AperTO - Archivio Istituzionale Open Access dell'Università di Torino

The PP&S Project: Process Control as an Information System Istance

This is the author's manuscript

Original Citation:

Availability:

This version is available http://hdl.handle.net/2318/147211 since

Publisher:

AICA

Terms of use:

Open Access

Anyone can freely access the full text of works made available as "Open Access". Works made available under a Creative Commons license can be used according to the terms and conditions of said license. Use of all other works requires consent of the right holder (author or publisher) if not exempted from copyright protection by the applicable law.

(Article begins on next page)

02 May 2024

The PP&S100 Project: Process Control as an Information System Instance

Claudio G. Demartini[1], Marina Marchisio[2], Marco Mezzalama[1], Claudio Pardini[3], Amelio Patrucco[4]

[1]Politecnico di Torino, Corso Duca degli Abruzzi,24, 10129 Torino
claudio.demartini@polito.it, marco.mezzalama@polito.it
[2]Università di Torino, Via Carlo Alberto, 10, 10124 Torino,
marina.marchisio@unito.it
[3]IIS Carlo Anti, Via Magenta, 7 – 37069 Villafranca di Verona,
dirigente@carloanti.it
[4]Fondazione Torino Wireless, Via Francesco Morosini, 19 – 10128 Torino
amelio.patrucco@torinowireless.it

Abstract *The Project PP&S100, Problem Posing & Solving, is part of a set of initiatives promoted by the General Directorate of the Italian Ministry of Education, Research and University, for supporting the many innovations that have recently affected curricula at the upper secondary level. Main goals of the project include strengthening computer science culture and enhancing its role as a scientific discipline; founding education processes on logics, mathematics and computer science weaved together to pursue an interdisciplinary scenario, building a culture of "problem posing and solving" by investing across a broad disciplinary group of subjects, ensuring growth of computer science based training of trainers and practicing activities within specific social networks and virtual learning environment to share learning materials, teaching supplies, mentoring and self-evaluation. This work presents a hierarchical conceptual model for "Computational Thinking" assumed by PP&S100, formalized according to the outcomes shown by information systems courses held at the Politecnico di Torino and validated using Process Control as an information system instance.*

Keywords: Problem Posing&Solving, Computer Science, Computational Thinking, Process Control, Collaborative Learning.

1. Introduction

The purpose of the project, which gathers among the primary proponents also the Italian Association for Automatic Computing (AICA), the National Research Center (CNR), the National Industrial Unions, the University of Turin and the Politecnico di Torino, is to sustain changes proposed at the regulatory level transforming "programs specified by the Ministry of Education" into national Guidelines for high schools, technical schools and vocational organizations. This change relies mainly on teachers for a clearer knowledge management and independent design of educational routes most suitable for achieving learning outcomes that guidelines declare and set for the entire national territory. The Project PP&S100 [Palumbo and Zich, 2012], Problem Posing & Solving for 100 schools, now grown to 150, is part of a set of initiatives promoted by the General Directorate of the Italian Ministry of Education, Research and University, for supporting the many innovations that have recently affected curricula at the upper secondary level.

It has an immediate impact in the scientific application domains (physics, chemistry, natural sciences and so forth) but also brings a relevant potential within all subject areas, even those placed in the social and humanistic domains, thanks to the basis provided by trans-disciplinary languages and computer science representations.

The project culturally focused on problem posing and solving, aims to tap the invaluable potential of information technology as an enabler of innovation. This already takes place when mathematics and computer science should strengthen one another, being delivered jointly in the same classes, as it is the case in the first two years at high school, where they should work both enforcing mutual outcomes - even if, for truth's sake and most of the times - this happens only on a formal base.

Main goals of the project include:

- strengthening computer science culture, enhancing its role as a scientific discipline, by introducing specific courses in the first two years where they are not now given
- set up education processes integrating logic, mathematics and computer science
- building a culture of "Problem posing & solving", by investing across the broad disciplinary group, within the various curricula, adopting a systematic activity based on the use of the logical-mathematical computer science for the formalization, quantification, simulation and analysis of problems of various complexity.
- ensuring growth of computer science-based training of trainers called to go with the transformation process.
- adopting a relevant amount of social network activity and virtual learning environment, by sharing learning materials, teaching supplies, mentoring and self-evaluation.

The project, lasting three years, has planned to focus the goals looking at the first two couples of years of high school. Its working organization has planned to involve at least 110 institutes of secondary education, which should act as "poles" of territorial networks. At the end of the first year, 150 classes of about 4500 students had already been engaged.

Since the beginning the Project addressed Mathematics and new technologies, also dealing with learning management with an advanced computer environment (ACE). The latter played a basic role to build new teaching plans in Maths with respect to the framework drawn by problem posing and solving. The Science Faculty of the University of Turin made a great experience in this field and, thanks to it, put in place a Moodle asset and a Maple suite for the project itself. Both of them, had been built tightly interlaced and were successfully used for several years in teaching Mathematics for all the students coming from the same Faculty, to whom a part of those attending other courses joined soon.

The whole infrastructure started first with a Moodle virtual learning environment, always available at the address http://minerva.i-learn.unito.it/. Then that Moodle platform grew integrating a Maple Suite, consisting of an advanced symbolic computing environment and MapleNet, supporting file distribution in such a way that interaction remains active even if Maple is not running locally on the computer. MapleTA added to soon, being a tool apt to develop tests and assignments, including definition of open mathematical questions and self-evaluations, and, in a while, MapleSIM followed, being useful to carry out virtual labs in Physics, Electronics, Mechanics, Thermodynamics and so forth. The whole system was able to sustain more than 150 teachers, which gave birth soon to a Community, taking advantage of the Moodle platform, which made a strong collaborative learning scenario possible, fully and constantly practiced.

An effective tutoring and coaching service was also established addressing teacher learning to pursue the constructivist approach [Brooks, & Brooks, 1999] in the classroom and to support products development to be shared on the Moodle platform. Since February 2013 more than 150 students communities grew in the project, where they could learn according to a multifaceted perspective: at school during their classes and in the labs, but also at home, by trying to solve problems together, understanding them first, according to the posing perspective, and choosing a suitable algorithm to carry out a possible solution, most of the times working inside the learning management platform as reported in [Zich et al].

A few classes – belonging to 10 high schools and 10 technical institutes, selected on a voluntary basis – are planned to start in the second project year (2013/2014). They are to provide for the breeding ground to put in place a proposal for a new discipline intended to deepen information and computer science: a first step towards the creation of a process able to deploy the same experience after the project ending. The discipline is being organized according to a framework articulated into two lab modules spread over two years of high school curricula.

Section 2 introduces emerging scenarios for computer science in the school, focusing on the layer abstract model assumed as a reference in the project. Section 3 describes the Computational Thinking Living Lab arguing on Process Control as an information system instance to validate the abstraction model, while Section 4 reports on concluding remarks and future work.

2. Learning Computer Science in Education: Emerging Scenarios

Inquiry based Science Education and Learning [Olson and Loucks-Horsley, 2011] can be considered as the result of different contributions given by several domains such as, among others, "constructivism" [Brooks and Brooks, 1999] [Marlowe and Page, 2005], "Bloom’s taxonomy of learning" [Forehand, 2005] and "multiple intelligences", without forgetting Paulo Freire’s “Pedagogy of the Oppressed” [Freire, 1993], where the terms “problem posing” occurred for the first time.

Problem posing and solving fits well the computer science discipline, which is able to give proper means, tools, methodologies and theories to describe real and conceptual phenomena. This is mainly because logics, formal languages and physical elementary devices are deeply rooted in the founding discipline principles.

A glance at the rest of the world [Peyton-Jones et al, 2011] shows that many other countries have expressed the same creative impetus that already hit England some years ago [Livingstone and Hope, 2011]. In fact, excitement around computing led UK to introduce this subject into schools curricula, making of that event an effective representative experience for all other countries. In particular, European countries [Gander et al, 2013], together with many others around the world, are now trying to collect information from UK, because they came to share what U.S.A. anticipated in their White Paper since 2005:

- a clear distinction between computer science as a discipline and rigorous computer applications or digital literacy should prevail
- recognize that all children and young students should learn information science in the same way they learn science or mathematics.
- be aware that growing demand for ICT professionals from the labor market reduces the potential supply of qualified teachers.

2.1 Computational Thinking

Computational Thinking (CT) is often elicited as a best practice of inquiry based science approaches, as defined by J. Wing: ”*Computational thinking is the reasoning processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent*” [Wing, 2008]. This definition can be assumed as a reference point with respect to the objectives pursued by the PP&S100 project, as the terms “formulating problems” and “formulating solutions” suggest.

The most important complex mechanism in CT is the abstraction process, which is used, well deepened and represented in Unified Modeling Language UML or Entity Relationship (ER) model, to define patterns, specializing entities or generalizing others, leaving from objects or other simpler entities (“generalization” abstraction and its converse, the so-called "specialization"). It is also used to capture essential common properties of a set of objects applying an abstraction procedure named "classification". Also an algorithm is usually given as an abstraction of a process having a determined objective, which operates executing a sequence of steps, working on a set of inputs to produce a corresponding set of outputs. A process itself can be considered an abstract data type representing a set of values and operations designed to manipulate those values, which are not visible from outside of that abstract data type representation. From that drawing comes out that efficient algorithms cannot be designed without using appropriate abstract data types to handle input data or internal temporary ones to be provided to the same process. Well depicted by both UML and ER models at least, abstraction gives analysts the power to deal with complexity scaling the representation of reality, leaving from the bit representation, as the lowest description level, up to the largest system you can find in realty. The hierarchical abstraction design follows a hierarchical sequence of layers, each one featuring its own specific abstraction level, thus recalling the communication domain where a layered architecture allows segmentation of services assigning specific functionalities to any layer to keep it disjointed from the adjacent ones. This enhances modularity, in fact when implementing or even changing functionalities, a specific layer is chosen to focus on, without affecting others, since adjacent interactions take place through specific inter-layer services. The loosely structured architecture of the Internet shows how effective and robust may be a well-formed abstraction. An example is provided by the TCP/IP layer, which, placed over the network and under the session layer has enabled a multitude of applications to proliferate at layers above, and a multitude of different platforms, media, and devices to operate at lower layers.

Fig. 1 depicts the model drawn to introduce the rationale of the computer science discipline to be developed in the project, fully adopting the "Inquiry based Science Education model" and its specialization represented by the "Computational Thinking" scenario. The most abstract modeling phase is placed at layer 4 (l4), the Problem Posing Layer, where the model is shaped through an inquiry-based process focusing on main questions linked to the problem to be understood. To give a flavor of that layer functionality, the Logical Framework Analysis [EU, 2004] has been recalled, to emphasize the fact that a collaborative interaction should also be taken into account and put in place among all the stakeholders involved in the problem.

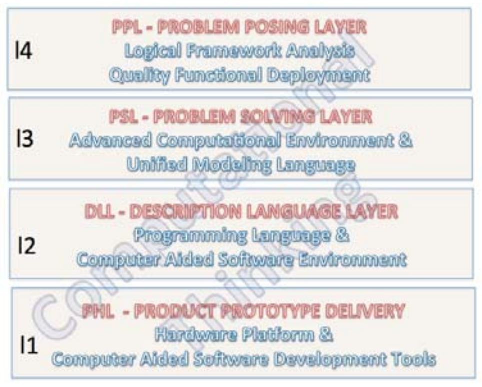
l4
PPL - PROBLEM POSING LAYER
Logical Framework Analysis
Quality Functional Deployment
l3
PSL - PROBLEM SOLVING LAYER
Advanced Computational Environment &
Unified Modeling Language
l2
DLL - DESCRIPTION LANGUAGE LAYER
Programming Language &
Computer Aided Software Environment
l1
PHL - PRODUCT PROTOTYPE DELIVERY
Hardware Platform &
Computer Aided Software Development Tools
Computational Thinking

Fig. 1 – CT and the Layer Abstraction Model

The next layer (l3) deals with the specification of the solution to be proposed for the problem posed at the previous one. Algorithms and data representation should be given there according to a formal abstract description, being they better shaped using specific environments, such as MAPLE/MAPLE SIM or

MATLAB/SIMULINK [Chonacky and Winch, 2005], LABVIEW [National Instruments, 28/8/2013] and UML [Wikipedia, 28/8/2013].

MAPLE, MATLAB and LABVIEW, together with their linked applications, may be considered as appropriate examples of Advanced Computation Environment (ACE) to specify mathematical models for complex phenomena. In parallel, for non-mathematical domains, in a complementary perspective with respect to ACE, the Unified Modeling Language [Unified Modeling Language, 28/8/2013] and IDEF (Icam DEFinition for Function Modeling, where ' ICAM' is an acronym for Integrated Computer Aided Manufacturing) [IDEF0, 28/8/2013], are also introduced. Thus a multi-faceted model description can be carried out, representing the same applications/phenomena by means of use-cases and activity diagrams to describe functionalities, class diagrams for static objects description, sequence diagrams for tracing object dynamic interactions and so forth. Both scenarios, mathematical and non-mathematical, are also able to sustain high-level simulations. In the most cases both environments can also translate given formal models descriptions into a first draft of the programming language description for the same systems/phenomena. These descriptions are handled at layer two (l2), where a more detailed picture can take care of the various aspects also linked to the hardware platform to be used subsequently at layer one (l1). At level two more precise simulations can also be planned, being given the infrastructure already traced in terms of operating systems, network features, file system and so forth, relying also on compilers (C, C++) or interpreters (Python, Scratch) to correctly implement the semantics of the language. To cite an example for all in terms of CASE tools, Visual Paradigm [Visual Paradigm, 28/8/2013], supporting also ER-Model based environments, and Eclipse [Eclipse Project, 28/8/2013] could be used as a specific reference. Hence, as shown by the experience already made at the Information Systems Course at Politecnico, conceptually showed in Fig. 1, Computational Thinking stems from both mathematical and engineering thinking. Unlike mathematics, where constraints take the shape of theorems, corollaries, assumptions, information systems are constrained by the physics of the underlying information-processing units and their related operating environment. This means that boundary conditions, failures, malicious agents, and unpredictability of the real world must also be taken into account, having in mind that, unlike other engineering disciplines like Mechanics or Constructions, Information Science — based on intangible artifacts like software, allows anyone to develop virtual worlds totally unconstrained by physical issues, in fact in cyberspace the only constraint to creativity may be given by imagination.

3. “Computational Thinking” Living Lab and the TestBed

This work deals with the development of a complete test-bed that aims at stating the feasibility of the learning process, based on previously exposed methodologies, when applied to a very simple but relevant context. The label assigned to this experience is “Living Lab”, focused on enhancing the relevance of computer science in the school as a means to represent realty with respect to both static and dynamic perspectives.

The Lab rationale has the following main points:

- check how learners can leverage learning approaches based on Problem Solving and Computational Thinking - being the latter a learning domain emerged recently and often considered in literature as a specialization of Problem Solving - when applied to scientific subjects in the secondary school to create a continuum from mathematics, to computer science, up to other "mathematics-based sciences".
- check how Simulation and Prototyping can act as "accelerators of understanding" of complex systems and physical phenomena.
- prepare a draft syllabus for the first module of Computer Science for the secondary schools, taking care of the features of the different specialization branches
- build an environment for trial studies around a "living-lab" framework, where basic elements are combined to develop prototypes, which span from simple process control systems together with their adjustment parameters, up to more complex systems like robots and organizations management networks.

The multi-disciplinary nature of the whole process is relevant for the objectives pursued by the project, since it encompasses perspectives linked to the "hardware" platform planning, together with a description of the system as a whole, traced at various abstraction levels. To validate what has been first modeled, several simulation phases are carried out, giving space to the following realization, which has to be made available at the final stage in terms of product or service.

To validate the abstraction model introduced in the project, a first example has been drawn using a heating system as an elementary process to which a regulation device should have been adopted. The reason for this choice relies on the easiness with which the system can be modeled and the wide number of applications available in different domains, ranging from home automation, industrial systems, chemical and rural plants, energy savings, and so forth.

3.1 The Driver

In order to make Computational Thinking well understood, and validate the abstraction model drafted above, a tangible and well-known ordinary living experience has been taken as an inspiring background. Since ancient times mankind has always tried to overcome any constraints imposed by the physical properties of being. Any natural phenomena has been analyzed and deepened with the aim of controlling its effects whether to get advantages, when positive, or to reduce its impact, in case of damage.

Hence, Controls and Regulators can be considered as ubiquitous and pervasive mechanisms, used to constraints economic systems behavior, when large-scale systems are considered, but also to control robots, or other smaller devices, when small-scale systems, even though complex, are examined. Controls and Regulators are also immediately perceived within people daily life, several examples do exist like heating and air conditioning systems, showers, and so forth. It is worth not forgetting that man lives because some physical features in its body, like blood pressure, heart beat rate, are regulated naturally, so that they are constrained within certain thresholds. Even when those features overcome those constraints, artificial devices, controllers or regulators, can be provided to force a correct behavior. An insulin pump device is a typical example where insulin levels corrections are executed to regulate glucose level in blood, when pancreas natural control system is not behaving normally. A picture showing what may happen without these controls is given by the consequences of a possible control failure: when blood glucose level is beneath its lower threshold, a loss of consciousness may occur, in contrast, when the level increases above the higher threshold, damage to liver, eyes, and kidneys may happen.

All the previous comments concern the definition of the possible main questions, related to the impact of a specific control device, to be placed mainly at the higher level (l4) of the layered model drawn in Fig. 1, corresponding to the “Problem-Posing” domain. The following layer (l3) is assigned to the “Problem Solving” domain, where a formal specification, representing the solution model, can grow, . While the third layer (l2) offers a translation of the formal specification into a descriptive programming language, layer l1 hosts an effective implementation on the basis of that language and a suitably chosen hardware platform. Hence, if the question “Why are we developing a control system” concerns the analysis of a specific problem, taking care of stakeholders, the impact of caused effects they should sustain, their influence on the domain, their interest to overcome the problem, and so forth, the question “What is a Control System?” should be placed at the second layer, since it concerns the solution to be realized to contrast the problem. Following the inquiry-based approach, it is expected that the next question should be “What is a System?”. Hence moving from it, the IDEF0 methodology is often suggested as a means to provide analysts with a suitable way to shape a graphical abstraction of what in nature, but also within organizations, can be perceived as a system, often understood simply through the analysis of its input and outputs. IDEF0 allows an extended representation of what is ordinarily named as a “block schema”, having the added value offered by three different sets of inputs, the first labeled as {constraints}, entering the box from above and representing the constraints the process has to respect, the other, labeled {resources}, from below, representing all the means used by the system/process when running, while {inputs} represent the set of information and/or raw material entering the process from the left side of the box. The picture is completed by an arrow coming out from the box, placed at the right side, labeled {outputs}, representing material and information processed and thus produced by the process. Always following an abstraction schema, a real

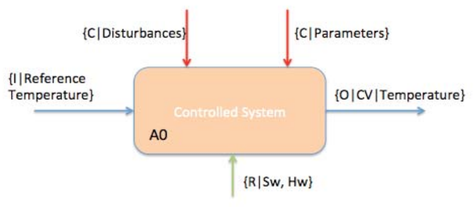

Fig. 2 - IDEF0 block to describe a Controlled System

controlled system can effectively be represented taking advantage of the same graphic representation, focusing on the conceptual model of a particular device, such as that one able to regulate the room temperature, which may be chosen as a first simple example to be dealt with the students.

Future developments will focus on the description of a MapleSim Diagram, choosing appropriate devices. UML diagrams will also be drawn to give an object paradigm perspective. Finally an Arduino platform will be used as a prototyping environment for the physical development.

4. Conclusions

PP&S100 aims at supporting the many innovations put in place by recent regulations adopted by the Ministry and affecting curricula at the upper secondary level of the national education system. One of the main objectives of the project deals with strengthening computer science culture and also enhancing its role as a scientific discipline. To this purpose a model is drawn in this work to shape the rationale of this discipline to be introduced at the secondary school, first cycle, fully adopting the “Inquiry based science education model”, rooted into the constructivist scenario and its specialization, represented also - among others - by the “Computational Thinking” approach. A hierarchical framework is defined using abstraction models shaped as an inquiry-based process and focusing on main questions linked to the problem to be posed and then described according to a “solution” perspective. A four-layer based hierarchy has been considered taking advantage of the project/product life cycle, which is usually founded on a process, which naturally evolves from the highest abstraction level (inception) up to a more practical (implementation and maintenance) phase, involving real world impact in terms of applied actions and sensed physical quantities. To enhance conceptual reading of functionalities provided by each hierarchical level, references to commercial available tools are also taken into account. To make “computational thinking” interpretation easier for large scale education and training systems, some catalyzers have been identified, which can suggest how enhancing and enabling interleaved disciplines domains, through ordinary problems faced by students and teachers in their daily experience.

References

[Palumbo and Zich, 2012] Palumbo C., Zich R. Matematica ed informatica: costruire le basi di una nuova didattica, Bricks, Anno 2, n. 4, ISSN 2239-6187, 2012, pp. 10-19.

[Olson and Loucks-Horsley, 2011] Olson S. and Loucks-Horsley S.” Inquiry and the National Science Education Standards: A Guide for Teaching and Learning, Ed., Comm. on the Devel. of an Add to the Nat. Science Education Stds on Scientific Inquiry, NRC, National Academy Press, Washington, D.C, 2011.

[Freire, 1993] Freire. P., Pedagogy of the Oppressed. Continuum International Publishing Group, 1970,1993.

[Marlowe and Page, 2005] Marlowe B., Page M., Creating and sustaining the constructivist classroom. 2nd Ed. Thousand Oaks, California: Corwin Press Inc., 2005.

[Zich et al] Zich R., Pardini C., Marchisio M. (2012). Moodle&Maple: una struttura integrata al servizio del Progetto MIUR su Problem Posing and Solving. Atti di MoodleMoot Italia, 2012, pp. 1-10.

[Forehand, 2005] Forehand, M., Bloom's taxonomy: Original and revised. In Ed. M. Orey, Emerging perspectives on learning, teaching, and technology, 2005.

[Brooks and Brooks, 1999] M. G. Brooks and Brooks J. G., “The Constructivist Classroom, The Courage to Be Constructivist”, November 1999, Vol. 57, Num. 3.

[Wing, 2008] Wing J. M., Computational thinking and thinking about computing, Phil. Trans., R. Soc., 2008, 366, 3717-3725.

[Peyton-Jones et al, 2011] Simon Peyton-Jones et al, “Computing at School, International comparisons”, Microsoft Research UK, 2011

[Livingstone and Hope, 2011] I. Livingstone and A. Hope, “Next Gen. Transforming the UK into the world’s leading talent hub for the video games and visual effects industries”, A Review by Ian Livingstone and Alex Hope, February, 2011, NESTA.

[Gander, 2013] Gander W., Informatics education: Europe cannot afford to miss the boat, Report of the joint Informatics Europe & ACM Europe Working Group on Informatics Education, April, 2013

[Wikipedia, 28/8/2013] Wikipedia, UML General Information, 12/10/2007, http://en.wikipedia.org/wiki/Unified_Modeling_Language, accessed: 28 August 2013.

[Chonacky and Winch, 2005] Chonacky N., Winch D., 3MS for Instruction: MAPLE, MATHEMATICA, and MATLAB, Computing in Science & Engineering , Vol.: 7, n. 4, July/August, 2005.

[National Instruments, 28/8/2013] National Instruments, LABVIEW, http://www.ni.com/labview, accessed: 28 August 2013.

[IDEF0, 28/8/2013] IDEF0, http://www.idef.com/IDEF0.htm, accessed: 28 August 2013.

[Unified Modeling Language, 28/8/2013] Unified Modeling Language, http://www.uml.org/, accessed: 28 August 2013.

[Visual Paradigm, 28/8/2013] Visual Paradigm, http://www.visual-paradigm.com/product/vpuml/, accessed: 28 August 2013.

[Eclipse Project, 28/8/2013] Eclipse Project, http://www.eclipse.org/, accessed: 28 August 2013.

[EU, 2004] European Commission-EuropeAid Co-operation Office, Guidelines on Aid Delivery Methods, Volume 1: Project Cycle Management, March 2004, Brussels.