferable to correct design errors and other shortcomings before the construction of the real building commences.

When a new concurrent system is being designed or an existing one is being investigated, there are similar reasons why it is beneficial to build a model of it and to build it as early as possible.

- **Insight.** The act of constructing the model and simulating it usually leads to significant new insights into the design and operation of the system considered. Typically the modeller gains a more elaborate and complete understanding of the system than what can be obtained by conventional means, for example, by reading design documents. The same applies to people to whom a model of a system is being presented. This insight often results in a simpler and more streamlined design. By investigating a model, similarities can be identified that can be exploited to unify and generalise the design and make it more logical, or we may get ideas to improve the usability of the system.

- **Completeness.** The construction of an executable model usually leads to a more complete specification of the design. Gaps in the specification of the system will become explicit as they will prohibit the model from being executed because certain parts are missing, or when the model is simulated, the designers and users will find that certain expected events are not possible in the current state. Modelling also leads to a more complete identification and understanding of the requirements to be placed on the system, in particular because models can be used to mediate discussions among designers and users of the system.

- **Correctness.** When simulations of a model are made a number of design errors and flaws are usually detected. Since the modeller is able to control the execution