K. Jensen, L.M. Kristensen, Coloured Petri Nets,
DOI 10.1007/b95112_15,
(C) Springer-Verlag Berlin Heidelberg 2009

Chapter 15
Teaching Coloured Petri Nets

This chapter describes a course on the modelling and validation of concurrent systems based on this textbook which we have been giving at the Department of Computer Science, at Aarhus University. The course uses CP-nets as a formal modelling language for concurrency and exposes students to the benefits and applications of modelling for designing and reasoning about the behaviour of concurrent systems. The course introduces participants to the CPN modelling language, its analysis methods, and its supporting computer tools. It also includes a presentation of industrial projects where CP-nets have been used for the modelling and validation of systems. After the course, the participants will have a detailed knowledge of CP-nets and practical experience in the modelling and validation of concurrent systems. The course emphasises the practical use of modelling and validation and has less focus on the formal foundation of CP-nets. The slide sets, CPN models, and suggestions for exercises and projects from the course are available via the Web pages for this textbook.

Section 15.1 describes the overall organisation of the course and explains its context in the curriculum. Section 15.2 discusses the intended learning outcomes of the course, and Sect. 15.3 presents the teaching and assessment methods used. Section 15.4 gives an example of a representative student project conducted during the course. Section 15.5 discusses our experiences obtained when developing and giving the course.

15.1 Course Context and Aims

The course is divided into two parts, each lasting seven weeks, and participants may choose to follow only the first seven weeks. Each part of the course corresponds to 5 ECTS (European Credit Transfer and Accumulation System), which means that the participants are expected to spend one-third of their study time on the course. The aim of the first part of the course is that the participants will obtain a detailed knowledge of CP-nets and gain experience in the modelling and validation of small

K. Jensen, L.M. Kristensen, *Coloured Petri Nets*, DOI 10.1007/b95112_15,
© Springer-Verlag Berlin Heidelberg 2009

concurrent systems. The aim of the second part is that the participants will gain practical experience in the application of CP-nets and CPN Tools for the modelling and validation of larger concurrent systems. The working methods of the second part are also intended to train the participants to plan and complete projects and to communicate professional issues.

The only prerequisite for the course is that the participants must have completed the first two short introductory programming courses of their bachelor's degree studies. These two programming courses correspond to 10 ECTS. This means that we assume that the participants are familiar with conventional programming-language concepts such as variables, types, procedures, and modules. The overall approach taken in the course is to introduce the CPN modelling language in a way similar to that in which programming languages are introduced, i.e., through concrete examples that illustrate the constructs of the modelling language and also the more general concepts of concurrency, synchronisation, and communication. The course is an optional advanced course, and the majority of the participants are in their third to fifth year of studies when taking the course. The course usually has 15–20 participants. It is important to emphasise that the course is a specialised course on the CPN modelling language and its supporting computer tools. There are several other courses in the curriculum at our computer science department aimed at giving a more general introduction to the theoretical and practical aspects of concurrency. The theoretically oriented courses include courses on automata, concurrency, and model checking introducing the students to labelled transition systems, communicating sequential processes (CSP), the calculus of communicating systems (CCS), and temporal logic. The practically oriented courses include courses on network protocols and internetworking, operating systems, and distributed systems.

15.2 Intended Learning Outcomes

The formulation of the intended learning outcomes of the course is based upon the *Structure of the Observed Learning Outcome (SOLO) taxonomy* of Biggs [6], which provides a tool and framework for specifying the learning outcomes of a course. The SOLO taxonomy has five levels, listed in Table 15.1, which determine a hierarchy of learning competences, where level 5 is the highest level. The verbs used in Table 15.1 to characterise the individual levels are very generic terms for learning competences and often need adaptation depending on the educational context in which the SOLO taxonomy is applied. Within our department, a variant of the SOLO taxonomy has been developed with verbs specifically aimed at computer science competences and these will be used below when we present the intended learning outcomes of the course.

The SOLO taxonomy has been adopted by the Faculty of Science at Aarhus University as a general means for formulating learning outcomes. It was introduced at the same time as a new Danish assessment scale with seven grades was introduced and an ECTS certification process was undertaken by Aarhus University. The pur-

Table 15.1 The five levels of the SOLO taxonomy (see [6], pp. 39–40)

Level 5:	**Extended abstract** Characterised by verbs such as *theorise*, *hypothesise*, *generalise*, *reflect*, and *generate*. These verbs represent competences at a level extending beyond what has been dealt with in the actual teaching.
Level 4:	**Relational** Characterised by verbs such as *apply*, *integrate*, *analyse*, and *explain*. These verbs represent competences in orchestrating facts, theory, actions, and purposes.
Level 3:	**Multistructural** Characterised by verbs such as *classify*, *describe*, and *list*. These verbs represents solid competences within each topic and a basic understanding of the boundaries of each topic.
Level 2:	**Unistructural** Characterised by verbs such as *memorise*, *identify*, and *recognise*. These verbs represent a minimalistic, but sufficient understanding of each topic viewed in isolation.
Level 1:	**Prestructural** This is the very bottom level, where no competences have been obtained.

pose of the new grading scale is to measure more explicitly than earlier the extent to which course participants have achieved the *intended learning outcomes* (ILOs). In Tables 15.2 and 15.3, the verbs that map into the five levels of the SOLO taxonomy are highlighted using bold italic type. The SOLO level to which a given verb belongs is written in superscript following the verb. For the first part of the course, seven ILOs, given in Table 15.2, have been defined. These intended learning outcomes express what the participants are expected to be able to do at the end of the course. In the following, we discuss each of these learning outcomes in more detail.

ILO1 (constructs and concepts) is concerned with learning the constructs of the CPN modelling language, which include the net structure, the CPN ML inscription language, and the concepts related to hierarchical and timed CPN models. ILO1 also includes concepts such as binding elements, steps, concurrency, and conflict.

In ILO2 (syntax and semantics), we require the participants to be able to formally define and explain the syntax and semantics of CP-nets. The purpose of ILO2 is to make the participants understand that CP-nets rely on a formal foundation. When they are introduced to the formal definitions, the participants explore CP-nets from a different angle than the example-driven introduction to the language. In this sense, the formal definitions represent a complementary view of the modelling constructs that can help the participants to consolidate their understanding. ILO2 does not require the participants to be able to formally define hierarchical CPN models and timed CPN models. The formal definitions for this limited subset of the CPN modelling language can be introduced using simple mathematical concepts.

In ILO3 (behaviour of concurrent systems), we require the participants to be able to define and explain the standard behavioural properties of CP-nets (such as boundedness properties, dead markings, and live transitions) and quantitative performance properties (such as delays, throughput, and utilisation). These concepts are

Table 15.2 Intended learning outcomes of the first part of the course

ILO1	***Explain***[4] the constructs and concepts of the CPN modelling language.
ILO2	***Define***[2] and ***explain***[4] the syntax and semantics of non-hierarchical untimed CP-nets.
ILO3	***Define***[2] and ***explain***[4] properties used for characterising the behaviour of concurrent systems.
ILO4	***Explain***[4] the basic concepts and techniques underlying state space analysis methods.
ILO5	***Explain***[4] the basic concepts and techniques underlying simulation-based performance analysis.
ILO6	***Apply***[4] CP-nets and CPN Tools to the modelling and validation of small concurrent systems.
ILO7	***Judge***[4] the practical application of CP-nets to the modelling and validation of concurrent systems.

used when the students work with the analysis methods of CP-nets, which include state space analysis and simulation-based performance analysis.

ILO4 (state space analysis) is concerned with the state space analysis methods of CP-nets. Here we require the participants to be able to explain the concepts of state spaces and strongly-connected-component graphs. Furthermore, we require the participants to be able to explain the techniques used to check the standard behavioural properties of CPN models from the state space and the strongly-connected-component graph. Finally, we require that the participants are able to explain the basic ideas underlying the advanced state space methods.

ILO5 (performance analysis) is concerned with simulation-based performance analysis of CPN models. Here we require the participants to be able to explain the techniques underlying simulation-based performance analysis such as workload generation, data collection monitors, and simulation replications. Furthermore, we require the participants to be able to explain the statistical concepts related to discrete- and continuous-time statistics.

ILO6 (modelling and validation of small systems) specifies that the participants must have operational knowledge of the topics taught in the course, i.e., be able to apply the modelling language and the analysis methods in practice.

ILO7 (judging the application of CP-nets) requires the participants to be able to determine whether CP-nets constitute an appropriate choice for modelling and validating systems within a given domain, i.e., to determine whether CP-nets are suitable for the modelling of a system and the validation of the properties of interest.

For the second part of the course, three intended learning outcomes given in Table 15.3, have been defined. ILO8 (modelling of larger systems) and ILO9 (validation of larger systems) require the participants to be able to model and validate concurrent systems of a size and complexity that appear in real system development projects. ILO10 (discussing application of CP-nets) requires the participants to be able to convey the results of modelling and validation, and issues arising from these results, to colleagues.

Table 15.3 Intended learning outcomes of the second part of the course

ILO8	***Construct***[3] and ***structure***[3] CPN models of larger concurrent systems.
ILO9	***Apply***[4] analysis methods for CP-nets to the validation of larger concurrent systems.
ILO10	***Discuss***[5] the application of CP-nets to the modelling and validation of larger concurrent systems.

We discuss the learning outcomes further in the next section, where we explain how the teaching methods were chosen to support the participants in achieving the intended learning outcomes, and how assessment methods were chosen to measure whether the participants had achieved these outcomes.

15.3 Teaching and Assessment Methods

The teaching and assessment methods used in the course were chosen according to the theory of *constructive alignment* [6]. In short, this theory states that the intended learning outcomes should be the focal point of the course and the teaching methods and activities used should be chosen so that they support the participants in achieving these intended learning outcomes. Similarly, the assessment methods used (e.g., the form of the exams) must be chosen so that they measure the degree to which the participants have fulfilled the intended learning outcomes. The overall goal of constructive alignment is to encourage and motivate students to take a *deep approach* to learning in contrast to a *surface approach*. A surface approach is characterised by students doing tasks with a minimum of effort using low-cognitive-level activities, whereas a deep approach to learning is characterised by students actively working with the topics using higher-cognitive-level activities. This means that the focus of constructive alignment is on the processes and products that result from the learning activities of the students. A fun and easy way to learn more about the SOLO taxonomy and the difference between surface learning and deep learning is to watch the award-winning 19-minute short film *Teaching Teaching and Understanding Understanding* [10], which is available via the Internet.

As explained earlier, the course is divided into two parts. The first part of the course has a duration of 7 weeks (called a quarter) and is organised into 14 sessions, as detailed in Table 15.4. The column 'Material' lists the chapter(s) that the lectures are based on. Each session lasts for two hours (2 times 45 minutes). The column 'ILO' lists the intended learning outcome addressed in the session. It can be seen that the course is a combination of lectures and workshops. In the workshops, the participants work in groups of two or three persons in front of a PC using CPN Tools to solve exercises and projects. The lecturers are present to help with technical questions and issues related to the projects and exercises. In our experience, these workshops are very useful, as they enable face-to-face discussion with

Table 15.4 Sessions in the first part of the course

Session	Topic	Method	Material	ILO	Projects
1	Why modelling and validation?	Lecture	Chap. 1	7	
2	Non-hierarchical CP-nets	Lecture	Chap. 2	1+3	
3	CPN ML programming	Lecture	Chap. 3	1	
4	Practical modelling	Workshop	Exercises	1+3+6	Start P1
5	Formal definition of CP-nets	Lecture	Chap. 4	2	
6	Practical modelling	Workshop	Project 1	1+3+6	
7	Hierarchical CP-nets	Lecture	Chap. 5	1	End P1
8	State space analysis (1)	Lecture	Chaps. 7+8	3+4	Start P2
9	State space analysis (2)	Lecture	Chaps. 7+8	3+4	
10	Practical state space analysis	Workshop	Project 2	3+4+6	
11	Timed CP-nets	Lecture	Chap. 10	1	End P2
12	Performance analysis	Lecture	Chap. 12	3+5	Start P3
13	Practical performance analysis	Workshop	Project 3	3+5+6	
14	Industrial applications	Lecture	Chap. 14	7	End P3

the participants and are effective in highlighting issues that need to be discussed in more detail – and which can then be discussed on demand at the workshops. In this respect the workshops facilitate an interactive teaching–learning environment. The workshops support the intended learning outcomes of the course, in particular ILO6 (modelling and validation of small concurrent systems), but the workshops also facilitate learning outcomes ILO1, ILO3, ILO4, and ILO5 as they stimulate discussions among the participants of the concepts covered.

There are three mandatory projects in the first part of the course: project 1, on modelling; project 2, on state space analysis; and project 3, on performance analysis. The projects start and end as indicated in column "Projects" of Table 15.4. The projects are conducted in groups of two or three participants and have to be documented in a short 5–10 page written group report. The first project is concerned with extending the CPN model of the protocol system shown in Fig. 2.10 to model a sliding-window protocol. The model of the sliding-window protocol must be validated using simulation. The second project is concerned with conducting state space analysis of the model developed in project 1 in order to verify the correctness of the protocol. It is interesting that 50–75% of the groups usually discover errors in their design of the sliding window protocol from project 1 – errors that were not discovered by means of the simulation conducted as part of project 1. This means that the participants experience at first hand the power of verification techniques such as the use of state spaces. Finally, project 3 is concerned with analysing the performance of the sliding-window protocol created in project 1 using simulation and comparing it with the performance of the protocol system described in Chap. 12. The three projects must be approved before a participant can enrol for the exam. This ensures that the participants have fulfilled learning outcome ILO6 (modelling and validation of small concurrent systems) before taking the exam. The exam is a 20-minute oral exam and the participants have approximately one week for preparation for the exam. In the exam, each examinee draws one question, covering ILO1–5 and ILO7. Table 15.5 lists the topics of the exam questions. Each question corresponds to a chapter in this textbook.

Table 15.5 Exam questions for the first part of the course

Question	Topic
1	Non-hierarchical Coloured Petri Nets (Chap. 2)
2	Formal definition of non-hierarchical Coloured Petri Nets (Chap. 4)
3	Hierarchical Coloured Petri Nets (Chap. 5)
4	State spaces and behavioural properties (Chap. 7)
5	Advanced state space methods (Chap. 8)
6	Timed Coloured Petri Nets (Chap. 10)
7	Simulation-based performance analysis (Chap. 12)
8	Industrial applications (Chap. 14)

The second part of the course is organised in a different manner, as the main aim is to train participants in the modelling and validation of larger concurrent systems. In this part of the course, the participants conduct a larger modelling and validation project. There is a high degree of freedom in defining the project which is to be done in groups of two to three persons. During the second part of the course there are no conventional lectures, but there are two progress workshops where the groups give a 25-minute oral presentation of the current state of their project. In the first progress workshop, the focus is on the modelling, and the groups discuss their models with the lecturers and the other participants, who provide feedback. In the second progress workshop, the focus is on the validation part of the project. The project is typically based on a natural-language description of a larger concurrent system. The following is a partial list of the systems that have served as a basis for projects:

- *Distributed file systems*. This project was based upon Chapter 8 of the textbook [24].
- *Dynamic Host Configuration Protocol (DHCP)*. This project was based upon the IETF Request for Comments document 2131 [31].
- *Data dissemination protocol*. This project was based upon the paper [12].
- *Dynamic MANET On-demand (DYMO) routing protocol*. This project was based upon the IETF Internet-Draft [16].
- *Internet Key Exchange (IKE) protocol*. This project was based upon the IETF Request for Comments document 6306 [66].
- *Mutual exclusion algorithms*. This project was based upon selected algorithms from the textbook [92].

- *PathFinder scheduling mechanism.* This project was based upon a description that can be found in the paper [53].

Each year we provide a set of five to ten project proposals, but participants may also choose other systems as a basis for their projects. Many of the projects have focused on communication protocols and distributed algorithms, but it is possible to choose systems from other domains such as workflow systems, manufacturing systems, and embedded systems. In the next section, we give an example of a representative project conducted during the second part of the course.

The assessment of the second part of the course is based on an evaluation of a written group report, which is required to have a length of 15–20 pages, together with an individual oral exam, where each participant is required to give a presentation of the group project. The final grade is the average of the grade for the written report and the grade for the oral performance. The act of constructing and validating a larger model supports ILO8 (modelling of larger systems) and ILO9 (validation of larger systems), whereas the progress presentations and the exam support ILO10 (discussing the application of CP-nets).

15.4 Example of a Student Project from the Course

As a representative example of a project conducted during the second part of the course, we consider a project carried out by a student group on modelling and validation of the Dynamic MANET On-demand (DYMO) protocol [16]. A mobile ad hoc network (MANET) is an infrastructureless wireless network consisting of a set of mobile nodes, where multihop communication is supported by the individual mobile nodes, acting as routers. DYMO is a routing protocol that is being developed by the IETF MANET working group [80] and is specified in a 35-page 'Internet-draft' giving a natural-language specification of the protocol.

Figure 15.1 shows the module hierarchy. We have omitted the names of the substitution transitions on the arcs, since the name of each substitution transition is identical to that of the submodule associated with the substitution transition. The complete CPN model is a medium-sized model consisting of 9 modules, 18 transitions, 45 places, 17 colour sets, and 20 CPN ML functions.

The CPN model is divided into four main parts. The ApplicationLayer module represents the applications that use the multihop routes established by the DYMOLayer module. The NetworkLayer module models the transmission of packets over the underlying mobile network, and the Topology module models the mobility of the nodes which causes the topology of the MANET to be dynamic. Figure 15.2 shows the MANET module which is the top-level module of the CPN model.

Figure 15.3 depicts the ProcessRREQ module, which is an example of a module at the lowest level in the CPN model. It models the processing of route reply (RREP) messages by the mobile nodes. Messages from the underlying network arrive at the place NetworktoDYMO at the lower right. The module captures the two possible cases that can arise when an RREP message is received: either the RREP message

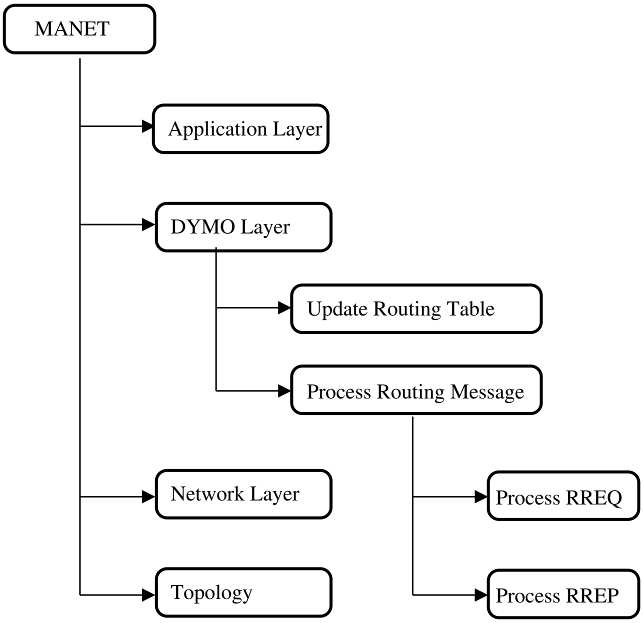

Fig. 15.1 Module hierarchy of the DYMO protocol model

has to be forwarded to the next destination address, i.e., the next mobile node on the route being established, or the mobile node is the target for the RREP. These two cases are modelled by the accordingly named transitions. If the RREP is to be forwarded, it is put on the place DYMOtoNetwork. If the mobile node is the target for the RREP, the message is put on the place ReceivedRREPs for further processing (not modelled).

The CPN model constructed captures a large subset of the DYMO protocol specification. Through the modelling the students demonstrated that they were able to take a complex system (in this case the DYMO protocol) and construct a CPN model at a good level of abstraction (see ILO8, modelling of larger systems). Furthermore, they showed that they were able to divide the CPN model into modules which naturally reflected the various operations of the protocol. In the process of constructing the CPN model, the students discovered several ambiguities and missing parts in the DYMO specification, and they used state space analysis to investigate non-trivial issues related to the operation of the DYMO protocol (see ILO9, validation of larger systems). The project was documented in a 20-page written report that introduced the basic operation of the DYMO protocol, presented the CPN model and the assumptions made in the modelling, and discussed the simulation and state space analysis results obtained (see ILO10, discussing the application of CP-nets for larger systems).

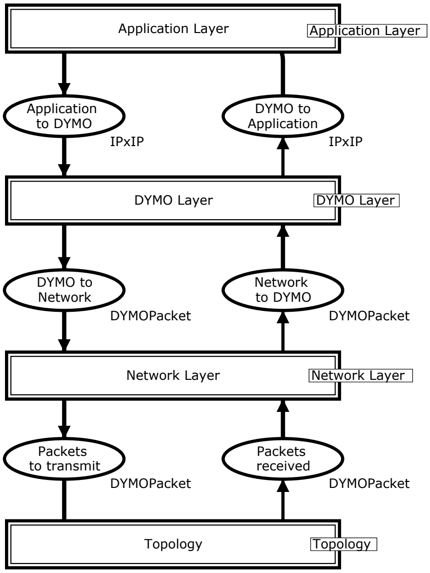

Fig. 15.2 MANET module: top-level module of the DYMO protocol model

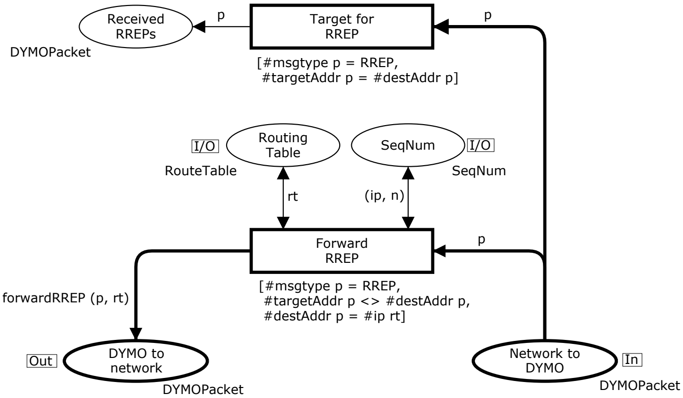

Fig. 15.3 ProcessRREP module, describing the processing of RREP messages

15.5 Experiences from Teaching the CPN Course

The course was developed in conjunction with this textbook, and we have gradually refined and revised the course material and the textbook based upon feedback received from course participants and our own experiences.

At the end of both parts of the course, we spend approximately 30 minutes with the participants on evaluating and discussing the course in an informal way. This informal evaluation is supplemented by a more formal on-line evaluation of the course organised by the Faculty of Science. Unfortunately, it is typical for all courses at the Faculty of Science that only a few participants fill out the on-line evaluation form. Table 15.5 provides a representative summary of the formal evaluation for one of the years in which the course was given. Altogether there were eight participants who filled out the on-line evaluation form, and each asterisk in a table entry represents the feedback of one participant. This means that a single participant represents 12.5% of the replies, and the evaluation results should therefore be interpreted with some care. Nevertheless, the feedback provides a useful indication of the participants' views of the course.

The evaluations that we have received are in general very positive. In terms of achieving the course goals, content, and level of interest, the participants are positive. It is also interesting to observe that the participants do not find the course to be particularly difficult. The participants are expected to spend one-third 1/3 of their study time on the course, which is approximately 15 hours per week, but the feedback shows that they spend less. This is probably related to the participants not finding the course difficult, which in turn may be related to the workshops, where the participants can work on their projects under our supervision. Issues that may arise can thereby be resolved quickly. Participants are also positive with respect to the learning outcomes, the lectures, the workshops, and the textbook. The overall evaluation of the course is also positive.

Compared with the old CPN textbooks [60, 61, 63] and the way we taught CP-nets earlier, we have added more material on the CPN ML programming language. Mastering the CPN ML programming language is important in order for the participants to be able to apply the CPN modelling language successfully to the modelling and validation of concurrent systems.

We have made the deliberate choice of introducing CP-nets directly without first introducing ordinary Petri nets (e.g., Place/Transitions Nets). The main benefit of this is that it enables us to use realistic model examples from the very beginning of the course without having to model data manipulation in an unnatural way using the net structure. Demonstrating that realistic examples can be modelled using relatively simple CPN ML constructs is a factor which contributes to the motivation of the participants. Our teaching activities rely heavily on the integrated use of CPN Tools. This choice is deliberate as it is, in our view, a very motivating factor for the participants and it encourages the participants to work actively with the topics.

A key characteristic of CP-nets is that the language has few but powerful modelling constructs. This is an advantage from a teaching perspective since there are relatively few concepts that have to be introduced and mastered. It is also to some extent a disadvantage in practical modelling, since certain parts of systems cannot be modelled in a straightforward, natural way. A further development of the CPN modelling language and CPN Tools to include constructs such as queueing places, capacities, and module parameterisation is therefore of interest also from a didactic perspective and would improve its convenience of use for modelling.

The first part of the course relies heavily on the protocol model that we have used as a running example. In the second part of the course, we have observed that it takes some efforts from the participants to get started on their own modelling project,

Table 15.6 Summary of on-line participant evaluation

	To a very large extent	To a large extent	To some extent	To a lesser extent	Not at all
Were course goals achieved?	*	*******			
Did content match description?	**	*****	*		
Was the course interesting?	*	*****	**		
Was the course difficult?			***	****	*
Hours spend	<4	5–8	9–12	13–16	17–20
		***	*****		
	Very good	Good	Either way	Bad	Very bad
Learning outcomes	*	******		*	
Lectures	*	*****	**		
Workshops	*	***	****		
Textbook	*	******	*		
Overall evaluation	*	*******			

which is concerned with a different system and sometimes lies within a different application domain. In courses with sufficient time it is therefore recommended that additional examples of CPN modelling should be integrated in the course. A number of small and medium examples can be obtained from the CPN Tools Web pages, and a list of larger examples from the literature is available via [40]. Altogether, this can give participants a broader perspective on CPN modelling and validation. This is useful when the participants are working on their final project, and hence are facing the challenges of modelling a larger system. It also contributes to ILO7 (judging the application of CP-nets).

As described above, we have recently adapted the theory of constructive alignment and the SOLO taxonomy [6] for describing course aims and learning outcomes of the course at our department. This has not prompted major changes to the way the course is being taught, but it has been very helpful in making the learning outcomes of the courses much more explicit than earlier. In our opinion, the SOLO taxonomy and constructive alignment provide a very good and practically applicable framework for reflecting upon the teaching and assessment methods used in a course.