In this article we present a structured set of Design Patterns (DPs) that deals with tracking students while they solve problems in specific domains such as programming or mathematics. Our collection of 17 DPs yields a three step approach: First step: to collect and analyze information on each student for each exercise; Second step: to build a higher level view of one student’s activity on a set of exercises; Third step: to build an overview of the whole class activity. To evaluate our DP set we investigate whether our DPs account for experiences from the literature as a first step toward a pattern language for students’ assessment.

Introduction

The usage of learning systems is a large research field and there is a lot of scattered work on this issue. In our work we assume that a Design Pattern approach is a way to collect and to share experiences, to have a meta reflection and to capitalize on context specific research results.

The first set of Design Patterns was suggested by Alexander (Alexander et al., 1977) in the architecture domain. Alexander’s definition of a Pattern is still a reference:
Each pattern describes a problem which occurs over and over again in our environment and then describes the core of the solution of that problem, in such a way that you can use this solution a million times over, without ever doing the same way twice (1977, p. x).

In Alexander’s perspective a network of related patterns creates a *language* to be used by every one involved in a design process whether one designs a house for oneself, or works with others to design offices or public spaces.

In the last three decades, the Pattern approach has found its way into many disciplines such as architecture, software engineering, human computer interaction, website design, and e-learning (Alexander et al., 1977; Buschmann, Meunier, Rohnert, Sommerlad, & Stal 1996; Schmidt, Stal, Rohnert, & Buschmann, 2000; Avgeriou, Papasalouros, Retalis, & Skordalakis, 2003; van Duyne, Landay, & Hong, 2003; Chung, Hong, Lin, Prabaker, Landay & Liu 2004; van Welie, 2004; Deng, Kemp, & Todd, 2005). In the e-learning research field, collections of pedagogical patterns are now available.

The Pedagogical Pattern Project (PPP) (PPP, n.d.) provides three collections of patterns for educational scenarios. They aim to capture experts’ practice, in that case, experienced teachers. These high-level patterns use an Alexandrian format and are narrative expressed in a you-form to address academic teachers’ or industry instructors’ problems. One collection of fourteen Patterns is presented as a step towards a pattern language for computer science course development: *teaching from different perspectives, active learning, feedback patterns, patterns for experiential learning and patterns for gaining different perspectives*. A second set of forty-eight patterns forms a pattern language to teach seminars effectively. A last set of five patterns about running a course is suggested. PPP patterns do not address explicitly the use of technology.

The E-LEN project (Avgeriou et al., 2003; Avgeriou P., Vogiatzis D., Tzanavari A., Retalis S., 2004; Goodyear et al., 2004; E-LEN, n.d.) provides a booklet with guidelines to develop Design Patterns for e-learning. It also provides a repository of forty patterns classified into four different special interest groups (SIG): Learning resources and learning management systems (LMS), lifelong learning, collaborative learning and adaptive learning. They aim to construct a knowledge base for educational designers and they promote a Design Pattern approach to collect and disseminate re-usable design knowledge, methods of sharing design experience and supporting the work of multidisciplinary teams. Their DP users are e-learning designers. Their DPs are high-level patterns. For instance, the DP *student tracking* suggests the functionalities to be implemented in the system but it is not clear how to implement these functionalities. It is the designer’s responsibility to generate a solution adapted to his/her context and eventually, to create a more spe-
cialized Design Pattern if he/she detects invariant in his/her solution. “A pattern suggests rather than prescribes a solution” (E-LEN, n.d.).

In, “The Design Patterns for Recording and Analysing Usage of Learning Systems” (Choquet, 2004) (DPULS), part of the Kaleidoscope European Network of Excellence is to come up with a set of Design Patterns (DPs) that allows the tracking of actors’ activity. As in the E-LEN project, the goal is to support good design decision-making but also to share experiences and to build a common language between a European set of research teams. The DPULS DPs focus on data collection and analysis to investigate more deeply the student tracking problem and to complement PPP and E-LEN projects.

In this article we present the subset of the DPULS DPs that deals with tracking students’ know-how and strategies while they solve problems in specific domains such as programming or mathematics. This subset is called Learners’ Assessment DPs (LA DPs for short). We first specify our research goals and methodology and compare them to related works. After introducing three different successful practices that grounded our DP design, we present the structured set of DPs on Learners’ Assessment. We end with a discussion comparing our approach to others and draw some perspectives.

**Research Goals and Methodology**

In communities that have adopted the pattern approach Design Patterns, there is a large agreement on the definition of a pattern as a solution to a recurrent problem in a context (Goodyear et al., 2004). However many issues are debated such as for example:

- **Who are DP users?** DPs are written to support experts, researchers, members of a community of practice, multidisciplinary teams, end-users of the designed product or even a machine to implement the solution.

- **Why create a DP set?** The purposes can vary, for example, to create a common language in a multidisciplinary or multicultural design or research team, to capture/disseminate expertise, to capitalize on previous experiences, to ensure quality, or to teach.

- **What is the format to express DPs?** Many efforts are devoted to define DP form (narrative or UML or XML schemes), structure, naming and referencing.

- **How to create a DP?** The bottom-up approach is the most common approach but some authors suggest a top-down approach or a “bottom-up approach informed by theory” (E-LEN, n.d.).

- **How to create a DP language?** Structure, granularity, DP combination, and specificity are key points to be discussed.

- **How to validate or evaluate a DP language?** Several ways of validation are presented in the literature: the “rule of three” (three examples of
successful experiences using the pattern), peer review, feedback from practitioners and usability testing.

Design Patterns are usually drafted, shared, criticized and refined through an extended process of collaboration. To this end, a five-step methodology was adopted in DPULS. First, we studied a number of practices in our context of interest (DPULS, 2005, 2-3). Second, from these practices, we selected a set of common and recurrent problems and we set up a know-how list (DPULS, 2005, 4). Third, we worked on descriptions of problem statements and solutions in a way general enough to cover each individual experience, and it was a major task. From this step we negotiated a common format to express the set of DPs (DPULS, 2005, 5) and to design a browser to support the navigation in this set of DPs (DPULS, 2005, 7). Fourth, at the same time we worked to reach an agreement on whether and how the different problems were linked. Many iterations were necessary between the third and the fourth step. Fifth, we came up with a set of DPs stable enough to account for other similar practices (DPULS, 2005, 6) and we entered the patterns in the browser. Every step involved interactions between partners.

The purpose of the DP set presented here was triple:

• To express invariance in solutions experimented by DPULS teams to solve assessment problems,
• To account for others’ experiences on the same class of problems using our Pattern language,
• To support designers who deal with a learner’s assessment in individual learning systems.

Here is a scenario that illustrates the sort of problem practitioners might face when designing a system for understanding a student’s actual learning activity.

*Sarah is a teacher who organizes lab work for a group of students. She wants a report on the lab work session to check whether some know-how has been mastered and to get an insight on the kind of strategies the students have used. Her aim is to plan the next session and to adapt her teaching to her students' understanding. She needs an overview on the students' activity during the lab session.*

How should the system be designed to make this scenario possible? Our DPs yield a three step approach:

• First step: to collect and analyze information on each student for each exercise;
• Second step: to build a higher level view of one student's activity on a set of exercises;
• Third step: to build an overview of the whole class activity.

In e-learning systems, there are different assessment approaches. Our approach goes beyond most current assessment practices where assessment allocates grades and is made through quizzes, multiple choice questions or numerical answers (e.g., management of online questionnaire (Avgeriou et al., 2003). We also aim to assess students' productions when they are asked to perform tasks specially designed to make the taught knowledge personally meaningful. We deal with problems where students have to perform complex cognitive operations. This type of assessment requires more than a right/wrong assessment. It requires a multidimensional analysis based on accurate pedagogical, didactical or cognitive studies of the student’s activity. In our scenario, this point can be illustrated by:

Sarah is an experienced teacher. She is aware of students' personal interpretations and, from her experience or from cognitive research results, she derived a typology of students' answers to a class of problem. She wants to group her students using this typology.

The DPULS set of DPs captures experiences from a multidisciplinary and multicultural team of researchers. It was built with a bottom-up approach to capture the participants' expertise and to integrate European research works. In this article we present DPs in a narrative format for human readability. A machine readable version exists that is processed by a DP Browser (DPULS, 2005, 7). They are high level patterns validated by peer review and by at least the “rule of three”.

Background Experiences

What characterizes a set of DPs is not the originality of its content – usually most of the ideas are known since they are proven solutions to well-identified problems. Rather, the merit of a set of DPs is to bring together and to structure existing best practices. In our work, we first generalized from six very different e-learning assessment experiences in the AIDA team. Six projects deal with students' assessments: Combien? (Combien, 2000), Diane (Hakem, Sander, & Labat, 2005), Java Course (Duval, Merceron, Scholl, & Wargon, 2005), Logic-ITA (Merceron, & Yacef, 2004), Pépite (Pepite, 2000), Math Logs (Vandebrouck, Cazes, Gueudet, & Hersant, 2005). In this section, we present three of these experiences so that readers can give a concrete content to the DPs presented in the next section. We selected them to show a large range of assessment contexts. Combien? is a learning environment to train undergraduate students in combinatorics. Pépite is a diagnosis system that analyzes students' productions to diagnose their algebraic competence in secondary school algebra. Logic-ITA is a web based tutoring system for training second year university students in propositional Logic.

Combien? trains students to solve combinatorics exercises and to justify their solution. In combinatorics (counting the number of ways of arranging given objects in a prescribed way), the main part of the solving process does not come from a clever chaining of inferences or calculations, but from the elaboration of a suitable representation and from the transformation of one representation into an equivalent one. This work is based on research that emphasizes the importance of representations in problem solving. Combien? is grounded in the constructive method (Le Calvez, Giroire, Duma, Tisseau, & Urtasun, 2003), a problem solving method set up by a group of teachers, adapted to the usual students' conceptions in order to give them access to the mathematical theory of the domain.

Students are asked to build a generic element (called configuration) of the Solution Set by describing their construction as a set and a set of constraints. Then, they have to reason about this construction to find the numerical solution. Combien? offers one interface for each class of exercises. At each step of the construction the system automatically determines whether the students' ongoing solution leads to a right construction or not. In the latter case it gives hints to help students to understand their errors.

Each year, one hundred second-year university students use the system in lab sessions. Students can also work with Combien? at home for personal training. All the students' actions (data input and validation) and their time stamping are recorded so that the session can be played again by the system. Combien? detects students' errors, classifies them, and records their type and their characterization. All this information is recorded in XML files analyzed a posteriori. Combien? offers two types of analysis. For each student, Combien? presents a detailed account of her session. For each exercise, it reports the total solving duration, the number of errors and their type, the number of hesitations, the exercise achievement, and the number of course consultations. This analysis is available for both students and teachers. In addition, Combien? produces a classification of the exercises according to their level of difficulty and also groups students according to the type of errors they made, and their success rates to the exercises. Teachers use these classifications to define learning activities adapted to each group.

**Experience 2: The Pépite Project (Pepite, 2000)**

Pépite is an application that collects students' answers to a set of exercises and builds a cognitive profile of their competence in algebra. It is based on an educational research that identified learning levers and obstacles in students’ algebra learning (Delozanne, Prévit, Grugeon, & Jacoboni 2003; Delozanne, Vincent, Grugeon, Gélis, Rogalski, & Coulange, 2005). A teacher gives a test to her students and Pépite provides her with three outcomes: first an overview on the whole class by grouping her students...
A Structured Set of Design Patterns for Learners' Assessment

Figure 1. One problem-solving step with Combien?

according to identified strong and weak points, second with a detailed report of each student’s cognitive profile and third with a list of learning activities tailored to each group.

Four hundred students took the Pépite test. Several classes of users used the Pépite diagnosis. Math teachers used Pépite to monitor learning activities in the classroom but also as a basis for a dialog to give a personalized feedback to a single student and as an entry to a meta-reflection on her algebraic competence. Educational researchers used Pépite to identify stereotypes of students and define appropriate teaching strategies to each stereotype. Designers used these experiments to improve the software design: collecting more students’ answers helps to strengthen the automatic diagnosis; analysing teachers’ uses helps to better understand teachers’ needs and to offer a better support for teachers’ activity.

Pépite automatically collects students’ answers to the exercises. Answers are expressed by algebraic expressions, by a whole algebraic reasoning, by using students’ own words, by multiple choices or by clickable areas. The students’ assessment is a three-step process. First, each student’s answer is
coded according to a set of 36 criteria on 6 dimensions (Figure 2): treatments (correct, incorrect, partially correct, not attempted, not coded), meaning of letters (unknown, variable, generalized number, abbreviation or label), algebraic calculus (e.g., correct usage of parenthesis, incorrect usage of parenthesis, incorrect identification of + or x, etc.), conversion (ability to switch between various representations: graphical, geometrical, algebraic, natural language), type of justifications ("proof" by example, proof by algebra, proof by explanation, "proof" by incorrect rule), numerical calculation.

The Pépite software automatically codes 80% of the answers. For each student, an XML file stores the students’ answers and the system coding for each exercise. A graphical interface enables the teacher to check or correct the system coding. Second, a detailed report of the student’s cognitive profile is built by collecting the same criteria across the different exercises to have a higher-level view on the student’s activity. It is expressed by success rates on three dimensions (usage of algebra, translation from one representation to another, algebraic calculus) and by the student’s strong points and weak points on these three dimensions. Third, the student’s profile is used to evaluate a level of competence in each dimension with the objective to situate the student in a group of students with “equivalent” cognitive profile. By equivalent we mean that they will benefit from the same learning activities.

![Figure 2. Pépite automatic coding of a student’s answer on six dimensions](image)
For instance, a student is “Usage of algebra level 1, Translation level 2, Algebraic Calculus level 2” when she used algebra to justify, to generalize and to formulate equations, she sometimes articulated relations between variables with algebraic expressions and linked algebraic expressions to another representation, she showed abilities in algebraic calculus in simple and well known situations but she still uses some incorrect rules.

**Experience 3 The Logic-ITA**

The Logic-ITA (Merceron, & Yacef, 2004) is a web-based tutoring system to practice formal proofs in propositional logic. It is based on the Cognitive Load Theory – the practice of many exercises should help students to build solving problems schemata, see (Sweller, van Merrienboer, & Paas, 1998).

The Logic-ITA has been used by hundreds of students from Sydney University in their second year of studies. It is offered as an extra resource to a face-to-face course. The system has a database of exercises, an exercise generator, and students can also enter their own exercises. A formal proof exercise consists of a set of formulas called premises and a special formula called the conclusion. Solving an exercise is a step-by-step process where students have to derive new formulas, using premises or already derived formulas and applying logic rules to them until they reach the conclusion. The system integrates a module with expertise on logic. It automatically evaluates each step of the student’s solution. In particular it checks that the logic rule chosen by the student is applicable and that the result of its application does match the formula entered by the student. The system provides the student with a contextualized feedback and gives her a hint in case of mistake. Students practice as they wish, training is not assessed nor marked. There is neither a fixed number nor a fixed set of exercises made by all students. For each exercise attempted by a student, the system records in a database the time and date, all the mistakes made, all the logic rules that were correctly applied.

![Figure 3. Screenshot of the Logic-ITA while a student is about to reach the conclusion](image)
used, the number of steps entered and whether the student has successfully or not completed the exercise. The database can be queried and mined to find pedagogically relevant information.

A Structured Set of DP

In this section, we present a subset of DPULS DPs on Learner’s Assessment we derived from our experience. The DPULS format was defined to build automatic browsing and searching tools. We present the DPs in a simplified format to ease the communication. Each DP has a name that describes both a problem and a solution (Alexander et al., 1977; Meszaros, & Doble, 1998). Combining all the DP names forms a pattern language; you can use the DP names to describe your design to solve problems in your context. In the Learner's Assessment category we structured our set of DPs in five hierarchies. In each hierarchy, the DP at the top deals with a problem and all the DPs below are alternative patterns to solve the same problem with different or complementary approaches. Bottom level patterns are more specialized than upper DPs. Our five hierarchies are not stand-alone; patterns can be combined to solve a design problem. Let us consider Sarah’s scenario.

To provide Sarah with an overview on students’ activity during the lab session, Aminata, a member of the LMS design team starts with the DP “overview of the activity of a group of learners on a set of exercises,” (DP LA4, top of the fourth hierarchy). She reads the DP solution description and finds that there are several kinds of overview (lower DPs in the fourth hierarchy) according to the user's objective when asking for an overview. She decides to get more information about the users’ needs. Then, in the related patterns section, she notices that this DP has some pre-requisites. This DP needs the results of the DP “overview of a learner's activity across a set of exercises,” (DP LA2, top of the second hierarchy). Indeed the group evaluation is based on each individual’s evaluation in an individual learning system. Likewise, there are several alternative or complementary overviews on a student's activity (lower DPs in the second hierarchy) and this DP uses the results of the DP “analysis of a learner's solution on a single exercise” (DP LA1, top of the first hierarchy). This DP gives several solutions to derive information from the students' logs.

Figure 4 shows the subset of DPULS DPs focusing on Learners’ Assessment. In the following section, we detail only the patterns mentioned in the above scenario to help Aminata solve her problem. In each pattern we illustrate the results with the three AIDA experiences: Combien?, Pépite, Logic-ITA.
**DP LA1 Multidimensional Analysis of a Learner’s Solution to a Single Exercise**

*DP LA1.1 Pattern matching to analyze the learner's solution*
*DP LA1.2 Specific software to analyze the learner's solution*
*DP LA1.3 Human assessor to check the automatic analysis of the learner’s solution*

**DP LA2 Overview on a learner’s activity across a set of exercises**
*DP LA2.1 The Learner’s strong and weak points*
*DP LA2.2 The Learner’s situation on a predefined scale of skills or competence*
*DP LA2.3 The Learner’s Progression in an Individual Learning Activity*
*DP LA2.4 The Learner’s Autonomy in an Individual Learning Activity*
*DP LA2.5 The Learner’s Performance in an Individual Learning Activity*

**DP LA3 Overview of the activity of a group of learners on a single exercise**

**DP LA4 Overview of the activity of a group of learners on a set of exercises**
*DP LA4.1 Automatic clustering*
*DP LA4.2 Relations between errors, success or usage*
*DP LA4.2.1 Association rules*

**DP LA5 Playing around with learning resources**
*DP LA5.1 Browsing Use of a MCQ*

**Figure 4.** The structured set of the Learners’ Assessment DPs  
(in bold the subset discussed here)

**DP LA1 Multidimensional Analysis of a Learner’s Solution to a Single Exercise**

*Abstract:* This pattern provides several approaches to automatically assess a learner’s solution to an online solved problem. You can merely assess the correctness of the solution or enrich the assessment by other dimensions such as strategy used, abilities, hesitations, categorization of errors etc.

*Context:* The learner answered a single question or solved a single exercise. The answer was recorded as well as usage data (time spent, actions, help requests, etc.). The local analysis of the learner’s answer can be immediate (e.g., if your system provides feedback) or delayed (e.g., in a diagnosis system).

*Problem:* How to automatically assess a learner’s solution to a problem or one step of a problem solution? Or if it is not possible, how to support human assessment? The problem is how can the system analyze, correct, comment on or classify the learner’s answer? If your system asks learners to solve complex problems, your system will let them build their own solution. In that case it is often impossible to predict the exact form of the learner’s answer because of the excessive combination of possibilities or because of learners’ cognitive diversity.

*Solution:* If your objective is to assess the correctness of a learner’s answer then you can provide an indicator like a grade or a code for success, failure or partial failure. If your objective is to provide feedback or to have a cognitive diagnosis, you may need a deeper characterization of the learn-
er’s answer. For instance, you may need information on the strategy used by the learner or on the skills put in evidence by the learner’s solution or on the categories of mistakes made by the learner or on the learner’s hesitations. This multidimensional characterization of the learner’s solution is often domain dependant. The analysis builds a composite indicator – it is often a set of codes identifying the learner’s solving process and a list of errors.

**Results:** If you implement this pattern, you will characterize the learner’s answer with some of the following items:

- A grade to a learner’s answer to an exercise. The grade can be a mark like A, B, C, D, or a message like correct, partially correct, incorrect;
- Cognitive characteristics of the learner’s answer;
- A set of codes (identifying the learner’s solving process);
- A list of errors.

**Discussion:** You may need an expert (experienced teacher, a cognitive psychologist or an educational researcher) to define a model of competence, a task model and/or a typology of mistakes linked to the problem to be solved by the learner.

**Examples**

- Pépite automatically computes a code to assess the answers on up to six dimensions: correctness, meaning of letters (unknown, variable, generalized number, abbreviation or label), algebraic calculus (usage of parenthesis, etc.), translation between various representations (graphic, geometric, algebraic, natural language), type of justifications (“proof” by example, proof by algebra, proof by explanation, “proof” by incorrect rule), numerical calculation (Figure 2). For instance an answer qualified as “partially correct, incorrect use of parenthesis and algebraic justification” is coded by “T2 M31 R1.” Similarly “T3 M4 R3” stands for “incorrect, incorrect identification of + and x, justification by numerical example.”
- Combien? computes an immediate feedback provided to the student on the correctness or on the types of errors (on average, twenty types of errors for each of the fifteen classes of problems). Combien? also stores usage data: action timestamp, action type (e.g., student’s input, asking for help, asking for a random drawing), action parameters.
- The Logic-ITA computes an immediate feedback to the student on the correctness or the error type. It also stores timestamps, a list of the errors made and a list of the rules correctly used.

**Related Patterns:** DP LA1.1, DP LA1.2, DP LA1.3.
**DP LA1.1 Pattern Matching to Analyze Learner’s Solution**

*Abstract:* In some types of problems a pattern matching approach can help you to assess the learner’s solution.

*Context:* It is the same as in LA1. This solution is relevant when an analysis grid is available for this exercise, providing patterns of answers, for instance when expected answers are multiple choices, arithmetic or algebraic expressions.

*Problem:* How to use a pattern matching approach to help you analyze a learner’s solution?

*Requisite:* You need indicators built from a pedagogical or didactical or cognitive analysis. For instance:

- A grid of correct answers. When there is a single way to express a solution in the system, an analysis grid gives correct and wrong answers. For Multiple Choice Questions, your system has to be provided with a grid of correct answers.

- Pattern of good answers or common learners’ answers. When there are several ways to express a solution, a pattern gives a general model of this solution.

*Solution:* For a class of problems, a pedagogical, cognitive or didactical analysis provides you with a set of criteria to carry out a multidimensional characterization of a pattern of solutions. Thus when you can match the learner’s answer with one pattern of solution, you know how to characterize the solution.

*Results:* See LA1.

*Discussion:* For open questions, it is hard work to provide patterns of solutions nevertheless it is sometimes possible.

*Example:* A very simple example in Pépite is: if E = 6P is a good answer, the system also accepts E = 6 * P, P = E / 6 etc. Thus, E= 6*P is a pattern for a correct answer: every algebraic expression equivalent to E= 6*P is correct and assessed as “correct, translating in algebra, correct use of letters”. But P = 6 E or E + P > 6 are incorrect and assessed as “incorrect, translation by abbreviating, use of letters as labels.”

*Related Patterns:* This pattern is a specialization of DP LA 1.

**DP LA1.2 Specific Software to Analyze the Learner’s Solution**

*Abstract:* Domain specific software can be used to help you analyze a learner’s solution.

*Context:* It is the same as LA1. In some specific domains like programming language, formal logic, mathematics, specific software (e.g., a compiler, a problem solver, an expert system, a Computer Algebra System) assesses the correctness and eventually gives information about errors.

*Problem:* How to use specific software to analyze a learner’s solution?

*Solution:* You can use the specific application to check the accurateness
of the solution until the solution is totally correct. Some applications provide error messages and thus you can use them to characterize errors.

Results: See LA1.

Examples: Combien? and Logic-ITA use an expert system:

• In Combien? the learner builds a solution to a combinatorics exercise. Each step of the solution is analyzed by the system using a “targeted detection of error” (Giroire, Le Calvez, Tisseau, Duma, Urtasun, 2002). If the learner’s construction cannot lead to a correct solution, Combien? provides hints to help the student achieve a correct solution.

• In the Logic-ITA, an exercise is a formal proof. The learner has to derive the conclusion from the premises using logical rules and producing intermediary formulas. The expert system checks whether each intermediary formula entered by the learner is correct and provides appropriate feedback. In case of a mistake, a hint is given to help the learner correct the mistake and enter a correct formula. The mistake is stored by the system. Otherwise the system stores the correct use of the logic rule.

Related Patterns: This pattern is a specialization of DP LA 1.

DP LA 1.3 Human Assessor to Check the Automatic Analysis of the Learner’s Solution

Abstract: Either the teacher herself assesses the answer, or the teacher completes, or verifies, or corrects the system's assessment of the learner’s answer.

Context: It is the same as LA1. This solution is relevant if the learners’ solution is composite, or if learners answered in their own words, or if the diagnosis expertise is not yet formalized, or if it is for teacher’s training purpose.

Problem: How to assess the learner’s solution when the automatic diagnosis failed or has a low level of confidence?

Solution: Your system provides a human assessor with a list of exercises where the automatic analysis failed or is not 100% reliable. Then it provides an interface to assist the human assessor.

Results: See DP LA1.

Discussion: If a large number of learners are enrolled in your course or if the teachers are very busy (and it is often the case) this solution is unrealistic because it is time consuming. But it is successful if you need a very accurate assessment of individual learners in a research context or a teacher development context for example.

Examples: Pépite does not fully assess the solution when learners use natural language and it provides a software tool in order to allow teachers to correct, verify or complete the automatic diagnosis.

Related Patterns: This pattern is a specialization of DP LA 1.
DP LA 2: Overview of a Learner’s Activity Across a Set of Exercises

Abstract: This pattern offers several approaches to provide different stakeholders with a general view of a learner’s work across a set of exercises or a whole course.

Context: The learner is asked to solve complex problems in an individual learning system or a diagnostic system. In both cases, the system collects the learner’s answers and assesses them. The objectives of this assessment are various, for example to group learners for remediation, to make learners aware of their strong points and weaknesses or to situate themselves on a predefined scale of competence.

Problem: The problem is: how can the system give a general view of the learner’s work successes, failures or cognitive profile? Strategic decision-making requires a high level description of a learner’s activity. For instance, in order to tailor learning situations, teachers need a synthetic view on the learner's learning activity or an account of the learner’s evolution. A classification of the learner on a predefined scale of competence may be useful to organize working groups or for learners to situate themselves according to expected skills. Thus, you want to define the main features that summarize the learner’s competence. In a diagnosis system, the general view is an instantaneous picture of the learner’s competence, in a learning environment the learner’s evolution over time can be analyzed.

Requisite: See DP LA 1 results.

Solution: To solve this problem you may collect the learner’s answers to a set of questions, exercises or to a whole course. You must first carry out the analysis of each answer on every exercise. Then, you define the different dimensions you need (with teachers or researchers) and, finally, you build an overview according to your stakeholders’ objectives. For example, one may decide to determine the learner’s strong points and weaknesses, or to situate the learner on a scale of competence, or to report on the evolution of a particular skill during the course.

Results: A synthetic analysis of the learner’s competence.

Discussion: It is crucial to define the dimensions of the overview and to pay careful attention to your stakeholders’ needs and how they will use the overview.

Examples:
• Combien? and Logic-ITA summarize items such as the global time spent, the number of exercises attempted, succeeded, failed, the errors made, the rules correctly used etc.;
• Pépite builds a cognitive profile of the learner’s competence in algebra (see DP LA 2.1 results).

Related Patterns: DP LA1, DP LA2.1
DP LA. 2.1 The Learner’s Strong and Weak Points

Abstract: To highlight strong points and weak points in a learner’s performance on a set of exercises is a frequent way to build an overview of a learner’s work.

Context: It is the same as LA2. The learner was asked to solve several exercises involving different skills.

Problem: How to define a learner’s strong points and weaknesses?

Requisite: See DP LA 1 results.

Solution: After assessing each answer to each exercise, you can have a cross analysis on a whole session or on a whole period of time. First, you calculate the success rates for each skill involved in the set of exercises (or in the course). Then, for each skill or error, you calculate the number of occurrences. Sometimes you may also need the context where the skills were obvious or where errors occurred. In that case you need a categorization of the exercises or a categorization of errors.

Results:

• List of skills and success rates on these skills;
• List of errors and number or context of these error occurrences.

Discussion: It is very important to have both strong points and weaknesses, and not only the learner’s errors. If you make a report, highlighting strong points encourages the student, and even the teacher. If you want to help the student to progress, teaching strategies may be different according to the learner’s mastered skills.

Example: Pépite offers an overview by providing a level on three dimensions (usage of algebra, translation from a representation to another, algebraic calculus) and, for each dimension, it provides strong points (a list of mastered skills with success rates and the context where they become obvious). It also provides weak points (a list of categorized errors with their frequency and contexts of occurrences).

Related Patterns: This pattern is a specialization of DP LA2.

DP LA. 4 Overview of the Activity of a Group of Learners on a Set of Exercises

Abstract: This pattern provides teachers with a general view of a group of learners’ work on a set of exercises.

Context: A set of exercises, either linked to a specific course or composing a diagnosis system, has been completed by a group of learners.

Problem: How can your system produce a general view of a group of learners' work on a whole set of exercises? Strategic decision-making may require a synthetic view of learners' work on a whole activity. For instance, this view can help to organize work groups in a classroom. It can help teachers to reflect about the quality and adequacy of exercises and course mater-
ial for example if it shows that some mistakes co-occur. It provides instructional designers or teachers with information that could help them improve their teaching or their design (See DP MV1).

**Requisite:** See DP LA1 Results.

**Solution:** If your objective is to group learners by abilities with respect to a grid of competence or with respect to learning objectives, then you will produce a map of the class. This map will be a classification of learners into different groups. You may predefine stereotypes. A stereotype is a means to identify groups of students who have the same characteristics. These characteristics can be as simple as a high mark, an average mark and a low mark. Or a stereotype may be more accurate and may describe learners who master skills C1, C2 and C3 and who do not master skills C4 and C5. Then you map each learner in the group she belongs to according to the analysis results you have got for each exercise and the relation you have defined between exercises and skills.

You may also let the system find groups for you. This may be done by using an automatic clustering algorithm from the Data Mining field. If your objective is to get an overview of learners’ performance, you can produce statistics or charts that group exercises by chapters or abilities. If your objective is to detect associations such as 'if learners make mistake A, then they will also make mistake B,' or 'if learners fail on exercise E, then they will also fail on exercise F,' then you may use an association rule algorithm from the Data Mining field.

**Results:** If you implement this pattern, you will characterize the activity of your group of students with some of the following items:

- A grade book for the whole class and statistics on all students’ performance;
- A map of the class, grouping learners by abilities;
- Links between mistakes that often occur together, between exercises often failed or succeeded together.

**Discussion:** Stereotypes can be very simple (low achieving, regular, high achieving students), multidimensional (ranking students on a multidimensional scale for instance in Second Language) or describing usage (player, systematic learner, butterfly, etc.)

**Examples:**

- In Pépite, stereotypes are used to classify learners by group of abilities and to offer a cognitive map of the class, grouping students by stereotypes.
- With the Logic-ITA, automatic clustering is used to find out whether failing learners can be split into different groups for better remediation.

**Related Patterns:** This pattern is specialized by DP LA4.1 and DP LA4.2. If your objective is to improve the course, see the DP MV1 hierarchy (DPULS, 2005, 6).
**DP LA 4.1 Automatic clustering**

*Abstract:* Use automatic clustering to get an overview of a group of learners’ activity on a set of exercises.

*Context:* It is the same as LA4. When you have a characterization of each learner by a set of attributes like the exercises she passed or failed, the mistakes made, the abilities she masters or lacks, then you can use automatic clustering to split the whole group of learners into different homogeneous groups.

*Problem:* How to get an overview of a group of learners using automatic clustering? You want to organize groups in a classroom. Whether it is better to work with homogeneous or heterogeneous groups is a pedagogical decision. In both cases, you have first to build homogeneous groups; for example, a group should be formed with learners who are similar for some characteristics. However, you do not have any predefined stereotypes, or you want to explore whether some other grouping would be sensible. You may try automatic clustering.

*Requisite:* As for LA4.

*Solution:* For each learner you have some analysis provided by the system on each answer submitted for each exercise. You concatenate these analyses, or you summarize them to obtain a characterization for each learner. For example, each learner can be characterized by the fail or pass obtained on each exercise. Another simple characterization would be the total number of mistakes made on all attempted exercises. From these characteristics, you select the ones that should be used for the automatic clustering algorithm and you run it. The automatic clustering algorithm produces clusters of learners. All learners belonging to one cluster are similar with respect to the characteristics that you have chosen to run the algorithm.

*Results:* Thus you will obtain a map of the class with students grouped in clusters.

*Discussion:* To use automatic clustering, you need some expertise in the Data Mining field. Choosing the right characteristics to run the algorithm and to interpret the resulting clustering is a crucial and difficult point.

*Example:* In the Logic-ITA experience, we have used automatic clustering to explore whether failing learners can be grouped in homogeneous clusters. Failing learners are learners who attempt exercises without completing them successfully. As a characteristic we have used the number of mistakes made. The result was two clusters – learners making many mistakes and learners making few mistakes. In the cluster of learners making many mistakes, one could identify learners with a “guess and try” strategy, using logic rules one after the other till they hit the right rule to solve an exercise.

*Related Patterns:* This pattern is a specialization of DP LA4.
**DISCUSSION**

How to evaluate a set of DPs? This is a hot issue in DP communities (Avgeriou et al., 2004; Buschmann et al., 1996; Chung et al., 2004; Salin-garos, 2000; E-LEN project; Todd, Kemp, & Philips, 2004). Most Pattern sets described in the literature are validated through a peer review process. Some authors suggest that a pattern can be accepted by a community if it is used in three experiences other than the one that proposed the pattern (Buschmann et al., 1996). In this section, we discuss how our DP set accounts for solutions adopted in other experiences. We checked the validity of our DPs in a three step process. In a first step, we discussed the patterns within the AIDA team. In a second step, the DPs have been discussed and refined within the DPULS consortium. In a third step, we looked at the AIED’05 workshop on “Usage Analysis” (Choquet, C., Luengo, V. & Yacef, K., 2005) contributions that deal with assessment to investigate whether these experiences match our approach.

We worked out the above patterns to generalize some success stories in the AIDA team namely the three systems presented in the second section and three others: Diane, Math Logs and Java Course. Diane is a diagnosis system. It identifies adequate or erroneous strategies, and cognitive mechanisms involved in the solving process of arithmetic problems by children from elementary schools (8-10 years old) (Hakem, Sander, & Labat, 2005). DP LA1 and DP LA2 account for Diane’s approach to Cognitive Diagnosis. Math Logs provides researchers with information about undergraduate students’ performance on mathematical exercises displayed on the Wims platform. The mathematical expressions entered by learners are checked by Computer Algebra Systems (CAS) as MUPAD, PARI, Maxima (accounted by DP LA 1.2). Math Logs displays to teachers average grades, average time spent by type of exercises, and indicators on the evolution of grades over time. It detects students’ usage strategies such as focussing on easier exercises or preferring challenging exercises (DP LA2, LA3 and LA4). The Java Course is an online course for introductory programming in Java. Students write programs and submit them for compilation and execution. Their programming and compiling errors are stored in a Database (DP LA1.2). For each exercise and each student, the system displays whether the exercise has been attempted, passed, the number of mistakes made and provides an overview for a chapter and for the whole set (DP LA2).

From the AIED’05 workshop on Usage Analysis of Learning Systems (Choquet et al. 2005), we selected six papers dealing with assessment.

Kumar (2005) presents a C++ Programming Tutor. In this system, each exercise involves exactly one learning objective. The system automatically checks whether the student’s answer is correct, partially correct, incorrect, missed or not attempted (DP LA1.1). Then an overview of each student’s work is produced.
This overview gives for each learning objective the fraction of exercises solved correctly, partially correctly etc. (DP LA2). For a whole class, the average performance for each exercise is computed (DP LA3) and, finally, the average performance of the class on each learning objective is also calculated (DP LA4). Designers use the latter to refine the templates to automatically generate the exercises according to the objectives (DP MV1 “Material Improvement”).

Feng and Heffernan (2005) describe Assistment, a math web-based tutor that provides the teacher with a report on the student’s activity and the student with assistance when she is stuck in solving problems. This assistance is given by scaffolding questions that are intended to help the student but also to help the diagnosis system to understand which knowledge components are involved in the failure when the problem involves more than one component (DP LA1). Then the system builds a grade book providing the teacher with each student’s results (DP LA2) and with an overview of the class (DP LA4). Another overview of the class is given by a Class Summary Report and a class Progress Report. Then an analysis of items helps determine what the difficult points are for students and improve both teaching and materials (DP MV1 “Material Improvement”). This system design is a very clever example of the best practices we wanted to account for in our DP Set.

Nicaud, Chaachoua, Bittar, & Bouhineau, (2005) model the student’s behavior in algebra calculation in Aplusix, a system to learn elementary algebra. The diagnosis is a two step process. The first phase is a local diagnosis of each student's transformation of an algebraic expression. From a library of correct and incorrect rules, the system determines the sequence of rules used by students during the transformation. The diagnostic system also characterizes the type of exercise and the algebraic context in a “Local Behavior Vector (LDV)” (DP LA1, for one step). The second phase uses a lattice of conceptions to build an overview on a student’s conceptions from the different LDV on a set of algebraic transformations (DP LA2). Aplusix offers to teachers a map of conceptions for the whole class (DP LA4).

Heraud, Marty, France, & Carron, (2005), Stefanov & Stefanova (2005), and Muehlenbrock (2005) do not focus on learners’ assessment. They describe approaches to build an overview of a learner from different perspectives. They work with a trace of a student’s actions in an e-learning platform. The trace records success and task completion on exercises, time stamp and course consultation or helps (DP LA1). Stefanov & Stefanova (2005), use a learning object approach in Virtuoso to provide students, teachers and designers with statistics on students’ weak points and strong points (DPLA 2.1), on success rates or failures to each learning objects (DP LA3) to improve the material (DP hierarchy MV). Muehlenbrock (2005) uses decision trees to classify students in three categories: low, medium or high achieving students (DP LA4). To have an overview on the student’s activity on a session (DP LA2) the system described by Heraud et al. (2005)
compares the student’s trace to the prescribed learning scenario. It displays the trace in a band showing the time spent for each task along with the prescribed time. The system shadows the prescribed line to provide what they call “a shadow bar” when there is a discrepancy between the trace and the prescribed time. A human assessor called “trace composer” when aware of this shadow zone in the trace can investigate other sources of information like the student’s logs in the server, the learner’s workstation or human observer. This experience suggests introducing “a human assessor” pattern in DP LA2 hierarchy. It also suggests to refining the hierarchy with usage data while so far we have focused on accurate cognitive information.

This review shows that our approach accounts for solutions adopted in many learning systems at least in the domains of Programming and Mathematics Tutors. Further investigation is needed to see whether our DPs are pertinent for other domains, though it seems that it is the case. For instance Language Standardized Tests (e.g., TOEFL, TOEIC, IELTS’, TFI) use an approach similar to our DPs. On the basis of students’ scores, they classify students on a predefined scale of competence in listening, reading, writing and speaking skills. This fits well with the multidimensional analysis of DP LA1.

In this section we discussed whether our DP collection on Learner’s assessment matches the state of art in the domain. Our objective was to capitalize on success stories and to help designers build successful e-learning systems. We estimate that we validated the first point but it is premature to validate the second. So far only our students used it. For technical reasons our DP set will be public soon on the Web. In particular it will be used in the Virtual Doctoral School of The Kaleidoscope Network. This will provide interesting feedback and hopefully the Pattern Language will be enriched.

CONCLUSION

In this article, we focussed on DPs dealing with Learners’ Assessment, a subset of the DP elaborated in the DPULS consortium (DPULS, 2005, 6). One outcome of our work is to put in evidence basic principles for designers whose objective is to provide teachers with an overview of the activity of a group of learners on a set of exercises, (DP LA4). A fundamental step is to obtain a multidimensional analysis of a learner’s solution on a single exercise, (DP LA1). DP LA1 is specialized by other DPs depending on how the student’s solution is analyzed (pattern matching, use of some specific software, human assessment). It can be used to obtain both an overview of a learner’s activity across a set of exercises, DP LA2, and an overview of the activity of a group of learners on a single activity, DP LA3. The two latter DPs are used by DP LA4. In case DP LA1 is applied to exercises that are characterized by precisely defined skills, DP LA4 gives a way to track students’ problem-solving abilities.
These DPs are high level DPs. A future work will consist in writing more specific DPs to describe how to implement the different solutions. To support LA4 we would like to work out patterns on data mining techniques. A complementary approach to build an overview is used by Pépite, Aplusix, Assistment or Second Language assessment. It consists in situating a student in predefined classes based on cognitive or educational research and not only on achievement.

Further investigation is needed to study what part of the assessment is domain independent in order to be implemented in e-learning platforms. On the opposite, we think that domain specific patterns would give more accurate guidance to designers. For instance, our DP set could be completed with patterns that collect practices on displaying assessment results to different actors in specific domains.

So far this work has demonstrated that DP is a very fruitful approach to generalize from different design experiences and to capitalize on them. It was a powerful integrative project because we had to distill out of our specific projects what was invariant in making good design solutions. “Patterns are very much alive and evolving” (Alexander et al., 1977). We hope that we will have feedback from practitioners to develop and enrich this first DP language on learners’ assessment.

References


A Structured Set of Design Patterns for Learners' Assessment


Deng, J., Kemp, E., & Todd, E. G., (2005). Managing UI pattern collections. CHINZ'05, Auckland, NZ.

deliverable 2: Merceron, A. Report on partners' experiences.
deliverable 3: David, J.P. State of art of tracking and analyzing usage.
deliverable 4: Pozzi, F. The set of recurrent problems and description of solutions.
deliverable 5: Verdejo, M.F., & Celorio, C. The design pattern language structure.


Notes
1 http://www.noe-kaleidoscope.org/ (consulted March 2006)
2 The DPULS consortium involved researchers and practitioners from six European countries and from different domains: educational design, educational research, teachers, engineers, LMS managers and LMS designers.
3 Aida is a consortium of Research Laboratories focusing on e-learning problems in the Paris area. The French Ministry of Research funds it. Aida belonged to the DPULS consortium.
4 It codes every answer expressed by multiple choices and by one algebraic expression, most answers expressed by several algebraic expressions, and some answers in students' own words.
5 International Language Testing System