Sharing Educational Scenario Designs in Practitioner Communities

Astrid Wichmann, Jan Engler, H. Ulrich Hoppe
Collide Group, Department of Computer Science and Applied Cognitive Science,
University of Duisburg-Essen, Forsthausweg 2, 47057 Duisburg, Germany
{wichmann, engler, hoppe}@collide.info

Abstract: The possibility of sharing results and ideas is an important benefit of networked communities. In educational design, practitioners (e.g. teachers) specify educational scenarios that can be (re-)used, exchanged and modified at a later stage. In addition to supporting educational scenario design as such, our graphical editor SCY-SE offers functions to retrieve scenarios created by others based on a similarity measure. To examine the validity of this similarity measure in terms of correspondence with pre-defined cases, 25 participants were asked to produce graphical scenarios from given textual descriptions. First results strongly indicate that the calculated similarity was much higher between corresponding scenarios (related to the same text, but by different modellers) than in non-corresponding cases. Also, scenarios created by the same person show high dissimilarity for different cases. These results suggest that similarity-based matching is an effective and acceptable method to support the exchange among educational designers.

Introduction

Networked communities of practitioners have a great potential for accumulating ideas and artefacts that can be (re-)used, exchanged and modified at a later stage. This view is consistent with Wenger's notion of communities of practice (Wenger, 1998). The ability to share and exchange ideas and/or artefacts needs a certain degree of awareness on the part of the participants, which includes awareness of other active members in the community and their interests (social awareness) as well as awareness of the objects or artefacts produced in the community (content awareness). In small local communities, social awareness and social interaction often guide the identification of objects of common interest. Yet, in realistically sized networked communities, this will only work to a very limited extent. Alternatively, objects of common interest have to be identified on the content level. Technically, this identification can be achieved through providing a similarity measure for pairs of objects. Here, we assume that a community repository will accumulate objects produced by community members, and that these objects can be seen as indicators of the current interests of their authors (at creation time). So, for each new object that is uploaded to the community repository we can automatically search for similar objects that already exist. These existing objects may be brought to the awareness of the new author, which may result in revising the content in regard of existing material or even in contacting the creators of the existing objects to coordinate further elaboration and other actions.

A similar approach within a community of science learners has been suggested by Hoppe et al. (2005). Here, the objects of interest were so-called "emerging learning objects" created by members of different local teams in an international learning community that shared different scientific challenges without personally knowing each other between the different local teams. Objects of common interest between two partners were also identified by similarity, exploiting the internal structure of the objects. The similarity measure had to work in the absence of user-defined metadata since it was assumed that a sufficiently complete and consistent indexing of emerging objects by the creators (high schools students) could not be expected. A comparable mechanism to identify congruence of interests in a community of scientists has been suggested by Francq & Delchambre (2005) based on a publication database with a similarity definition using automatic indexing techniques. Generally, similarity measures can be used as a source of continuous information on themes of personal interest. Similar functions are served by *recommender systems* (cf. Konstan & Riedl, 2002), which usually rely on information about usage processes, namely by computing object-object associations through co-occurrences in user traces.

In our study, we target a community of educational designers who produce semi-formal designs of structured learning scenarios using a graphical editing tool. The potential benefit of automatically identifying objects of common interests (i.e. specific educational designs) as a content awareness function is evident: It will be possible to relate new suggestions to existing ones early at design time. Thus, unnecessary redundancies can be avoided and synergy between similar approaches can be exploited. The question, however, is if the underlying similarity measure does really capture those features that would be considered as relevant by human judgment. Suggestions based on irrelevant or artificial similarities would soon lead to rejection of the support mechanism. To validate the similarity mechanism in this sense, we provided users with a number of different textually defined learning scenarios that were to be represented (or modelled) graphically using our editing tool. To be "valid" and acceptable the mechanism should adequately reproduce the original differences irrespective of personal preferences and idiosyncrasies. This makes another assumption, which is that the modelling task with the given tool itself is well-defined in that a substantial amount of relevant information can be transferred from the textual description to the graphical model. A negative outcome would still not clarify if the problem was with the similarity calculation or with the coding task (or both). A positive outcome, however, would confirm both the task-tool fit and the adequacy of the similarity measure. In the sequel, we will first describe the scientific context of this study in a European research project, then elaborate on the technical ingredients (editor and similarity measure) and finally report our empirical findings.

Educational design of inquiry-based learning scenarios

The requirements for the work presented in this paper stem from a new European research project called SCY[1]. In the SCY project (de Jong, Joolingen van, Weinberger, 2009), students engage in inquiry learning activities supported by computer tools such as simulations and modelling software. In SCY-Lab (the SCY learning environment), students work on missions and meet challenges collaboratively and individually supported by learning material, tools and scaffolds. To meet the challenges of inquiry learning, the configuration of SCY-Lab is adaptive to learners' capabilities and progress. In order to support the cyclic nature on inquiry learning, SCY-Lab provides tools and scaffolds, which provide just-in time support (Anjewierden, Chen, Wichmann, & Borkulo van, 2009). Another characteristic of inquiry learning is learners' rich interaction with learning material. In SCY missions, students not only receive learning material (e.g. in form of text), they also produce material (e.g. in form of data by running experiments). The construction of artefacts that emerge from the learning process has been articulated as one of the central ideas in SCY. These emerging learning objects (ELOs) are re-usable and sharable products of learning activities, which are created by learners. As part of SCY's pedagogical approach, several so-called *learning activity spaces* (LASs) have been identified. These are characterised by specific combinations of tools, activities and types of input/output objects. *Learning scenarios* are defined on an upper level by a set of LASs with partial sequencing.

One goal of SCY is to support a community of practitioners (teachers and instructional designers) in creating and sharing educational scenarios. Existing languages and tools in the area of learning design such as LAMS (Dalziel, 2003) or IMS/LD[2] appeared to be not sufficient for our purposes for different reasons: LAMS is too coarse grained with respect to capturing object-tool-activity relationships and results in quite fixed sequential models with too little flexibility. IMS/LD is more expressive with regard to tools, activities and roles, yet it lacks a conceptual model and graphical representation that would be usable by non-technical experts (cf. Miao, Harrer, Hoeksema, & Hoppe, 2007). Both approaches lack possibilities to adequately represent input and output relations with ELOs.

As argued above, effective community support has to include the sharing and re-use of previously developed scenarios. For teachers who develop new scenarios for SCY missions it is important to be able to find scenarios with specific characteristics. For instance, several strategies exist to introduce scientific texts to learners. One teacher might favour a concept mapping activity to identify and integrate new concepts while another teacher would request the students to summarize the text using a note-taking tool. Different approaches to facilitate sharing and re-use have been suggested. One approach is to select and filter scenarios based on certain categories such as age and subject. However, single categories like these may not be sufficient to capture what is relevant for the teacher (cf. Ronen, Kohen-Vacs, & Raz-Fogel, 2006). Another method is to let users (here: teachers) freely tag their design objects. Tagging, however, bears several problems. First, it is an additional effort, which is often avoided. Second, the degree of freedom and subjectivity in free tagging may lead to inconsistencies of various types, which will lead to unreliable results when used for retrieval. Therefore, we propose a similarity measure that reflects the two-level structure of our learning design and uses several content categories (such as activities, tools and object types). For instance, a teacher may want to focus on argumentative activities and might thus be interested specifically in other scenarios involving argumentation tools and ELO types compatible with argumentation.

The SCY- SE Scenario Editor and graphical language

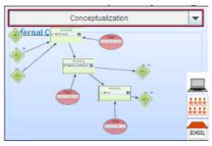
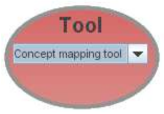
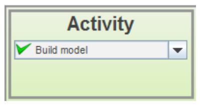
The SCY Scenario Editor (SCY-SE) has been designed and implemented following the definition of pedagogical requirements and the ensuing specification of a graphical representation language (de Jong et al., 2009). SCY-SE has been implemented as a plug-in to the multi-functional graphical modelling environment FreeStyler (Hoppe & Gaßner, 2002). FreeStyler provides a uniform platform for various visual languages with free-hand annotation

[1] SCY – "Science created by You" is an EU project of the 7th Framework Programme. For more information, see http://www.scy-net.eu (last visited in October 2009).

[2] For the full specification and more information in IMS-LD, see http://www.imsglobal.org/learningdesign (last visited in October 2009).

facilities and shared workspace facilities. The graphical user interface of SCY-SE offers a two-level view reflecting the pedagogical design: the scenario view combines several LASs with input and output ELOs, whereas the lower-level LAS view specifies activities, tools, intermediate ELOs and other resources. SCY-SE provides a toolbox ("palette") from which objects can be dragged onto the workspace.

Table 1. Concepts in SCY-SE

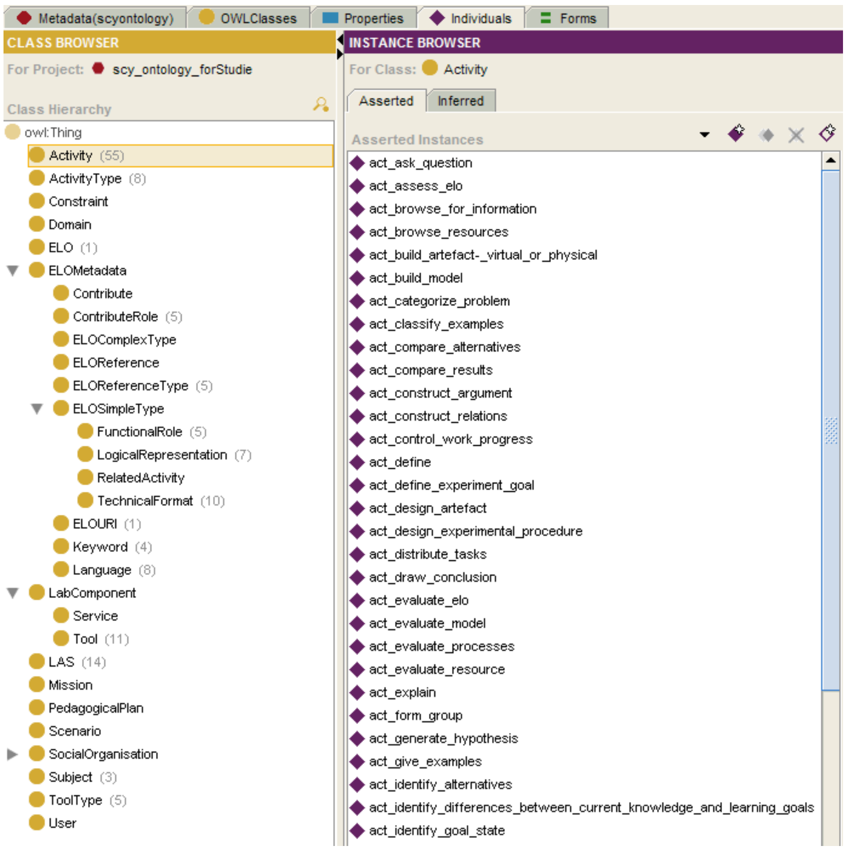
Concept		Specified by the SCY-SE user
ELO (Emerging Learning Object)		ELO characteristics (functional role, technical format, logical representation, belonging activity)
LAS (Learning Activity Space)		- LAS type (e.g., conceptualization, build, debate, experiment) - Location (at home, classroom, field) - Setting (peer to peer, whole class, alone) - Computer use (yes, no)
Tool		Tool type (e.g., simulation tool, modelling tool, note taking tool)
Activity		Activity type as suggested by the LAS type (e.g., build model, run experiment, define)
Scaffolds		Scaffold that is connected to a tool or LAS that represents scaffolding mechanisms for the specific entity.
Resources		Resource that is used for an activity (e.g. video, audio, text)

The table above (Table 1) shows the objects that are currently implemented in SCY-SE and part of the SCY language. In the study, SCY-SE was modified so that ELOs did not need to be specified with respect to format etc. and scaffolds were not available in the toolbox.

Background: Blackboard architecture and ontology

The SCY environment relies on loosely coupled components with intelligent agent support. Agents communicate with other system components as well among themselves using a so-called *blackboard architecture*. The basic idea of the blackboard architecture is to provide interaction between components (such as agents) not by addressing other components directly, but by communicating only via writing and reading messages on the blackboard. The advantage of this paradigm is that every component is running autonomously and little knowledge about other components is necessary. This approach leads to a more robust system and provides an architecture that can be easily and flexibly extended or modified. A well-known technical approach to implementing a blackboard architecture is TupleSpaces (Gelernter, 1985). Here, a dedicated server acts as an operational centre for exchanging information. The information is stored on this server in form of tuples, which are ordered sequences of typed data. There are several implementations of TupleSpaces like JavaSpaces (Sun Microsystems) or TSpaces (IBM). We are using an implementation called SQLSpaces (Weinbrenner, Giemza, & Hoppe, 2007). SQLSpaces offer some extra features such as asynchronous call-backs, programmable expiration time of tuples, blocking and non-blocking operations, a web-based visualization, versioning and user management. Furthermore, SQLSpaces come client interfaces for several programming languages, including Java, C#, Ruby, Prolog and PHP. In the case of SCY-SE, the agent that calculates the similarity between scenarios is written in Prolog, a logic programming language.

SCY-SE also makes use of a pedagogical ontology, which captures the basic terms describing activities, tools, object types and their interrelationships. The checking mechanism, which is described later, utilizes these relations to check whether the relations between the concepts of the user are valid or not. Using the ontology for the concepts and relations results in a system, which is easily extend- and adaptable. New constraints or types of concepts can be inserted into the ontology using an editor like Protégé[3]. Using an ontology allows us not only to access concepts related to SCY, but also to specify relations between these concepts, which is useful for determining similarity between concepts.



For Project: scy_ontology_forStudie

Class Hierarchy

- owl:Thing
  - Activity (55)
  - ActivityType (8)
  - Constraint
  - Domain
  - ELO (1)
  - ELOMetadata
    - Contribute
    - ContributeRole (5)
    - ELOComplexType
    - ELOReference
    - ELOReferenceType (5)
    - ELOSimpleType
      - FunctionalRole (5)
      - LogicalRepresentation (7)
      - RelatedActivity
      - TechnicalFormat (10)
    - ELOURI (1)
    - Keyword (4)
    - Language (8)
  - LabComponent
    - Service
    - Tool (11)
  - LAS (14)
  - Mission
  - PedagogicalPlan
  - Scenario
  - SocialOrganisation
  - Subject (3)
  - ToolType (5)
  - User

For Class: Activity

Asserted Instances

- act_ask_question
- act_assess_elo
- act_browse_for_information
- act_browse_resources
- act_build_artefact-_virtual_or_physical
- act_build_model
- act_categorize_problem
- act_classify_examples
- act_compare_alternatives
- act_compare_results
- act_construct_argument
- act_construct_relations
- act_control_work_progress
- act_define
- act_define_experiment_goal
- act_design_artefact
- act_design_experimental_procedure
- act_distribute_tasks
- act_draw_conclusion
- act_evaluate_elo
- act_evaluate_model
- act_evaluate_processes
- act_evaluate_resource
- act_explain
- act_form_group
- act_generate_hypothesis
- act_give_examples
- act_identify_alternatives
- act_identify_differences_between_current_knowledge_and_learning_goals
- act_identify_goal_state

Figure 1: Protégé screenshot of the ontology and its concepts.

Similarity algorithm

The overall process of calculating the similarity is shown in Figure 2. In the first step, SCY-SE constructs lists of the relevant entities of the scenarios to be compared. These relevant entities consist of the three components ELOs, tools and activities, which are grouped by the corresponding LASs of the two scenarios. These can again be divided into two categories: Activities and tools are atomic values, so that lists of them can be compared directly through the “weighted symmetric distance” of two sets. This distance is based on the cardinality of the symmetric difference, which is then divided by the cardinality of the union of the two sets A and B:

$$\Delta_{A,B} = \frac{\#((A \cup B) \setminus (A \cap B))}{\#(A \cup B)}$$

This distance calculation has been modified by component weights based on the ontological proximity between the single elements. The resulting similarity value ranges between 0 and 1.

ELOs, on the other hand, are not suited for a plain comparison as mentioned above. In fact, they are built up from four properties and therefore have an intrinsic complexity that requires a different approach for calculating the similarity. At first glance, it is possible to calculate the similarity of two ELOs A and B by comparing the four properties, so if two ELOs share three of their four properties, they would have a similarity of 0.75. Nevertheless, the comparison of A and C might result in an even higher similarity. So, each ELO of the first list needs to be compared with each of the second one. The outcome of this step is a matrix of similarity values. To condense this matrix to one single similarity value that expresses the similarity of these two lists, we decided to use the so-called “Hungarian method” for graph matching (Kuhn, 1955) to solve this assignment problem (to determine which ELO is identified with which other ELO).

[3] For a documentation and download location visit: http://protege.stanford.edu/doc/users.html (last visited in October 2009)

As shown in Figure 2, the calculation inside the Prolog agent is done in several steps. After the lists have been written to the SQLSpaces server, a "start comparing" tuple is written to inform the agent about a new query. Then the agent fetches the lists and calculates the weighted symmetric distance for each of the lists. At that point the matrix for the ELOs is already calculated by using the "Hungarian method". The next step uses the merged matrix of these three matrices. The Hungarian method is then used again to determine the best mapping between the LASs of the two scenarios with respect to the similarity encoded in the matrix. The method may take several iterations, but will eventually terminate and return the optimal solution. From the resulting mapping, a single value can be easily calculated by normalizing the selected matrix entries. The returned "similarity value" (Sim(A,B)) is transformed in such a way that Δ=0 corresponds to Sim=100% and Δ=1 corresponds to Sim=0%.

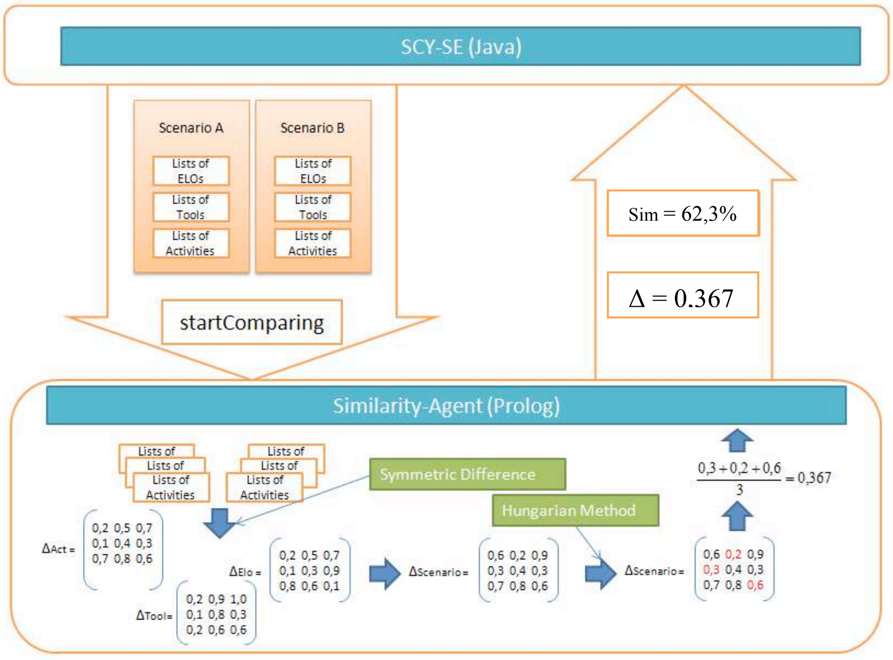


Figure 2: The overall process of the similarity measurement.

For the study, not all components of the similarity measure had an effect on the result because of the limited version of SCY-SE that participants were using. Since the limited version did not include options to specify ELOs with respect to technical format etc., all ELOs were similar. Therefore, the similarity measurement was based on other factors (e.g., number of LASs, number and type of tools and activities).

Checking

SCY-SE allows for the automatic checking against several types of constraints and constraint violations (such as incompatibilities of object types) in given designs, some of which are ontology based. The more obvious ones apply to the way nodes are connected. To distinguish between ELOs that are emerging from a LAS and ELOs that work as starting material for a LAS, SCY uses the notion of input and output ELOs. An output ELO needs some activity that produces it and an input ELO needs some activity that consumes it. Moreover, ELOs can have more or less precisely defined types. These types also have a "compatibility" relation to tools (that support activities) based on the tool's potential output data formats. This compatibility relation is stored in the ontology. Violations of any rule are displayed next to each incorrect node with short information about the problem and possibly a hint for resolving this violation.

Research goals and questions

The main goal of this study was to test the validity of the similarity algorithm. We prepared three textual descriptions of scenarios consisting of inquiry learning activities. Based on these textual scenarios, students created graphical scenarios using SCY-SE. In addition, expert solutions were created beforehand by researchers, to which the student scenarios were compared. To distinguish between student scenarios and expert solutions, scenarios created by students use a lower case letter (a, b, c) and expert solutions use an uppercase letter (A, B, C). The following questions were of interest:

1. a) Are SCY-SE scenarios created based on the textual scenario a, more similar to a corresponding expert solution A than to a non-corresponding expert solution B?
   Hypotheses: Sim(a, A) > Sim(a, B); Sim(a, A) > Sim(a, C); Sim(b,B) > Sim(b,A) etc.

b) Are the differences induced by the given textual scenarios still discernable in the graphical scenarios produced by the same subject?
Hypotheses: Sim(a, b) < Sim(a, A); Sim(a, c) < Sim(a, A); Sim(b, c) < Sim(b, B) etc.

Additionally, we were interested in the influence of checking and ensuing feedback:

2. a) Does the checking feature help to guide users during the process of creating a complete SCY-SE scenario?
   b) Does using the checking feature increase the similarity to the corresponding expert solution?

Design of the Study

Participants, procedure and instrument

Twenty-five students of media-technology or teaching were participating in the study, taking place at a university lab in Germany. Participants listened to a half hour introduction to the SCY-SE approach and to the language including a demo of how to use SCY-SE. After the introduction, everyone received a small manual and the task description. Every student worked individually on one computer. The task was to translate three textual scenarios into graphical scenarios using SCY-SE. Participants had to decide which LASs, activities, tools and ELOs need to be selected to represent the textual scenarios graphically. When a student was finished with the last scenario, she was requested to save the *SCY-SE scenario c* under a different name, to use the checking button, and to revise *SCY-SE scenario c-checked* accordingly. The participation time was two hours. A questionnaire including demographic questions and single-item questions with respect to perceived difficulty was administered right after the study.

Material

Every textual scenario consisted of four paragraphs. Each paragraph represented one LAS. All textual scenarios described activities following an inquiry approach within the subject matter of photosynthesis. The textual scenarios differed from each other with respect to activities, tools and ELOs. The following paragraph is an example of how activities were described in the textual scenario:

> "Students work in groups to do their experiments using SCY Simulator. They define their experiment goals in SCY Simulator. Afterwards the students start doing their experiment runs. Then, the students compare the results with their hypothesis stated earlier. They make notes, which they upload together with their experiment data."

The translation of this paragraph into a SCY-SE scenario may look like this:

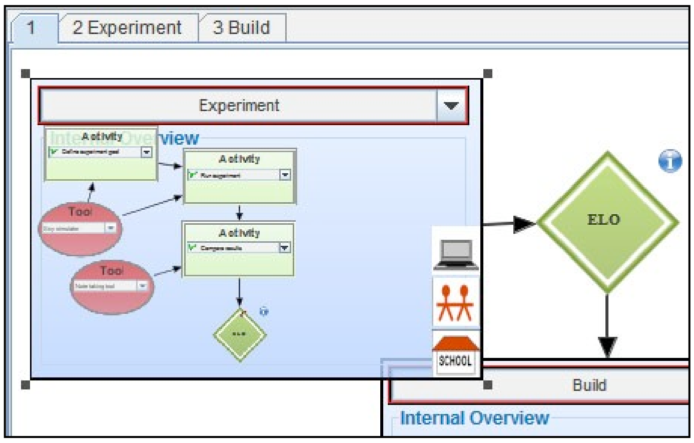


Figure 3. SCY-SE Scenario view

Source of Analysis

Hundred SCY-SE scenario files (four – three unchecked and one checked- from each participant) were collected and analyzed. Similarities were calculated. For answering research question 1a, SCY-SE scenarios were compared with corresponding and non-corresponding SCY-SE scenario expert solutions. Concerning question 1b, SCY-SE scenarios were compared with SCY-SE scenarios by the same student. Answering research question

2a required logging the number of errors detected by the checking mechanism and question 2b required to compare SCY-SE scenario c with SCY-SE scenario c-checked.

Results

Due to low sample size, descriptive statistics were favoured and non-parametric methods were used. From the questionnaire, we found that most participants (72%) reported that textual scenarios were of similar difficulty level. Some participants (24%) identified one of the three scenarios as most difficult. Most participants (76%) reported that it was most difficult to select the tools (besides LASs and Activities).

First, we compared every student's scenario a, b, c with the corresponding expert solution and with the non-corresponding expert solutions. As expected, if student scenarios were compared with the corresponding expert solution, the percentage with respect to similarity was higher (M=81, SD=07) than if student scenarios were compared with the non-corresponding expert solutions (M=55, SD=04). Wilcoxon's Signed-Rank test showed that this difference (Table 2 and Table 3) was statistically significant (z=-4.37, p=.000).

Table 2. Descriptives for similarity between student scenario and corresponding expert solution in %

student scenario | expert solution	N	Min	Max	Mean	Std. Dev.
a | A	25	25	85	74	12
b | B	25	62	99	83	09
c | C	25	62	96	86	08

Table 3. Descriptives for similarity between student scenario and non-corresponding expert solution in %

student scenario | expert solution	N	Min	Max	Mean	Std. Dev.
a | C	25	24	68	54	10
a | B	25	19	69	55	10
b | C	25	40	68	54	06
b | A	25	44	69	56	05
c | A	25	47	55	58	05
c | B	25	41	67	48	04

Second, we compared student scenarios with student scenarios (Table 4) done by the same student (Sim(a,b); Sim(a,c), Sim(b,c)). As expected, similarity was low (M=53, SD=08) and comparative to similarity between expert solutions A, B, C (M=58, SD=49).

Table 4. Descriptives for similarity between (non-corresponding) scenarios for every student

student scenario | student scenario	N	Min	Max	Mean	Std. Dev.
a | b	25	14	68	55	10
a | c	25	21	75	53	11
b | c	25	39	65	51	06
Valid N (listwise)	25				

Checking

Results show, that the checking feature was necessary for most participants to create complete and correct SCY-SE scenarios. On average, participants made 4.7 (SD=3.3) errors that were detected by the checking mechanism. In order to find out whether the checking feature would increase similarity between student scenario and expert solution, we compared the last scenario (*SCY-SE scenario c*) that was developed with the corresponding expert solution (*SCY-SE Scenario expert solution C*). We found no significant difference with respect to similarity comparing unchecked and checked scenarios. On average, *SCY-SE scenario c* was 86% similar to the *SCY-SE scenario expert solution C*. After using the checking mechanism, average similarity was still 86%. In order to find out why the checking feature did not yield higher similarity, an in depth analysis with respect to the types of errors was done. The errors that were displayed while using the checking feature were mostly related to incorrect usage of links between objects or missing links. For example, many participants missed making links between tools and activities. Since the similarity algorithm did not take into account the number of links, the checking had no positive effect on the similarity measure.

Conclusion and Implications

The results demonstrate that the similarity measure captures well the plausible similarity between graphical scenario designs originating from the same textual source across subjects (as "modellers"). We could also show that dissimilarities of scenarios developed by the same person but based on dissimilar textual scenarios were detected (and therefore discriminated). It was somewhat surprising that scenarios that were based on non-corresponding textual descriptions still resulted in a similarity of over 50%. One reason was that all three textual scenarios (which were the basis for the graphical representations in SCY-SE) had approximately the same length and the same number of LASs. Together with certain tool similarities suggested by the ontology this resulted in a quite high baseline. Yet, as the similarity of scenarios based on corresponding textual scenarios was over 80% (which results in a statistically significant difference), we were able to reliably discriminate similar scenarios from dissimilar scenarios. Another finding was that guiding a practitioner during the creation of a scenario using SCY-SE is beneficial. Using a checking mechanism helps to reduce the number of errors and to use the SCY graphical language consistently.

References

Anjewierden, A., Chen, W., Wichmann, A., & Borkulo van, S. P. (2009). Process and product analysis to initiate peer-to-peer collaboration on system dynamics modelling. In J. Bourdeau and R. Mizoguchi (Eds.), *Intelligent and Innovative Support for Collaborative Learning Activities (WIISCOLA).* Workshop conducted at the Computer-Supported Collaborative Learning Conference CSCL 2009, Rhodes, Greece.

Dalziel, J. R. (2003, August). Implementing learning design: the learning activity management system (LAMS). *Paper presented at the 20th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education*. Adelaide, Australia.

De Jong, T., Joolingen van, W. R., Weinberger, A., & the SCY team (2009, June). Learning by design. An example from the SCY-project. *Paper presented at the CSCL 2009*, Rhodes, Greece.

Francq, P., & Delchambre, A. (2005). Using documents assessments to build communities of interest. *Proceedings of the IEEE/IPSJ International Symposium on Applications and the Internet* (SAINT 2005) (pp. 237-333). Los Alamitos, USA: IEEE Computer Society.

Gelernter, D. (1985). Generative communication in Linda. *ACM Transactions on Programming Languages and Systems, 7*(1), 80-112.

Hoppe, H. U., & Gaßner, K. (2002). Integrating collaborative concept mapping tools with group memory and retrieval functions. In G. Stahl (Ed.), *International Conference on Computer Supported Collaborative Learning: Foundations for a CSCL Community* (pp. 716–725). Mahwah, NJ: LEA.

Hoppe, H. U., Pinkwart, N., Oelinger, M., Zeini, S., Verdejo, F., Barros, B., Mayorga, J. I. (2005). Building bridges within learning communities through ontologies and "thematic objects". In *Proceedings of* Computer-Supported Collaborative Learning Conference CSCL 2005 (pp. 211-220). Mahwah, NJ: LEA.

Konstan, J.A. & Riedl, J. (2002). Collaborative Filtering: Supporting social navigation in large, crowded infospaces. In Höök, K., Munro, A., & Benyon, D. (Eds.). *Designing Information Spaces: The Social Navigation Approach* (pp. 43-82). Berlin, Germany: Springer.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2, 83-97.

Miao, Y., Harrer, A., Hoeksema, K., & Hoppe, H. U. (2007). Modelling CSCL scripts – a reflection on learning design approaches. In F. Fischer et al. (Eds.), *Scripting Computer-Supported Collaborative Learning* (pp. 116-134). Berlin, Germany: Springer.

Ronen, M., Kohen-Vacs, D., & Raz-Fogel, N. (2006). Structuring, sharing and reusing asynchronous collaborative pedagogy. In Y. Kafai, W. Sandoval, A. Noel Enyedy & F. Herrera (Eds.), *International Conference of the Learning Sciences* (Vol. 1, pp. 955-605). Bloomington, Indiana: LEA.

Weinbrenner, S., Giemza, A. & Hoppe, H. U (2007). Engineering heterogeneous distributed learning environments using TupleSpaces as an architectural platform. *Proceedings of the 7th IEEE International Conference on Advanced Learning Technologies* (pp. 434-436). Los Alamitos, USA: IEEE Computer Society.

Wenger, E. (1998). *Communities of Practice: Learning, Meaning, and Identity*. Cambridge University Press.