

D29.2.1 (Final)

Framework on multiple goal dimensions for computer-supported scripts

Main author : Lars Kobbe (KMRC)

Nature of the deliverable : Report

Dissemination level : Restricted

Planned delivery date : November 2005

No part of this document may be distributed outside the consortium / EC without written permission from the project co-ordinator

Prepared for the European Commission, DG INFSO, under contract N°. IST 507838
as a deliverable from WP29
Submitted on 30-11-2005

Summary

Collaboration scripts aim to facilitate effective interaction patterns for collaborative learning that do not occur spontaneously. So far, diverse non-generic scripts have been conceptualized and investigated in CSCL environments. The specification of collaboration scripts aims to provide a common terminolgy for describing scripts and to abstract the core design principles of scripts to better understand effects and mechanisms of collaboration scripts and to apply and re-apply collaboration scripts in different learning environments.

Beginning with a review of the original conception of scripts for collaborative learning and its evolution in context of CSCL, we present recent approaches towards a specification of scripts by Dillenbourg (2002), Dillenbourg and Jermann (in press) and Kollar, Fischer and Hesse (in press). The framework that we propose here can be regarded as an expanded and revised consolidation of these efforts. We then discuss the kind of activities that the instructional approach of scripting collaboration aims to foster. Our paper subsequently presents the components and mechanisms that constitute a script and illustrates them with example scripts. We finally give an outlook on the future task of formalizing scripts and conclude with a brief evaluation of the framework.

History

Filename	Status	Release	Changes	Uploaded
D29-02-01-F.pdf	Final	1		30/11/2005
	Draft	1		01/01/1970

Table of Contents

1 The original conception of collaboration scripts and its development 3

2 Approaches to the specification of collaboration scripts 4

3 Specifying activities for learning 6

4 Specifying collaboration scripts – a framework 8

4. 1 Script components 9

4. 2 Script mechanisms 11

4.3 Script examples 12

5 Towards a formalization of collaboration scripts 19

6 Conclusions 20

References 21

Appendix: Guidelines for describing collaboration scripts 24

Framework for the Specification of Collaboration Scripts

Successful collaborative learning relies on effective interaction of learners. However, when learners are left to their own devices, they rarely engage in asking each other questions, explaining and justifying their opinions, articulating their reasoning, or elaborating and reflecting upon their knowledge. Collaboration scripts aim to support these learning activities by structuring otherwise deficient interaction. In cognitive psychology, the term "script" is used to refer to personal or culturally shared knowledge about everyday situations in the form of generalized procedures (Schank & Abelson, 1977). For instance, the "restaurant script" informs us what to do and to expect when we go to a restaurant, the roles we and other participants play, and the sequence in which all events are supposed to take place. In educational science, collaboration scripts are given activity programs that structure the interaction of collaborative learners (O'Donnell & Dansereau, 1992). Similar to the "restaurant script", collaboration scripts can guide the learners in what to do in the learning task, the roles they play, as well as the sequence of activities to engage in. Since then, this approach of what they called "scripted cooperation" has been adopted by many researchers and educators in the field of computer-supported collaborative learning (CSCL) and produced a wide range of innovative yet non-generic script examples. Recently, computer scientists have aimed to model and formalize scripts in order to foster reusability and portability between different computer-supported learning platforms. However, the approaches that are referred to as scripts today are highly diverse (e.g. practically anything that qualifies as a prescribed sequence of activities) and highlight the need for a common conceptual framework that can be referred to in research and development of collaboration scripts.

Beginning with a review of the original conception of scripts for collaborative learning and its evolution in context of CSCL, we will present recent approaches towards a specification of scripts by Dillenbourg (2002), Dillenbourg and Jermann (in press) and Kollar, Fischer and Hesse (in press). The framework that we propose here can be regarded as an expanded and revised consolidation of these efforts. We will then discuss the kind of activities that the instructional approach of scripting collaboration aims to foster. Our paper will subsequently present the components and mechanisms that constitute a script and illustrate them with example scripts. We will finally give an outlook on the future task of formalizing scripts and conclude with a brief evaluation of the framework.

1 The original conception of collaboration scripts and its development

Scripted cooperation, as described by O'Donnell (1999), differs from other collaborative learning methods chiefly in that scripts specify the cognitive activities that learners are expected to engage in, whereas most other methods leave them unspecified or vague. Thus, scripts aim to promote engagement in activities that would otherwise occur rarely or not at all. By specifying activities that have emerged from research findings in cognitive and educational psychology as being strongly related to learning (e.g. Cohen, 1994, Webb & Palincsar, 1996), scripts are assumed to lead to higher level cognitive processing and therefore to better learning outcomes. Furthermore, participants are asked to play specific roles that are switched during the course of a script in order to provide equal learning opportunities and a more balanced interaction. For example, the "MURDER"-script designed by Dansereau and colleagues (1979) for text processing specifies a sequence of comprehension fostering activities that the learners go through once for each passage of text to read (cf. table 1).

Table 1. The "MURDER"-script by Dansereau et al. (1979).

Preparation:
An even amount of students are grouped into pairs. Each group has the same set of text passages available for reading. In each pair, one student assumes the role of summarizer and the other student the role of listener.
Procedure:
Within each pair and for each passage of text, …
… both students set the **M**ood for studying.
… both students read the text material for **U**nderstanding.
… the summarizer **R**ecalls the material.
… the listener **D**etects errors/omissions and gives feedback.
… both students **E**laborate on the learning material.
… both students **R**eview the learning material and what they have learned.
-- after each cycle, students switch roles --

While scripts were originally used in the context of individuals or independent small groups, their use was later extended to classroom settings with interacting small groups and alternating phases of individual, group and classwide activities (MOSIL group, 2004). With

the advent of new technologies such as the internet and mobile computing, an increasing amount of scripts were designed that made use of computers as a support for collaborative learning. Scripts featuring asynchronous or quasi-synchronous communication tools, for instance, offered a larger scope of duration and granularity of specification and sequencing of roles and activities on the basis of single conversational turns up to collaborative phases of several weeks or more. And besides instructing students in advance in how to follow a collaborative learning script, computer support could also facilitate prompting and guiding students step-by-step on their way through the script.

2 Approaches to the specification of collaboration scripts

In a pioneering attempt to analyze the different kinds of computer-supported scenarios that were regarded as collaboration scripts, Dillenbourg (2002) identified a number of aspects that served as a preliminary framework for script comparison and design. Dillenbourg (2002) described scripts as a sequence of phases, each characterized by five attributes: type of task to be accomplished, group formation and composition, distribution of task within and among groups, type and mode of interaction/communication (e.g., co-located vs. remote, synchronous vs. asynchronous, text-based vs. voice-based, etc.), and timing/duration of the phase. From phase to phase, each of these attributes can change. The allocation and re-allocation of roles and activities, as well as physical or virtual resources are considered to be part of the task distribution.

In their conceptual framework, Dillenbourg and Jermann (in press) expanded the scope of collaboration scripts to encompass more than just small group interaction. While collaborative activities are still being regarded as essential to the learning process and constitute the core learning mechanism ,“integrative scripts” also include individual and class-wide activities. Thus, collaborative “core scripts” are positioned within a “didactic envelope”, i.e. pre- and post-structuring activities that enable scripts to blend optimally into the lesson plan (e.g. introducing the topic, reflecting on what was discussed, etc.) and contribute to their effectiveness and consistency. The integrative aspect of scripts becomes even more evident in the case of computer-supported collaboration scripts, in which scripts integrate virtual with physical activities and manage the data flow between them. For example, in the ArgueGraph script (see table 4c), class-wide discussion is based on a graph that was computed from priorly collected individual reponses to a questionnaire.

According to Dillenbourg and Jermann (in press), the didactic envelope features two salient dimensions: The temporal and social structure of scripts. Each activity in a script can be mapped to a certain phase or point of time in the overall sequence and is also limited in its duration (time structure). Furthermore, each activity can be mapped to a specific "social plane" (in reference to Vygotsky, 1978), such as an individual, group, class, community or world plane (social structure). The didactic envelope basically manages the integration of activities from phase to phase and from plane to plane.

The core script typically involves a task distribution among all group members. Dillenbourg and Jermann (in press) viewed interaction as the means to overcome these task splits and proposed a model according to which the nature of the task distribution also determines the nature of the members' interaction (e.g., explanation, argumentation, mutual regulation, etc.). Following Schwartz' (1995) definition of collaborative learning as the effort necessary to build a shared understanding, learning would result from the interactions students engage in to build a shared understanding despite the fact that the task is distributed (Dillenbourg & Jermann, in press). According to the design principle "Split Where Interaction Should Happen", the model was named "SWISH".

Kollar, Fischer and Hesse (in press) were interested in comparing scripts from two research traditions: Computer-supported collaborative learning (CSCL) and instructional psychology. Five conceptual components were identified on which scripts could be differentiated: *Objectives*, *activities*, *sequencing*, *roles*, and *type of representation*. Scripts from both research traditions had similar objectives regarding the group task. However, as all of the CSCL scripts made use of computer-mediated communication, a major part of these scripts was devoted to the support of smooth and coherent communication and coordination in respect to the inherent weaknesses of the medium (e.g. Herring, 1999). Although scripts did not differ much in the kind of activities that were promoted (e.g. argumenting, explaining, question asking), scripts from instructional psychology gave much more support on how to carry out these activites (through training or detailed instruction). CSCL scripts typically lacked a prescribed sequential structuring of activities, although the interface sometimes implicitly induced a certain sequence of activities. Also, CSCL scripts often lacked clear role distribution in contrast to scripts from instructional psychology. Scripts from both traditions were rather inconsistent in the way they were represented during the learning phase (e.g. represented internally in the learners' mind or represented externally by buttons on the screen,

prompt cards, etc.). These findings point out the diversity of scripts both in realization as well as in function and purpose.

In a conceptual framework, Kollar et al. (in press) used a distributed cognition approach to incorporate the five components mentioned before: Since the group task is solved or the *objective* is reached by engaging in certain activities, it can also be defined by a specific set of activities. In a script, this set of activities can be conceived of as being distributed among the learners as distinct *activities* or a bundle of activities in the form of *roles*. These activities and roles are linked by a specific *sequence* that follows a meaningful strategy and ultimately leads to the solution of the task. The script is to some degree *represented* externally (e.g. written, graphical or oral) and internally (as procedural knowledge).

3 Specifying activities for learning

As O'Donnell and Dansereau (1999) pointed out in their instructional approach, rather than setting the conditions for collaborative learning (i.e. positive interdependence and individual accountability; Johnson & Johnson, 1998; Slavin, 1983), scripts are essentially about activities that promote learning from a cognitive perspective. We therefore need to specify what kinds of activity qualify as such.

Cognitive psychologists typically differentiate between learning activities that either promote lower-order or higher-order thinking, though the distinction is not so clear cut (cf. Lewis & Smith, 1993). Following King (in press), lower-order thinking is associated with activities such as repetition, rehearsal and retelling, which are merely effective in memorizing factual material. Summarizing and paraphrasing are considered effective for promoting understanding and demonstrating comprehension. Higher-order thinking goes beyond memorization and comprehension in that it involves more complex cognitive processes. According to King (in press), activities that research has found to be particularly effective in promoting such cognitive processes include (cf. table 2): elaborating on content (Webb, 1989); explaining ideas and concepts (Chi, deLeeuw, Chiu & LaVancher, 1994); asking thought-provoking questions (King, 1994); constructing arguments (Kuhn, 1991), resolving conceptual discrepancies (Piaget, 1985) and cognitive modeling (Bandura, 1997). These activities can be carried out effectively by an individual learner as in the case of "self-explanation" (Chi et al., 1994) and "self-questioning" (King, 1989), by learning partners (e.g.

asking someone else) as well as collaboratively through joint knowledge construction (e.g. several learners building on each other's reasoning to explain a concept).

Besides single activites, there are also effective combinations of activities such as question-asking and answering or constructing arguments and counter-arguments. By distributing complementary activities that induce learners to monitor and support each other's learning activities, metacognitive processes are fostered that are vital to all learning activities. Thus, the effectiveness of summarizing, for instance, can be greatly enhanced by pairing the summarizer with an active listener who monitors and encourages the partner for a high quality of the summary. A list of effective sets of activities may emerge from a careful meta-analysis of collaborative learning studies, since assigning complementary activities to learners is a common practice in scripts as well as other collaborative learning methods.

From a strictly cognitive perspective of learning, the specification of effective learning activities is an improvement to merely setting the conditions for collaborative learning, but still does not ensure an adequate level of cognitive processing. The engagement in these activities may for instance lack epistemic quality (appropriate strategy), elaborateness (richness and interrelation of information) or transactivity (operating on each other's reasoning). Furthermore, learners often lack the knowledge or skill for formulating good arguments or questions. Therefore, quite a few scripts explicate the activities to engage in by decomposing them into a sequence of more fine-grained activities and providing additional instructional support such as prompts or illustrations.

Table 2. Examples of learning activities recommended for collaboration scripts.

asking thought-provoking questions
clarifying ideas and relations
comparing concepts
constructing arguments
critiquing proposals
drawing conclusions
elaborating on content
evaluating the significance of findings
explaining ideas and concepts
predicting consequences

4 Specifying collaboration scripts – a framework

Based on the presented frameworks by Dillenbourg (2002), Dillenbourg and Jermann (in press) and Kollar et al. (in press), we want to propose a framework that consolidates recent approaches with current ideas and conceptualizations of our research group.

Our framework distinguishes between four different levels of abstraction for the specification of scripts as defined by Dillenbourg and Jermann (in press): Script schemata, classes, instances and sessions. Script schemata are set at the highest level of abstraction and describe the design rationale or the core design principle through which the script is expected to trigger specific interactions. For instance, many scripts use a variation of the Jigsaw method and form pairs with complementary knowledge, provide them with complementary information or roles in order to create socio-cognitive conflict, or assign and alternate roles that foster reciprocal activities such as questioning or tutoring. Thus, these core design principles are called the Jigsaw schema, the conflict schema and the reciprocal schema (Dillenbourg & Jermann, in press). We use the term ‘schema’ instead of 'pattern' to avoid confusions with the term 'design pattern', which has a more technical meaning in software engineering. Besides these schemata, several other schemata are expected to exist that we aim to identify in the future.

On a lower level of abstraction, script classes can be constructed from a script schema. A script class covers a range of scripts that represent acceptable variations of a prototype, e.g., the ArgueGraph script (cf. table 4c) and the ConceptGrid (cf. table 4f) script are prototypes of the conflict schema, whereas the Universanté Script (cf. table 4b) is a prototype of the Jigsaw schema. This raises the basic question: which variations of a class prototype preserve the core principle? This core principle cannot be modified without affecting the effectiveness of the script. For instance, in the ArgueGraph script, selecting peers with similar opinions instead of opposite opinions would counteract the conflict generation principle. Script classes are meant to be highly generalizable, in particular with regard to the amount of participants and resources needed as well as the content it is used with. The amount of participants and resources may be set to a threshold for minimum and maximum values, but may also be interdependent on each other (e.g., one case description per participant) or on other variables such as desired group size (e.g. an even amount of participants for dyads) or available roles. The limitation of content generalizability does not simply reflect domains (maths, history, …) but the affordances of specific contents with respect to the core design principle (can it be argued about, can it be explained, etc.). For instance, the ArgueGraph script (cf. table 4b)

could be used with multiple contents, but not with any contents. It is obviously only relevant when the topic at hand can be argued about.

Beyond the first two levels of abstraction, script instances and script sessions can be distinguished. A script instance is a script that has been instantiated with a specific content. At this level, many further design decisions are made, e.g. how collaboration scripts are implemented and externally represented, to what extent learners are informed about the script beforehand, whether and how the learners are guided through the script, etc. A script session is the most concrete form of scripts in that it makes use of concrete details (e.g., the dates, the names of students, the duration of phases, etc.) that are relevant for running the script in one particular session. Thus, script sessions are hardly reusable at all.

Our current framework is constrained to a specification of scripts on the level of script classes only, because scripts at this level can be specified in enough detail to allow for formalization and later implementation whereas script schemata are too scant on details and many of the design choices made at the level of script instances are initially irrelevant for formalization.

Regarding the structural composition of scripts, we distinguish between mechanisms on the one hand and components on the other. This distinction has not been made in earlier approaches. For example, Kollar et al. (in press) regarded “role distribution” as a script component, whereas we refer to roles as a component and distribution as a mechanism. Furthermore, we do not regard earlier components like “type of representation” or “mode of interaction“ as either mechanism or component, because these aspects are merely design decisions that can be used to build variants of one and the same script (cf. script instances). Script components and mechanisms will be detailed in the following sections.

4. 1 Script components

Our framework allows scripts to be described with a small number of components: The individuals that participate in a script, the activities that they engage in, the roles they assume, the resources that they make use of and the groups they form.

Groups

Groups form a hierarchical structure with larger groups being composed of one or more smaller groups. Participants can be grouped by referring to existing common features (e.g.

gender, age or national groups) or can be distributed to new groups based on certain criteria such as desired group size, amount or composition (see group formation mechanism in section 4.2).

Participants

Participants are used in synonym with users, persons or people, i.e. it is used as a general abstraction of concrete individuals. Scripts usually have certain requirements regarding the numer of participants that they can handle, sometimes given as a variable range (e.g. from 3 to 8 participants) or as a multiple of another script component (e.g. 2 participants per text book).

Roles

The main function of roles in collaboration scripts is to refer to specific participants when assigning activities or allocating resources. Roles are also associated with privileges, obligations and expectancies. For example, someone in the role of a teacher has the privilege to evaluate a student's work, the obligation to administer an exam and is expected to help the student in preparing for the exam. As roles are closely tied to learning activites, their title is usually predictive of the activities which participants are allowed, expected or oblieged to engage in. Thus, specific roles may foster particular learning activities, such as a "scientist" promoting planning, observing and drawing conclusions. Participants may assume one or more roles at a time and can exchange these roles with other participants.

Activities

Activities form a hierarchical structure in which any greater activity can be decomposed into lesser, more fine-grained activities, and any lesser activity can be subsumed by one or more greater, more low-grained activities. For instance, discussing can be decomposed into explaining, constructing arguments, question asking, etc., and asking somebody to check a report for mistakes can be generalized as help-seeking. The kinds of activities specified are of high relevance to the type and degree of learning taking place (see section 3).

Resources

Resources comprise virtual or physical objects that can be allocated to learners (e.g. text books, tools, etc.). Although resources have not been identified as script components in earlier

approaches, they are important to the context of scripts because they often constitute a common object of or reason for interaction. Resources often foster interaction by being distributed among learners in a way that induces social interdependence (i.e. each learner needs access to each other's resources), a method commonly employed in collaborative learning scenarios such as Jigsaw and conceptualized in the SWISH-model (cf. section 2) by Dillenbourg and Jermann (in press).

4. 2 Script mechanisms

Script mechanisms help to describe the distributed nature of scripts, i.e. how individual learners are distributed over groups (*group formation*), how components are distributed over participants (*component distribution*) and how both components and groups are distributed over time (*sequencing*). Each of these mechanisms features particular principles that are important for issues of scaleability (e.g. for applying the same script to a varying number of participants) and modeling (see section 5).

Group Formation

While some groups just happen to exist by definition (e.g. gender groups such as men and women or hair color groups such as blondes, brunettes and redheads), other groups need to be formed by a particular procedure or principle. In most cases, group formation is very simple, such as forming groups by amount (e.g. dividing a class into four groups), by size (e.g. dividing a class into groups of four) or by a combination thereof (e.g. forming four groups of four). Some scripts make use of more complicated principles that do not only take into account group numbers and size, but also the composition of each group and the overall balance among the groups. The Universanté script, for instance, forms groups by the principle of obtaining at least one group for each case description with at least two participants from each nation, all participants being balanced evenly over the case groups (cf. table 4b).

Component distribution

A key feature of scripts is the distribution of components such as providing participants with only one part of the information that they need in order to foster knowledge exchange with each other. Also, decomposable activities can be distributed in a way that one learner engages a cognitive activity while the other learner engages in a supportive metacognitive activity. Although distribution can in general be ascribed to a principle of complementarity, it is not always easy to determine which and when script components actually complement each other.

Besides script components, other covert yet important aspects such as knowledge and interdependence may also be distributed in a script. Distribution typically makes use of a function, i.e. the elements of one set (e.g. roles) are mapped through a specific function (e.g. one-to-one) to the elements of another set (e.g. participants). In some cases, this function may become quite complex, such as arranging roles, participants and groups in a way that each participant assumes a different role in every group (cf. the Universanté Script in table 4b).

Sequencing

Beyond prescribing a specific linear sequence of activities, scripts often feature a repetition of activities with minor variations: In the case of the MURDER-script (cf. table 1), it is the text passage and the role allocations that change with each cycle. The most prominent principles underlying such repetitions are traversion, rotation and fading. A traversion describes a repetition in which all elements of one set are looped through, with only one element being in use at a time (the text passage in the MURDER-script). A rotation permutates the order of elements in a given set (the roles in the MURDER-script). Fading refers to features that are gradually added (faded in) or removed (faded out) from a script (in the MURDER-script, setting the mood may just as well be omitted after the initial cycle).

4.3 Script examples

The interplay of components and mechanisms is best demonstrated by examples. The following collaboration scripts (Table 4a to 4f) have been used in research by Weinberger et al. (2005), Dillenbourg and Jermann (in press), Hämälainen and Arvaja (subm.) and are presented here as best practice models for different script schemata. Guidelines for arriving at a semi-formal description like the one demonstrated in these examples are provided in the appendix of this paper.

Table 4a. The “Social Script” (Weinberger et al., 2005)

Components:
Resources & Participants: An equal amount of at least 2 case descriptions and participants.
Groups: Case groups.
Roles: An analyst and a critic.
Activities: [1:] applying theoretical concepts to cases, constructing arguments; [2:] critiquing in the sequence of elicitating clarification, identifying conflicts and constructing (counter-)arguments

Group formation & component distribution:

For each case description one "case group" is formed, composed of all available participants. All case descriptions are distributed evenly among all case groups. Roles are distributed among all participants and among all groups in a way that each participant assumes the role of analyst in one group and the role of critic in all other groups (cf. table 5).

Table 5. Role distribution in the social script. The number before each role signalizes the sequence in which these roles are to be assumed by the individual participant.

	Group 1	*Group 2*	*Group 3*
Participant 1	1. Analyst	2. Critic	3. Critic
Participant 2	3. Critic	1. Analyst	2. Critic
Participant 3	2. Critic	3. Critic	1. Analyst

Sequencing:

Within each case group, …

… the analyst writes a case analysis [1]

-- wait for all case group analysts to be done --

… each critic in turn writes a first critique of the case analysis [2]

-- wait for all case group critics to be done --

… the analyst considers each critique and writes a reply to each in turn [1]

-- wait for all case group analysts to be done --

… each critic in turn reads the reply and writes a second critique [2]

-- wait for all case group critics to be done --

… the analyst considers all critiques and writes a new case analysis [1]

Regarding the sequencing principle: The roles of each participant rotate twice over the course of the script. Each participant begins with the role of analyst and changes his role to the first critic, second critic, etc. This cycle is repeated and ends with each participant resuming the role of analyst.

Table 4b. The "Universanté Script" (Dillenbourg & Jermann, in press)

Components:

Resources: Case descriptions from at least 2 themes, with at least 2 case descriptions per theme.

Participants: Participants from at least 2 nations, with at least as many participants per country as there are case descriptions.

Groups: Theme groups, case groups and country groups.

Roles: none

Activities: [1:] analyzing, discussing; [2:] summarizing/synthesizing, presenting; [3:] analyzing, comparing, discussing; [4:] presenting; [5:] giving feedback, critiquing; [6:] problem solving

Group formation & component distribution:

For each case description one "case group" is formed, composed of at least 1 participant per country, balanced among the groups. All case descriptions are distributed evenly among all case groups. For each theme, one "theme group" is formed, composed of at all "case groups" with case descriptions of this theme.

Sequencing:

Within each case group, all participants discuss the case [1].

Within each country group, the members of each theme group in turn present a synthesis of their case experience [2].

Within each theme group, the members of each country group create a fact sheet concerning the theme's status in their country [2].

Within each theme group, all participants discuss the similarity and difference between the fact sheets of different countries [3].

Within each country group, and for each theme group in turn, …

… the members of the theme group present their fact sheet [4]

… everbody else provides comments on the fact sheets [5]

… the members of the theme group modify their fact sheet according to the comments[2].

Within each case group, all participants propose a solution for the case problem [6].

Table 4c. The "ArgueGraph Script" (Dillenbourg & Jermann, in press)

Components:

Resources: One copy of a questionnaire for each participant and another copy for each small group. The questionnaire consists of multiple-choice questions (all choices are equally valid) with space for providing an argument to justify the choice. One argument sheet per question of the questionnaire and as many copies of these sheets as are needed to provide one copy for each participant.

Participants: An even number of at least 4 participants (works best with 20-30 participants) and a tutor.

Groups: Class group and small groups.

Roles: none

Activities: [1:] judging, writing arguments; [2:] comparing, interpreting, discussing; [3:] negotiating, writing arguments; [4:] explaining, justifying; [5:] summarizing

Group formation & component distribution:

In the "survey" phase, all participants together form the class group and receive one copy of the questionnaire. In the "conflict" and "elaboration" phase, all participants are distributed evenly among groups of two, composed of participants with maximimal difference in their responses to the questionnaire. Each small group receives another copy of the questionnaire.

Sequencing:

"Survey" phase:

Within the class group, all participants individually fill out the first copy of the questionnaire.[1]

The tutor displays the aggregated results of the questionnaire (the participants' choices are plotted as anonymous points in the graph) to the participants.

Within the class group, all participants jointly discuss the displayed results of the questionnaire. [2]

"Conflict" phase:

Small groups are formed based on each participants' responses to the questionnaire.

Within each small group, all participants jointly fill out the second copy of the questionnaire, i.e. they negotiate on a single choice and generate a shared argument for this choice. [3]

"Elaboration" phase:

The tutor collects all questionnaires.

For each question of the questionnaire, the tutor compiles all choices and arguments from each small group on an argument sheet.

For each small group in turn, the tutor asks the participants to comment on their arguments and gives advice on how to relate their arguments to theories and concepts. [4]

"Reflection" phase:

The tutor distributes all copies of the argument sheets among all participants.

Within the class group, each participant individually writes a synthesis of all arguments on the argument sheet, taking into account the advise of the tutor. [5]

Table 4d. The "Case Script" (Hämäläinen & Arvaja, subm.)

Components:

Resources: Theoretical background information and at least 2 case descriptions.

Participants: At least 2 participants for each case group.

Groups: One case group for each case description.

Roles: Analysts and critics.

Activities: [1:] reading; [2:] problem solving, discussing; [3:] writing; [4:] critiquing; [5:] revising

Group Formation:

For each case description one "case group" is formed, composed of at least 2 participants.

Component Distribution:

All case descriptions are distributed among the case groups. Each case group is assigned the role of analyst of their own case and the role of critic of another case.

Sequencing:

Within each case group, …

… each participant reads the case description [1]

… each participant reads the theoretical background information [1]

… all participants jointly discuss the case and decide on a shared solution [2]

… all participants jointly write and submit a proposal for the case problem [3]

-- wait for all case groups to be done with writing a proposal –

… all participants read the proposal of one other group [1]

… all participants jointly write and submit a critique to the proposal [4]

-- wait for all case groups to be done with writing a proposal –

… all participants read the received critique and jointly revise their own proposal [1+5]

Table 4e. The "Secure Script" (Hämäläinen, subm.)

Components:

Resources: a) 1 overarching complex problem with 4 sub-problems, b) 1 domain of expertise for each participant of the group, c) 2 conflicting positions, d) 1 sub-task for each participant of the group. Copies of all resources for each small group.

Participants & Groups: At least 4 participants for each small group.

Roles: Experts in different domains, pro- and contra advocates.

Activities: [1:] problem solving; [2:] reading; [3:] knowledge sharing; [4:] arguing; [5:] coordinating

Group Formation:

Small groups are formed with at least four participants each.

Component Distribution:

All small groups get a copy of the overarching complex problem. In the "Open Collaboration" phase, all small groups get a copy of the first sub-problem. In the "Shared Expertise" phase, all small groups get a copy of the second sub-problem and expert roles are distributed among the small group members. In the "Conflict Resolution" phase, all small groups get a copy of the third sub-problem and pro- and contra-advocate roles (together with information on these two conflicting positions) are distributed evenly among the small group members. In the "Need for Coordination" phase, all small groups get a copy of the fourth sub-problem and the sub-tasks are distributed among the small group members.

Sequencing:

Within each group, all participants are introduced to the overarching complex problem.

"Open Collaboration" phase:

Within each group, all participants jointly attempt to solve the first sub-problem. [1]

"Shared Expertise" phase:

Within each group, …

… each participant studies a different domain of expertise and becomes an expert. [2]

… all experts jointly attempt to solve the second sub-problem. [3]

"Conflict Resolution" phase:

Within each group, …

… each participants studies one position and becomes either pro- or contra-advocate [2]

… all advocates jointly attempt to solve the third sub-problem. [5]

"Need for Coordination" phase:

Within each group, …

… each participant solves a different sub-task on his/her own (all sub-tasks in total solve the fourth sub-problem). [6]

"Integration" phase:

Within each group, all participants jointly attempt to solve the overarching complex problem.[1]

Table 4f. The "ConceptGrid Script" (Dillenbourg & Jermann, in press)

Components:

Resources: a) Background information for each role, b) list of 16 concepts that relate to the background information (4 concepts per background information) c) grid table, in which the concepts and the relations between concepts can be inserted.

Participants: 4 participants for each group. A tutor.

Groups: Small groups.

Roles: 4 roles, based on theoretical background materials (names of real-life scientists are useful).

Activities: [1:] reading; [2:] linking concepts to theories; [3:] defining concepts from a theoretical point of view; [4:] relating concepts to each other

Group Formation:

Small groups of four are formed.

Component Distribution:

Each small group gets a copy of the grid table, the background information for each role and the list of concepts. The roles are distributed among small group members by themselves.

Sequencing:

Within each group, …

… each participant reads background information related to his/her personal role. [1]

… all participants decide on how to distribute the sixteen concepts among their roles.[2]

… each participant writes a definition of the concepts allocated to his/her role. [3]

… all participants discuss and decide how to arrange the concepts into a grid table and how to relate them to each other. [4]

… the tutor reviews the grids and points out omissions, inconsistencies and errors.

---

5 Towards a formalization of collaboration scripts

From a computer science perspective, a framework for the specification of collaboration scripts is expected to facilitate the process of formalizing collaboration scripts in order to enable computer-supported learning environments to make use of scripts. Although the script examples presented above aim to use uniform terms and wording, it is obvious that natural language is ill-suited for the task of providing the kind of accurate and unambiguous descriptions of collaboration scripts that is needed for computers. However, the development of such a formal, machine-readable language is also quite challenging.

First of all, CSCL scripts are defined on a wide range of granularity, from coarse-grained pedagogical scripts that define whole learning processes to fine-grained scripts that scaffold the learners, when specific skills, such as how to construct argumentative sequences properly, are lacking. Thus, a formal language has to enable modelling of scripts at different levels of granularity; hierarchical combination (Harrer, Malzahn, Jermann, Dillenbourg, 2005) as well as composition of separate components are required properties for the target language. With that, a stepwise refinement can be achieved by embedding fine-grained script segments into coarser grained segments of a script.

Furthermore, the language used for formalization must refer to the concepts the practitioner is familiar with. This is especially important to enable non-computer scientists to use this formal

language for describing collaboration scripts. Since our framework defines these concepts and takes into account well-established theories, the formal language will use the framework concepts as basic constructs and offer methods to give a more detailed description of the scripts. In addition to textual representations of scripts, graphical notations for representing scripts can be used to capture the core design principle of collaboration scripts.

Finally, for the use in computer-supported learning environments, the formal description of the script has to be mapped to an operational level, i.e. each language construct must have a well-defined semantics. This enables a computer system to interpret and execute all specified elements of a given formalized script exactly and unambiguously. The semantics has to be defined exactly for the chosen language or has to be mapped to an existing precise semantics, e.g. the statechart semantics (Harel & Naamad, 1996).

6 Conclusions

Beginning with a review of the original conception of scripts for collaborative learning, we presented recent approaches towards a specification of collaboration scripts. Since the instructional approach of scripting interaction is essentially a cognitive approach, we discussed what kind of activities foster learning from the point of view of cognitive psychologists. Besides activities, we also introduced a number of other essential components and mechanisms that constitute a script. The interplay of components and mechanisms was demonstrated with script examples that represented prototypes of script schemata.

The proposed framework serves a variety of different purposes. First of all, it provides a common terminology as the cornerstone for knowledge exchange and accumulation in scientific research on scripting collaborative learning. It is aimed at facilitating the systematic exploration of script mechanisms as well as the comparison and integration of research results. Nevertheless, the framework does not impose a specific theoretical perspective on the interplay of script components and mechanisms but is intended to be useful with a wide range of theories. For practitioners, the components and mechanisms elaborated here may serve as a checklist for the design of scripted CSCL environments, as they are of high relevance to collaborative learning. We also expect the prototypical examples to be useful as models of good practice. The guidelines for the description of scripts that were used to semi-formally represent scripts might prove helpful as well. Ideally, they could help practitioners and

designers in better understanding commonalities and differences in already existing or newly developed, alternative scripts for collaborative scenarios.

In the near future, we aim to go beyond the semi-formal description demonstrated in the script examples above to a formal modelling language that can be used in computer-supported learning environments. Since the linear structure of text-only representations is ill-suited to model concepts such as rotation, fading or traversion, at least part of the formal language will use a graphical notation. The use of statecharts diagrams for the modelling of collaboration scripts is currently being explored and has so far shown promising results.

Nevertheless, we believe that a semi-formal script description is more than just a transitional step in the formalization process and has a value of its own: Many researchers and practitioners (even in the field of technology enhanced learning) may not find themselves at ease with reading formal, computational languages, let alone applying this formalism to write down scripts they have designed themselves. Thus, the semi-formal description may serve as an intermediate representation that bridges the gap between very different affordances.

References

Bandura, A. (1997). *Social Learning Theory.* Englewood Cliffs, NJ: Prentice-Hall.

Cohen, E. G. (1994). Restructuring the classroom: Conditions for productive small groups. *Review of Educational Research, 64*(1), 1-35.

Chi, M. T. H., deLeeuw, N., Chiu, M., & LaVancher, C. (1994). Eliciting self-explanations improves understanding. *Cognitive Science, 18,* 439-477.

Dansereau, D. F., Collins, K. W., McDonald, B. A., Holley, C. D., Garland, J. C., Diekhoff, G., et al. (1979). Development and evaluation of a learning strategy program. *Journal of Educational Psychology, 71,* 64-73.

Dillenbourg, P. (2002). Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In P. A. Kirschner (Ed.), *Three worlds of CSCL. Can we support CSCL?* (pp. 61-91). Heerlen: Open Universiteit Nederland.

Dillenbourg, P., & Jermann, P. (in press). Designing integrative scripts. In F. Fischer, H. Mandl, J. Haake & I. Kollar (Eds.), *Scripting computer-supported collaborative learning: Cognitive, computational and educational perspectives.* New York: Springer.

Dillenbourg, P., & Tchounikine, P. (to appear). *Flexibility in macro-scripts for CSCL*. Submitted to International Conference of the Learning Sciences 2006.

Doise, W., & Mugny, G. (1984). *The social development of the intellect.* Oxford: Pergamon Press.

Hämälainen, R. & Arvaja, M. (submitted). Scripted collaboration and group-based variations in a higher education CSCL context.

Hämälainen, R. (submitted). Designing and evaluating collaboration in a virtual game environment for vocational learning.

Harel, D. & Naamad, A. (1987). The STATEMATE Semantics of Statecharts. *ACM Transactions on Software Engineering and Methodology, 5*(4), 293-333.

Harrer, A., Malzahn, N., Jermann, P. & Dillenbourg, P. (2005). *Bridging the gap - Different levels of granularity for CSCL scripts*. Presentation at the workshop on „ Computer-supported scripting of interaction in collaborative learning environments", CSCL Conference 2005, Taiwan.

Herring, S. C. (1999). Interactional Coherence in CMC. *Journal of Computer-Mediated Communication, 4*(4).

Johnson, D. W., & Johnson, R. T. (1998). Cooperative learning and social interdependence theory. In R. S. Tindale & L. Heath (Eds.), *Theory and research on small groups.* (pp. 9-35). New York, NY, US: Plenum Press.

King, A. (1989). Effects of Self-Questioning Training on College Students' Comprehension of Lectures. *Contemporary Educational Psychology, 14*, 366-381.

King, A. (1994). Guiding knowledge construction in the classroom: Effects of teaching children how to question and how to explain. *American Educational Research Journal, 30*, 338-368.

King, A. (in press). *Scripting collaborative learning processes. A cognitive perspective.* In F. Fischer, H. Mandl, J. Haake & I. Kollar (Eds.), *Scripting computer-supported collaborative learning: Cognitive, computational and educational perspectives.* New York: Springer.

Kollar, I., Fischer, F. & Hesse, F. (in press). Computer-supported cooperation scripts – A conceptual analysis.

Kuhn, D. (1991). *The skills of argument.* Cambridge: Cambridge University Press.

MOSIL (2004). *Framework for integrated learning*. Unpublished manuscript. Retrieved from http:// www.iwm-kmrc.de/cossicle/resources/D23-05-01-F.pdf

O'Donnell, A. M. (1999). Structuring dyadic interaction through scripted cooperation. In A. M. O'Donnell & A. King (Eds.), *Cognitive perspectives on peer learning.* (pp. 179-196). Mahwah, NJ, US: Lawrence Erlbaum Associates.

O'Donnell, A. M., & Dansereau, D. F. (1992). Scripted cooperation in student dyads: A method for analyzing and enhancing academic learning and performance. In R. Hertz-Lazarowitz & N. Miller (Eds.), *Interaction in cooperative groups: The theoretical anatomy of group learning.* (pp. 120-141). New York, NY, US: Cambridge University Press.

O'Donnell, A. M., Dansereau, D. F., Hall, R. H., & Rocklin, T. R. (1987). Cognitive, social/affective, and metacognitive outcomes of scripted cooperative learning. *Journal of Educational Psychology, 79*(4), 431-437.

Piaget, J. (1985). *The equilibration of cognitive structures: the central problem of intellectual development* (T. Brown & K. J. Thampy, Trans). Chicago: University of Chicago Press.

Schank, R. C., & Abelson, R. (1977). *Scripts, Plans, Goals, and Understanding.* Hillsdale, NJ: Earlbaum Assoc.

Slavin, R. E. (1983). When does cooperative learning increase student achievement? *Psychological Bulletin, 94*(3), 429-445.

Vygotsky, L. S. (1978). Mind in Society. *The Development of Higher Psychological Processes*. Cambridge, Massachusetts: Harvard University Press.

Webb, N. (1989). Peer interaction and learning in small groups. *International Journal of Educational Research, 13*, 21-39.

Webb, N. M., & Palincsar, A. S. (1996). Group processes in the classroom. In D. C. Berliner & R. C. Calfee (Eds.), *Handbook of educational psychology* (pp. 841-873). New York: Macmillan.

Weinberger, A., Ertl, B., Fischer, F., & Mandl, H. (2005). Epistemic and social scripts in computer-supported collaborative learning. *Instructional Science, 33*(1), 1-30.

Appendix: Guidelines for describing collaboration scripts

General rules:
Try to use keywords such as "each", "all", "at least" as much as possible in order to allow for scaling up/down in numbers.
Try to point out interdependencies among components, such as "one case description per group", which is more general and works with any number of groups.
Don’t explain why things are done this way or another way and how things relate to each other. The script should be self-explaining.
First, go through the components of your script. Then begin with specifying the mechanisms.

Resources:
Specify the kind and amount of resources that need to be prepared in advance. Generally, resources are distributed to the participants, i.e. they exist before interaction starts. Any resources created during the script do not have to be specified in this section. Make sure to specify whether a set of resources is composed of equal or unequal items. One way to do this is to refer to “copies” of something in the case of equal resources. For example, if all participants get the same one case description, say that “x copies of the case description are distributed among the participants” instead of “x case descriptions” which could be interpreted as meaning quite different case descriptions.

Participants:
Specify the amount of participants needed. Aim for the lowest possible number of participants. If there's a clear maximum value, also specify the range. If more than the minimum number of participants is desirable, also specify the recommended number of participants. For example: At least 2 participants (3 recommended). If the number of participants is dependent on the number of resources, roles, etc., say so. For example: “At least 2 participants (3 recommended) for each xxx”.

Groups:
Give all groups that are made use of in your script an identifying name, such as “case group”, “boy group”, “expert group”. If you refer to the whole class, use “class group” instead, so it resembles the other groups. Generally, groups are formed when the script begins. Therefore, they are referred to in detail in the sequencing part "group formation".

Roles:

Specify the names of roles required besides "participant". The default role of each student is "participant", because the role “student” and “teacher” need to be reserved for peer tutoring scenarios. Important: Only make use of roles when the reference to participants does not suffice. For example, if you have a pair of students who both have to summarize a paper, you refer to them as "all participants" or "the members of each group" with the activity of summarizing. However, if one of them is supposed to summarize while the other is supposed to listen, you need to refer to the one as "the summarizer" and to the other as "the listener". Thus, roles need to be used when members within a group engage in different concurrent activities. Roles are also used in order to refer to the way particular participants are different from other participants. For instance, when participants become experts in different fields of study, it makes sense to differentiate between them, e.g. by referring to “all experts of xxx” or “one expert from each field of study”.

Activities:

Keep this point open until you are all done with the sequencing. Then look at the activities you have used and try to describe each one of them in terms of acitivies that are relevant to learning. Thus, an activity like "writes a summary" becomes "summarizing". You can also become more specific about the activities you want the participants to engage in. For example, it is generally a good idea to keep the sequencing description short, so you write "participants discuss xy". In the activities section, you can then specify what kind of discussion you intend them to have and what kind of activities you expect them to engage in, e.g. "comparing, critiquing, formulating arguments". In some cases, the sequencing description already gives sufficient detail on activities in the script, so you don’t need to specify them in any further detail.

Group Formation:

Unless the participant characteristics make group membership obvious (e.g. a “boy group” is composed of all boys), describe how group are to be formed, giving details on group size (min/max/desired), amount of groups (min/max/desired) and their group composition (such as males & females, nations, expert/novices, etc.). Typically, groups are formed with a specific goal on group size or amount, but not size and amount. It helps to decide whether size or amount is more important when you think of how you would handle a number of students that

is larger than the one you need. For example, having 8 participants that you would like to group to groups of three, you have to decide whether the 7th and 8th will make a third group (in this case, you aim for a specific group size) or will be spread over two groups (in this case, you aim for a specific amount of groups). If group formation is dependent on other variables such as resources (e.g. "case descriptions"), then try to formulate it like this: "For each xxx, form one group of ...".

Component Distribution:

Consider each of the available components (in particular resources and roles) and describe which components are distributed over participants or groups of the script and how they are distributed. As in group formation, consider what would happen if you have more components than participants/groups (like 5 roles for 4 participants in each group) or less than needed (like 3 roles for 4 participants). Would that be allowed and how would you handle this? You could an expression like "evenly distributed" or "balanced" which means that a surplus of components is spread over all participants/groups so that groups have almost the same amount. Note that component distribution only states how components are initially distributed. Whether or not they are redistributed later on is stated in the sequencing section.

Sequencing:

In sequencing, you try to convey what is happening in a short hand form that gives barely enough details to understand how the script is to be conducted. First of all, consider the kinds of loops or repetition in your script. Do the participants traverse through a series of elements in a set (e.g. chapters in a book, levels in a game)? Do the participants rotate roles or resources? Do you find a repetition with changes that resembles fading (in/out)? Once you have discovered these loops, try to distinguish between different social planes, i.e. classroom, small group (if you have more than one kind of small group, you need to distinguish these, such as the case group, theme group and country group in the Universanté script) and solitary (individual) work. Always try to describe the interaction within this social plane, i.e. if you refer to members of a small group, describe it as "Within each small group, all participants ...". If you refer to the interaction between small groups describe it as "Within the class, all small groups ...". Thus, the "within" clause defines the scope of activities that you talk about. This way, you use a group perspective to describe the sequence of events that takes place. Now you assemble the sequence of blocks by stating the social plane and (if needed) the kind of repetition, such as “Within each country group, and for each theme group in turn, …”.

Then use activities that produce something, such as “write a summary”, “propose a solution”, “fill out a questionnaire”, “give an argument”, etc. This way, you can refer to a product when you describe the next steps in the activity sequence, such as “reads the summary”, “critiques the solution”, “corrects the questionnaire”, “reponds to the argument”, etc. Using such products allows you to describe some kind of “data flow” between the activities. Be careful to stick to the names and terms introduced. The general structure of the sequence begins with specifying the social plane that you are referring to (e.g. “Within the theme group”) and then specifying the activity on this social plane. If there is a sequence of activites, form a paragraph like:

Within each country group, …

… the members of the theme group present their fact sheet

… everbody else provides comments on the fact sheets

… the members of the theme group modify their fact sheet according to the comments

If group formation or component distribution takes place in a later phase of the script rather than at the very beginning, you can separate the sequencing into phases. This is also helpful if you need to refer to a specific section of a script that gets repeated. Also, use meaningful names for phases rather than numbers.