in Siraj-Blatchford J. (Ed), Developing New Technologies for Young Children, pp.93-174,
Trentham Books Limited, UK, 2004

Children's playful learning with a robotic construction kit[1]

Augusto Chioccariello, Stefania Manca, Luigi Sarti[2]
Istituto Tecnologie Didattiche – C.N.R.
Via de Marini, 6
16149 Genova
Italy
e-mail: {augusto | manca | sarti}@itd.cnr.it

Introduction

Robotic constructions offer the opportunity to explore a class of cognitively relevant concepts such as emerging behaviours, theory of control, etc. The overall aim of our research was to assess the suitability of a robotic construction kit for young children (5 to 8-year-olds): could children of this age build and program robots on their own? If so, what kind of activities, tools, environments are best suited for the task?

To this end, we focussed on two distinct but parallel approaches: the LEGO MindStorms kit was partially redesigned to increase the typologies of possible constructions (through the design of new sensors, actuators and pre-assembled mechanical components). A programming language was designed that is context specific and extensible, and is therefore capable of facing a range of problems that are tightly related to the typologies of activity for which it has been predisposed. This chapter provides a description of the theoretical background, the adopted research methodology, the evolution of the physical play material, the features of the programming environment, and some selected findings of the field-testing. Possible evolutions of the work are outlined, that would enable children to build and program robots in less supportive contexts.

This chapter describes the work carried over within the *Construction kits made of Atoms & Bits* project (CAB) [Askildsen et al, 2001a], whose objective was to explore and investigate the relationships and attitudes of young children towards behaving objects.

The appeal of ideas such as building constructions that expose interactive autonomous behaviour to preschool and elementary school children was one aspect at the centre of our research. In particular the CAB project strived to:

- experiment with and validate a methodology which fosters children's interaction with computers through the use of cybernetic constructions;
- encourage children between four and eight years old to approach and use cybernetic objects in play/explore contexts;
- investigate how the structure of learning processes develops when children encounter and experiment directly with multimedia approaches while working with cybernetic constructions;
- explore the possibilities of making the design of the software and the play material suited for young children.

From a functional point of view, the investigation of the relationship between young children and cybernetics could be broken down into the areas of play material, software and learning background/practice. This structure is reflected in the composition of the CAB consortium. While the LEGO Group, Denmark, focused on the play materials, the Istituto per le Tecnologie Didattiche (ITD), Italy, investigated and designed programming environments including support material [Askildsen et al, 2001b]. The Högskolan för Lärarutbildning och Kommunikation (HLK) at Jönköping, Sweden, and the Comune di Reggio Emilia (CRE), Italy, anchored the research efforts

[1] This is a revised version (March 2009) of the chapter published in 2004. The main difference is the new set of figures coming from the Reggio Emilia field testing [Barchi et al. 2001]
[2] With the contribution of Edith Ackermann, e-mail: edith@media.mit.edu

to classroom evaluation of the material. They provided cognitive and educational interpretations of observed results that led to further improvements to the materials and methodology [CRE, 2001; Gustafsson & Lindh, 2001].

Constructionism stresses the role that concrete objects play in the complex process of knowledge construction [Papert, 1980, 1993; Harel & Papert, 1991; Turkle & Papert, 1992]. The traditional operational and experiential approach of *learning by doing* [Dewey, 1910] is therefore re-interpreted unfolding the potential that the construction of objects has for learning: knowledge emerges as a result of an active engagement with the world through the creation and manipulation of artefacts (tangible or not), e.g. sand castles, computer programs, LEGO constructions etc., that have relevant personal meaning and, above all, are objects to think with. Similarly important is the negotiation of meaning in the social world, considered as a crucial component of children's cognitive development: learning and intelligence emerge in the social groups where individuals interact and collaborate to build a common body of shared knowledge. Children act together with pairs and with older subjects, who can provide support and motivation in coping with new cognitive tasks [Resnick, 1996]. In the constructionist framework computer programming has always played a special role, as it is considered a tool to "think about thinking". But what is the meaning of "programming"? There is no simple answer to this question. Programming can be many things to many people, and not everyone agrees on its potential to foster human learning and development. To some, programming is about writing code, while to others it is a way of thinking [Papert, 1980]. Some perceive its potential in helping children sharpen their thinking, or become better "scientists" [Resnick et al, 2000], while others stress its ability to foster human creativity [Edwards et al, 1998], and enhance self-expression [Maeda, 2000].

Programming is a Pygmalion of sorts: it becomes what you want it to be. To a scientist, for example, it turns into a tool to master the world (through simulation). To a poet, it serves to create fiction, or build a virtual world. Designers use it as a dynamic modelling tool. Literary critics see it as a new form of literacy. And for the developmental psychologist the "hidden" value of programming lays primarily, and not surprisingly, in its ability to promote the exploration, expression, and reflection of children's "budding" selves-in-relation [Ackermann, 2000]. Thanks to their programmability and inspectionability, *robots* and *programmable bricks* are among digital toys that today offer specially interesting features. They impact on the way of thinking to life, as they position themselves on the boundary between the animate and the inanimate [Turkle, 1995]. Toys that actually behave elicit novel ways of exploring relational issues, like agency and identity. Their hybrid nature makes it possible to play out the fine line between objectifying minds and animating things, and come to grips with the hardships that identity formation involves [Ackermann, 2000].

Cybernetic construction kits, conjugating the physical building of artefacts with their programming, can foster the development of new ways of thinking [Resnick et al, 1996] that encourage new reflections on the relationship between life and technology [Martin et al, 2000], between science and its experimental toolset [Resnick et al, 2000], between robot design and values and identity [Bers & Urrea, 2000]. As constructionism supporters argue, thanks to these objects many concepts that are usually considered prerogative of adults, who can deal with symbolic and abstract knowledge, are made accessible and comprehensible for children as well [Resnick et al, 1998; Resnick, 1998].

The research methodology

Cybernetic construction kits are nowadays absent from the practices and culture of young children. Papert himself suggests that the programmable brick can be used by preschool children [Papert, 2000], but the available material is currently not adequate for this age. We had to choose whether to redesign the material first (designing for children) or to involve the children from the beginning in our development effort (designing with children). We chose the latter option, beginning with the

LEGO® MINDSTORMS™ Robotic Invention System that was commercialised at the same time the project started. This allowed us to provide the schools involved in the project with stable and reliable material; however, we were aware that such material would not be suitable for autonomous use by children.
This decision posed us the following questions: what happens when a product that has been designed for 12-year-olds becomes an educational object for 5-year-olds? How could such a choice be legitimated at the pedagogical level? What has to be done to remove the factors that limit the accessibility to technology?

For the whole experimental phase, which lasted two years, we chose to include the children's activities with the programmable brick into the context and practice of everyday work. We followed the Reggio approach to early-childhood education: special attention is paid to the development of meaningful learning contexts that facilitates the children's work; activities are situated within projects that address a broad range of issues over an extended period of time; adults try to avoid acting invasively and yet know how to listen and document what goes on, striving to sustain the children's motivation [Malaguzzi, 1998].

The rationale of the experimental activities is therefore twofold: on the one hand, we consider the cognitive potential of toys as deeply anchored to the usage context and culture that give them meaning. On the other hand, learning is seen as a social and contextualized process in which children receive support and scaffolding from adults in their exploration activities. The role of the teacher, as a mediator of knowledge and skills, was crucial for coping with the shortcomings of the available technology for this age group. The teachers also engaged in the documentation of children's activities as an integral part of their everyday work [Rinaldi, 1998]. This documentation, produced in a variety of formats (texts, images, video, etc.), was made available to the entire CAB community (Fig. 1).

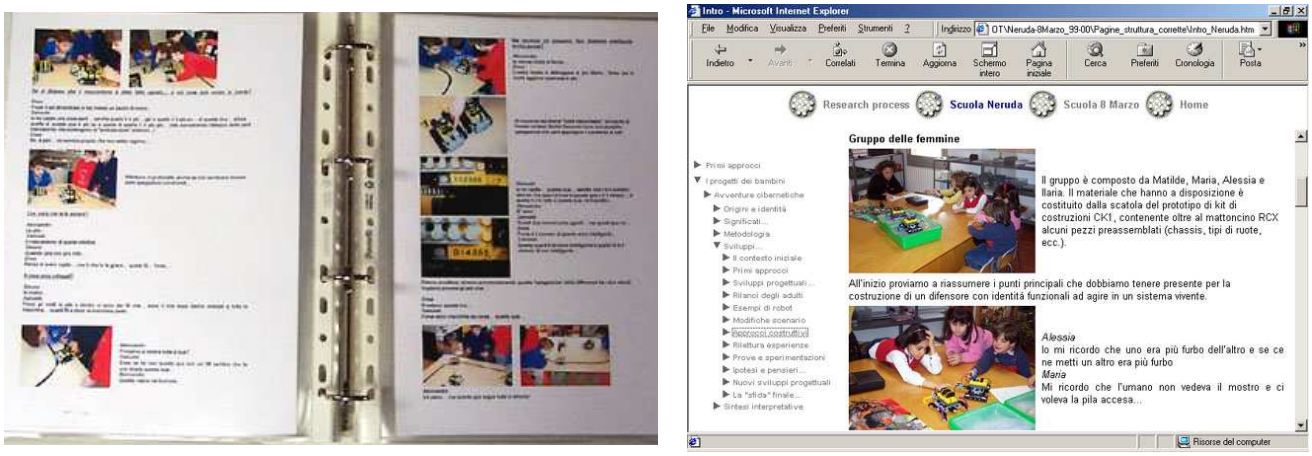
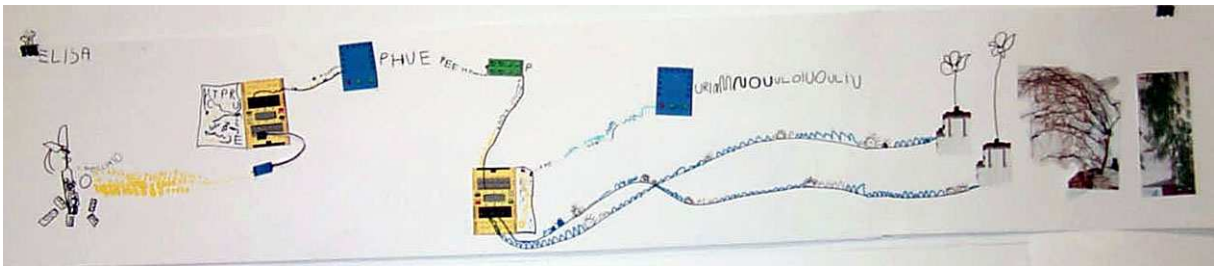
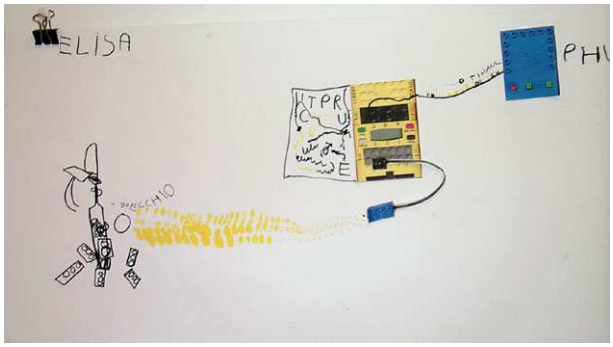


Figure 1: documentation containing the words of the children and teachers, drawings of the children's hypotheses and photographs both on text and on the project web site

The reflections and interpretations adults made on the issues that emerged from the work with the children, made it possible to consider the "theories", albeit provisional, the children had developed during their experiments with LEGO MindStorms and with the subsequent versions of the construction kit (Fig. 2). Our aim was to collect descriptive and narrative accounts of what happened in the encounters between the children and the construction kit, and note how the situation developed. We saw the documentation as a process of reflection and elaboration, that enabled us to elicit from the children the requirements for the evolution of the LEGO kit. The LEGO kit was incrementally modified according to these requirements through an iterative process.





"When the sunlight goes in the sensor, I think it shrinks to get in…inside the sensor these little yellow balls get smaller and then the light becomes a voice because the wire that connects the sensor to the RCX works really hard."

"When the sunlight goes in the RCX and the tape recorder, it gets specialised; when it becomes the voice there's a little bit of the sunlight that's left."

Figure 2: A fragment of a child theory on the inner working of the first version of the "Giving another life" project (see the description of the project in the case studies section).

Evolution of the play material

A typical LEGO construction contains some parts that are essential for its stability, some that provide extra abilities, and some that are purely decorative and meant to give the construction a certain character. To provide the best possible set of basic components in a construction kit a balance needs to be stricken in the granularity of the components, maximizing freedom in what can be constructed and at the same time minimizing the intrinsic complexity. This is the trade-off between specific/powerful and generic/open-ended components. In the field-testing most objects had to be designed and constructed by the teachers, thereby diminishing the meaning for the children of the constructions. The problems encountered can be clustered into three categories: 1) the complexity of the LEGO Technic mechanical subsystem; 2) the opaque design of sensors and actuators; 3) the bias of the LEGO MindStorms kit toward mobile robots.

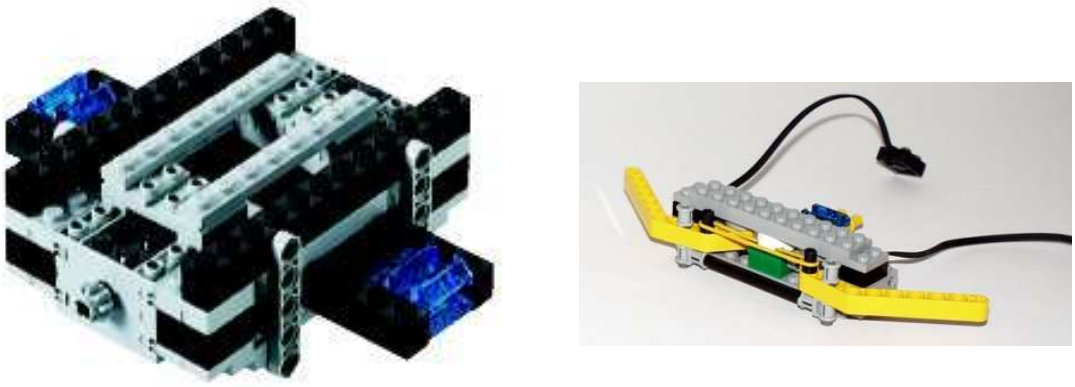


Figure 3: Subassemblies - a vehicle chassis and a pluggable module with two contact sensors.

The mechanical subsystem

LEGO Technic is a very flexible and powerful mechanical kit that enables experts to build complex robotic constructions. A rich body of literature is devoted to support adult users in mastering the complexity of this system [Martin, 1995; Ferrari & Ferrari, 2001]

Our approach was to identify and build pre-assembled mechanical modules aimed at improving children's autonomy in using the kit. Such subassemblies (Fig. 3) include a standard, ready-to-use vehicle chassis; a locomotion system with caterpillars; a structure to host a pair of contact sensors, to be used on a variety of vehicles; a conveyor belt; a motorized rotating cradle hosting the programmable brick.

This choice allowed children to include complex mechanics in their constructions, at the cost of restricting the creative exploration of the material (see Fig. 4).

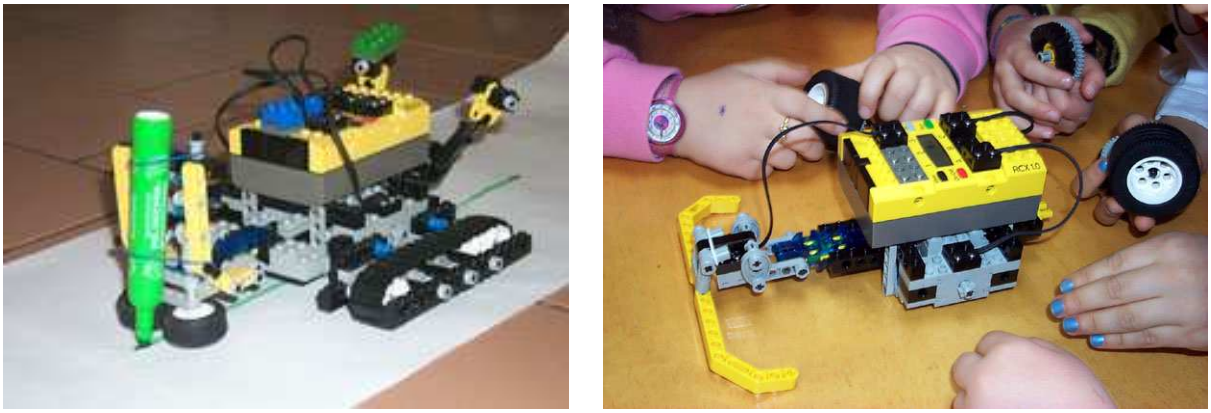


Figure 4: vehicles assembled by the children for the "Giving another life" and "Cybernetic adventures" projects.

A better mechanic system for children should not start at the level of gears and axles; rather, it should abstract the different types of motions and make them combinable. The children could be provided with small modules that embed mechanical gear designs. This would allow them to investigate and apply a range of mechanisms, e.g. produce radial lever movements or translate high-torque rotation into faster low-torque, without having to build them from scratch.

Types of constructions

A construction kit, depending on the components it offers, inevitably favours some types of activity and hinders other types, thus imposing an implicit bias on the typologies of allowed constructions. The LEGO MindStorms kit is designed to favour the construction of vehicles, mobile robots that interact with the environment. In order to encourage the development of other usage scenarios further construction types were identified: "kinetic sculptures" (the robot has moving mechanical parts, although not necessarily does it move around); "animated constructions" (the robot features reactive behaviour using sound, light, messages etc.); "cybernetic soft toys" (similar to Furbies, but easy to inspect and modify). Not all these proposals earned the children's and teachers' favour; some required a review and revision cycle or even abandonment. This was true of cybernetic soft toys (Fig. 5), which raised more perplexity rather than acceptance among teachers, who were afraid to offer a pedagogically poor proposal.

Figure 5: A soft toy that will play back a recorded message when it hears something and play different sounds when somebody bends its antennas.

The use of typologies other than vehicles reduces the need of complex mechanical constructions and allows for the exploration of reactive behaviours that fit, for instance, in story telling, where we have observed children combine a variety of building material (LEGO, paper, fabric, clay). In these scenarios the reduced mechanical complexity lets children express their creativity and address the behaviour definition early in the construction process.

The design of active components: sensors and actuators

From the discussion about construction types the need emerged to define additional sensors and actuators components. For example, children love to add recorded voice and sound to their computer-made artefacts. Accordingly, we designed and prototyped a tiny digital recorder that could "give voice" to the robots.

Other added components include: a light chain, a bend sensor, a sound sensor and an infrared transmitter. Having a variety of components each with its peculiar properties raises the issue of its readability: during the field-testing teachers often reported the difficulty to explore the features of sensors and actuators, as the components did not communicate their functionality to the children. Following each phase of the field-testing, discussions regarding proposals for more evocative and communicative materials led to suggestions for the improvement of their design. It would be desirable for sensors to be active: i.e., a sound sensor (microphone) could glow with differing intensity depending on the volume of sound it detects. The idea is to endow sensors with LEDs arranged in a line acting as a visual gauge (fig. 6). This solution would provide the children with a concrete reference level: as the sound rises in loudness they would see an increased glow of the LEDs. This would be a step forward from the abstract numerical representation of sensor values that is currently displayed on the programmable brick.

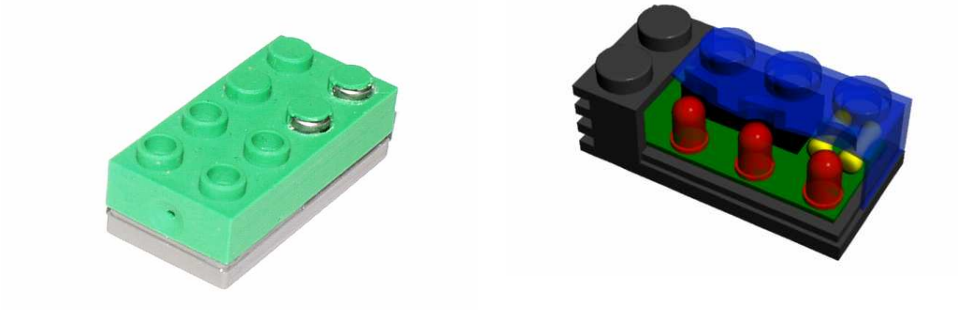


Figure 6: The sound sensor prototype and a sketch of the proposed LED gauge.

Looking at design-functionality, we opted for an operational interface that was independent of the programmable brick. Our recorder prototype (fig. 7) allows for the recording of two short messages; a button for recording and another two for playing the sounds are placed on the recorder itself.

Figure 7: Listening to a message recorded on our prototype.

The experiments with the children, however, showed that they wanted both more and longer messages. One might privilege the interface transparency by associating just one component to each message: if longer messages are needed, one might always sequentially connect several recorders [Ananny, 2001]. Such an approach allows for a story to be assembled by manipulating the order the various recorders are linked together. This enables complex interactions among sensors and sound sequences: for instance children could build a "sound wall" that emits different sequences if the temperature rises above a certain level, if someone touches a hot-spot or if two bricks exchange a message. If the same features were to be available for vehicles, practical constraints would emerge on the weight and size of components, thus inducing the design of a single, smaller recorder, at the price of some opacity of its interface.

Building the robot's behaviour

Putting in the children's hands the tools to program the behaviour of a cybernetic construction is often questioned since programming is not easy and many believe it should be left to specialists. If this were true, we should abandon the idea of a construction kit, and limit ourselves to the design of cybernetic toys whose behaviour, although highly interactive, could not be modified by children. On the contrary, we believe that a kind of programming is indeed possible also for non-programmers, provided that it is supported by problem-specific tools (environment, language etc.).

The development of tools that make programming for specific problem solving accessible to people who are not particularly experienced – nor interested – in computer science is a thriving research area. In particular the work done by Bonnie Nardi (1993) shows that users who are expert in a specific domain – or who are interested in practicing it – can learn and manage formal languages relevant to that domain. Significant examples are tools such as spreadsheets or statistical analysis packages, which provide the user with a programming language useful to extend the system functionality. Tools of this type allow for the growth of a user population who, at various skill levels, benefit from the available programming functions.

To apply similar considerations to the children world we have to assume that they can manage the level of complexity implicit in the control of robot behaviour. So, it makes sense to set up a programming environment for children. In the CAB framework we have verified that children can indeed deal with robotic constructions, provided that the context is well structured.

What is the conceptual model that best supports the definition of robot behaviour? Let us start with a sample task: make a vehicle turn around a square-based obstacle. A beginner would probably think in terms of driving a car via a remote control, devising a solution in the imperative programming style of the Logo turtle: "Go straight on along one side of the obstacle", "Turn left 90

degrees" and "Repeat these two instructions for the other three sides". However, robots are endowed with sensors that "perceive" the surrounding environment and allow them to react accordingly: programming a robot thus entails handling a number of sensors at the same time. The imperative programming style that is adequate for a broad range of situations (scientific computation, accounting, etc.) is inadequate for robot programming [Resnick, 1991; Papert, 1993].

If we add a touch sensor to our vehicle, for instance, we can address the problem in a radically different way: we can simulate the behaviour of a person who, in the darkness, has to circumnavigate an obstacle following its contour by hand. The program is built by relating the data coming from the sensors with the commands to the motors. A robot which "touches" the wall as it goes on is hard to build with LEGO pieces; it is easier to make the vehicle oscillate bouncing in a zigzag fashion: the robot moves away from the wall when the sensor touches it and re-approaches it when the sensor loses the contact. In other words: "if the sensor touches, turn on the motor on its side and turn off the one on the other side; if the sensor does not touch, turn its motor off and on the other one".

A solution of this kind offers a number of advantages against the imperative approach: the behaviour emerges from the interaction between the robot and the obstacle, independently from the shape and size of the latter. Besides, this approach allows, with minimal morphological changes, for the solution of other problems. For instance, should we want the robot to follow a line on the floor, it would be enough to replace the touch sensor with a light sensor while keeping the same program structure: the robot will zigzag along the line contour. In all these cases, rather than representing the map of the world in the program, it is the "playing field" that works as the map of itself [Brooks, 1991]. In cases where the environment properties (the shape and size of the obstacle, the geometry of the line, etc.) are not known in advance the robot has to exhibit a level of adaptivity that can only be obtained through the use of sensors.

Domain orientation

We chose to represent the behaviour of a robot via a set of rules. A rule associates a condition (a test on the state of a sensor) to a sequence of actions (commands for the actuators), e.g. *if* the light sensor finds a high value of brightness *then* turn the motor on. The ease of using this rule system depends on the availability of conditions and actions that encapsulate the hardware details and are directly operational. The usability of conditions and actions in turn relies on assumptions on the type of construction. For instance, a vehicle with two motors can move forward and back, rotate left and right. Thus, turtle-like commands are provided for vehicles. For each construction type of which the programming environment is aware, a set of primitives is defined, that specialize the available functions to the problem at hand.

The overall behaviour of a construction emerges from the composition of simple behaviours that act concurrently. For example, a vehicle with touch sensors that moves in reaction to obstacles can be controlled by two behaviours. The first instructs the vehicle to move forward; when it touches an obstacle a second behaviour tells the robot to move back and turn in the direction opposite to the side of the collision.

Behaviours can be constructed and tested incrementally. If two or more behaviours are in charge of the same actuator, a priority mechanism decides which one is in control. In the last example, the behaviour that manages bumps has a higher priority than the one that moves the vehicle forward.

The programming environment presents a gallery of the existing projects and the possibility to start a new one. A project is composed of one or more constructions, and encompasses both the programs and a multimedia documentation of the children's work. The environment allows defining various types of construction to support the specialization of programming components (behaviours, conditions, actions). A construction type makes certain assumptions on its mechanical components. A vehicle has a chassis equipped with two motors and is capable of moving and steering. When

equipped with suitable sensors a vehicle can execute a range of built-in behaviours like “follow a line”, “search for light”, “follow a wall”, etc.

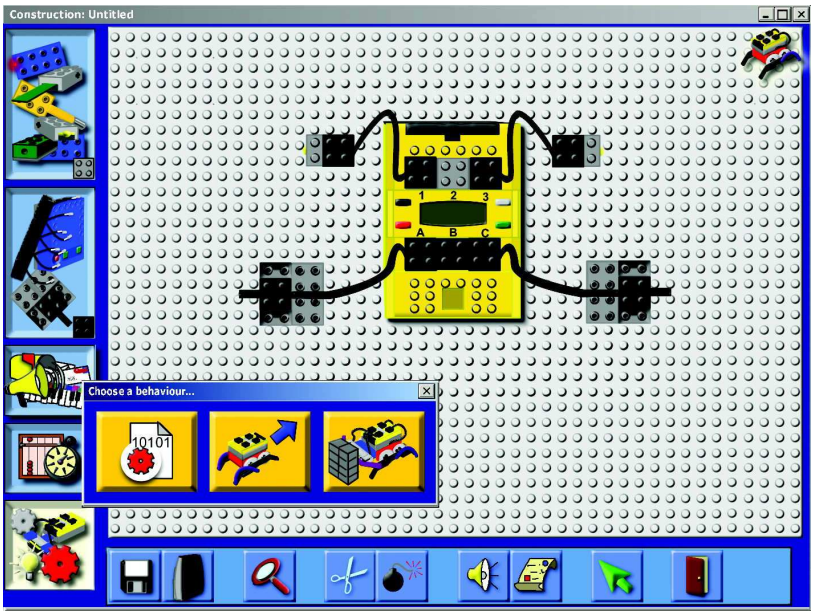


Figure 8: A schematic representation of a vehicle with two touch sensors. The behaviour menu is selected, allowing the choice among the available behaviours that match the construction input and output set, or the definition of a new one.

The environment is capable of suggesting the possible available behaviours depending on the sensors used in a given construction (Fig. 8). When defining a new behaviour, only the conditions and actions that match the current hardware configuration are presented (Fig. 9). Thanks to this specialization mechanism, it is possible to let the environment evolve according to specific needs of a project.

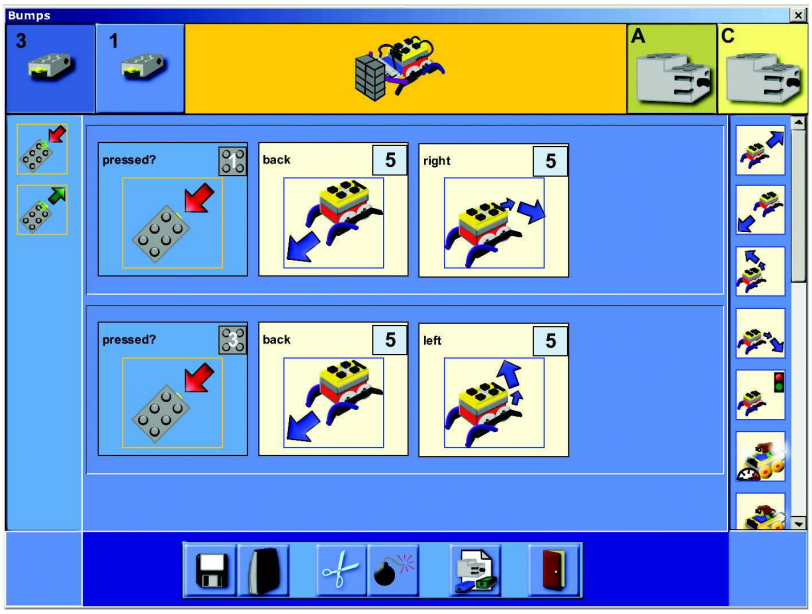


Figure 9: The two rules that define the “bump” behaviour. Note that only the conditions and actions associated with the selected input and output devices are shown.

Tangible programming

A key challenge has been to empower children to construct programs their own programs out of physical components as they do with LEGO bricks. Tangible programming [Suzuki & Kato, 1995] is an active field of research where many projects aim at young children [McNerney, 1999; Wyeth & Wyeth, 2001; Montemayor et al, 2002]. The benefits of a tangible interface are twofold:

- it enables a small group of children to build programs together – unlike when using a screen-based programming environment because only one child at time can be in control of the mouse or keyboard;
- children can take advantage of the dexterity of their hands - in a graphical user interface objects are manipulated via a mouse or other suitable pointing devices.

Given the age of our target group even small advantages are of value. A tangible version of the CAB programming environment, where behaviours, conditions and actions are themselves physical manipulative components of the kit, could realize the vision of mixing Atoms and Bits in a concrete and child friendly way. Fig. 4 provides an illustration of how this could be done using existing technology. The tiles contain electronics components with their ID and once connected are capable of communicating their topology [Gorbet et al, 1998]. A tiles dock reads the tiles configuration, generates a program and downloads it into the programmable brick. The dock also communicates with a computer to connect the tangible interface with the one on the screen. The kit can be extended redefining the meaning of a tile (Fig. 10).

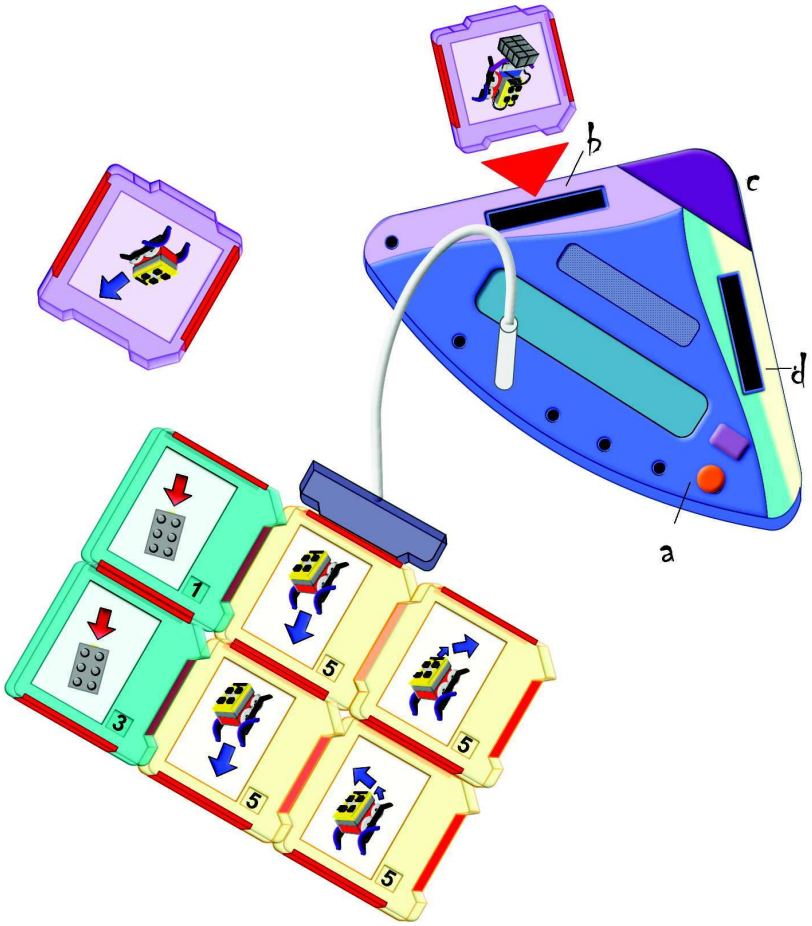


Figure 10: A tangible programming version of the bump behaviour. The tiles' dock is annotated with labels indicating its features: a) conditions and actions tiles can be connected to this side of the dock to define a rule; b) the current rule can be used to define a new behaviour inserted in this slot, behaviours can be connected directly to this side; c) a communication device; d) an additional slot to redefine the meaning of a tile.

The computer would still offer distinct advantages, for example in storing and documenting previous work or exchanging behaviours at a distance, but would not be required to start a project.

Metacognitive and social support

We feel that software which can retain a memory of the product and process of the children's programming by means of the visibility of trials, tests, errors, and variations, can offer opportunities for learning. And not only for the children who created the programme, but also for the other children involved. Such software can thus become, through metacognitive processes, knowledge that can be re-applied and re-used. [CRE, 2000a]

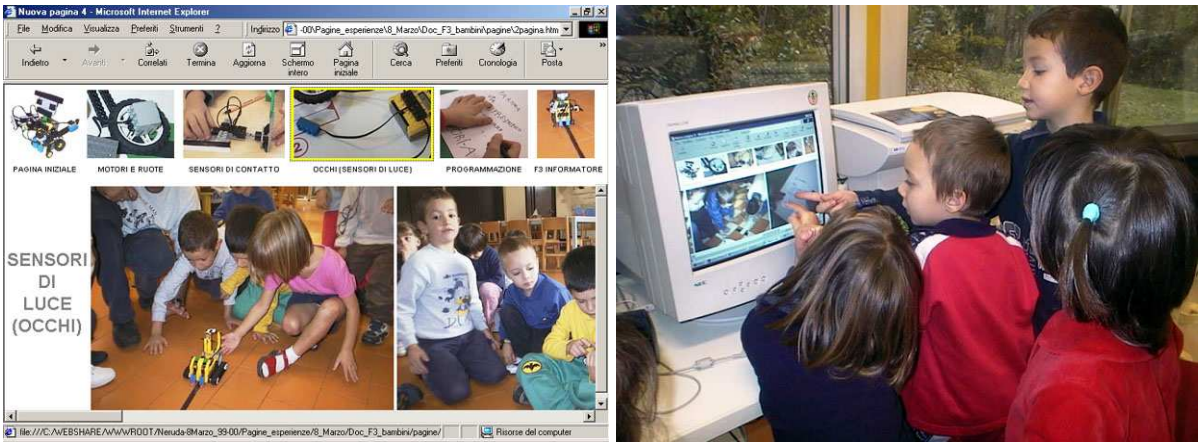


Figure11: A 'digital photo album' easy and intuitive for the children to consult and which encouraged them to revisit and reinterpret their learning experiences.

Formalizing the behaviour of a robot by means of rules has important cognitive and metacognitive implications. On the one hand, the rule reifies the cause-effect relationship and supplies the children with an important linguistic instrument to talk about and reflect on reactive behaviours ("If the temperature increases then the robot turns on the fan"). On the other hand, the immediacy of interpretation and the readability of the rules allow the children to revisit their problem solving approaches ("… then we added this rule to teach the robot to turn on the fan when it's hot …"). This is especially useful when their programs do not produce the expected results. Typically, children prefigure a wide and articulate context, with many actors involved, where their fantasies shape up and evolve. Therefore, a project needs ways of supporting the memory of the work done, both for documentation purposes and as a representation of the history of the programming and building choices (Fig. 11 & 12).

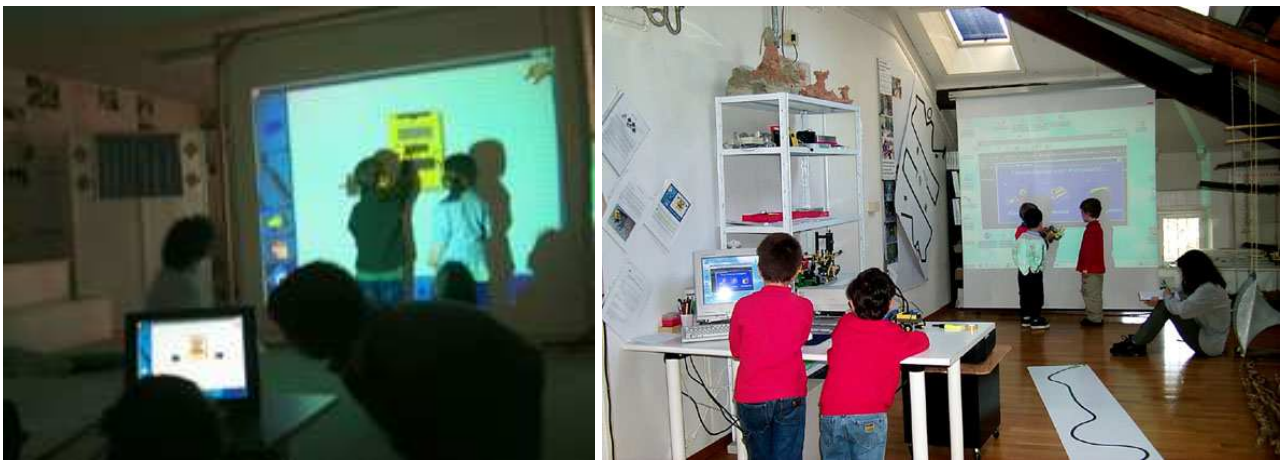


Figure12: Use of the video projector to construct "zones of dialogue" among the children, zones that are difficult to create around the small monitors of the computer.

Moreover, the environment assumes a social context of use that is articulated on three roles: children, teachers, experts.

- The *children* collaborate among themselves and with teachers in all phases of the project, from the identification of the problem to the invention of a solution. They discuss and compare possible alternatives, inspect examples and modify them to suite their needs; they explore the potential and the limits of the technology; they are engaged in an iterative process of socially shared construction in which the hypotheses that emerge are subject to the judgment of the group and to empirical verification.

- The *teachers* mediate between children and technology to smooth the interaction and support the children creativity and motivation. Some of the options of the programming environment enable the teacher to configure it for specific project requirements. They can change the icons and names of the objects (actions, conditions, behaviours), and tune the parameters of actions (e.g., the scale factors of commands such as: forward; wait, etc.) and conditions (the thresholds of sensors).

- The *experts* can extend the environment adding the definition of new construction types, actions and conditions.

Case studies

The project field-testing covered two school years and involved three infant schools of the Reggio Emilia Municipality and three elementary schools in Sweden. For an in-depth description we refer to the final reports of CAB's educational partners [CRE 2001; Gustafsson and Lindh, 2001]. Here, our objective is to focus on how the research in the classroom has influenced the design of a programming environment usable by five-year-old children. So, we limit our case studies only to projects made in the Reggio infant schools. The following three projects show that:

- there are no cognitive obstacles to children programming of cybernetic creatures;
- in the presence of a well defined context and specialised tools, children are capable of programming a robot;
- to support children's projects the programming environment must provide powerful, specific primitives;
- the proposed environment was usable thanks to its appeal, the appropriate nature of its granularity, inspectability and suitability for being an object of discussion and reflection.

RoboSports

A group of children of the "La Villetta" infant school has experimented with the RoboSports kit. This system was especially developed for the visitors of LEGOLAND Parks and allows them to quickly participate into a robot contest. This kit comprises a playing field for two teams to compete in making a vehicle that carries as many balls as possible in a hole. The field is a table with two tracks each composed of one black line and one back-lighted hole. The mechanical components are specialized, thus allowing the construction of a limited set of vehicles capable of transporting and pushing the balls into the hole. The software environment supplies primitives such as: a "follow-the-line" behaviour, a condition that can be used to stop it when the light sensor detects a back-light, translation and rotation commands to push the balls in the hole. The kit comes with video tutorials to help the users build and program the vehicles. At "La Villetta" school parents and teachers have built the playing field and the children have set up and programmed their vehicles for the contest (Fig 13).

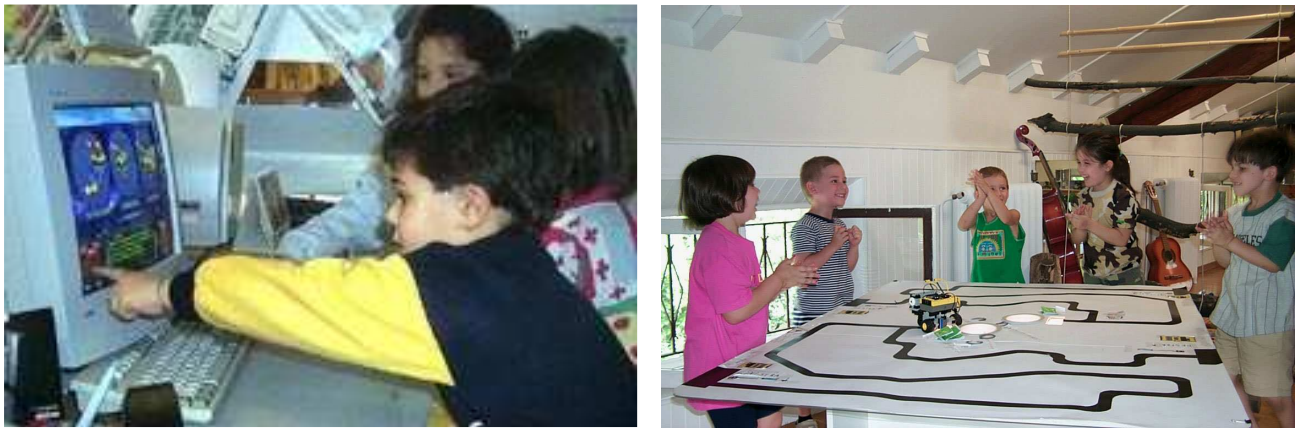


Figure 13: RoboSports software and playing field.

This experience has demonstrated that the children succeed in using the features of the programming environment to solve the problem, because the specialization of the components simplified the construction of a vehicle suited to the task, and the visual programming environment that supplies only a limited, but powerful, set of primitives enabled the children to compose the program autonomously. Moreover, when engaged in group discussions to overcome programming errors, they spoke in terms of the icons of the programming language to annotate the playing field (Fig. 14), as a symbolic representation of the program execution, when discussing the effects of the instructions given to their robots [CRE, 2000b].

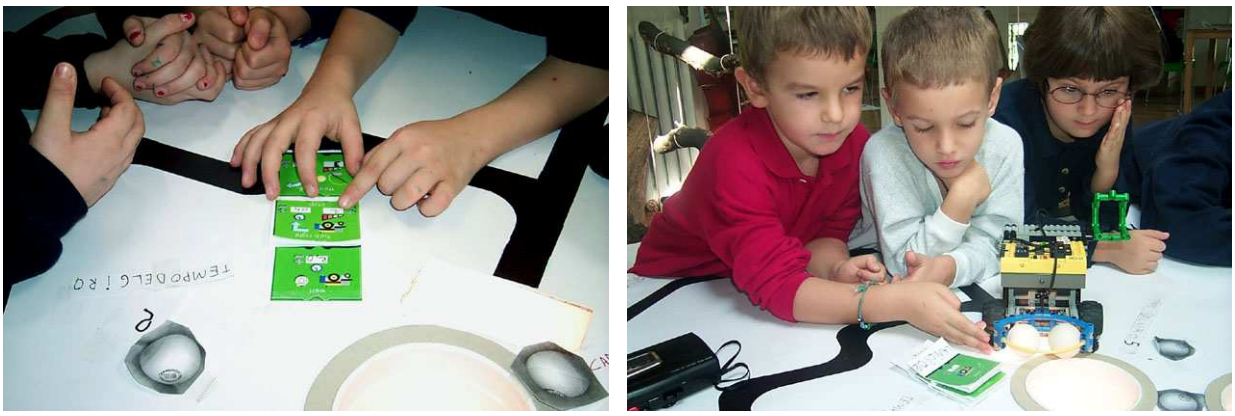


Figure 14: Children annotating the playing field.

RoboSports is a good example of the potentialities of a context-oriented system. Its limit comes from being too specialized: the hardware and software components can only be used for this contest, or contests of this type, thus limiting children's creativity.

Cybernetic adventures

The widespread interest shown in monsters, by a class of children in the Neruda school over several years provoked the idea of constructing a scenario (Fig. 15) where the different identities of single cybernetic subjects and the characteristics of the context would allow the creation of a 'possible life'. This life would develop and evolve according to the frequency and quality of relations between the 'actors' (monsters or defenders of the city) [Barchi et al, 2001].

Figure 15: Monster and defender scenario planning.

The monsters attacked a city; the inhabitants constructed walls and traps to defend themselves and organized a team of defenders to hold back the monsters (Fig. 16). The monsters and the defenders have been constructed with defined behaviours that reflected the dynamics of a battle whose evolution and outcomes are unpredictable. Each monster was equipped with two touch sensors used to avoid obstacles and a light sensor pointing to the floor, to stop the vehicle if it enters a coloured zone. The monsters had a light mounted on the back that makes them recognizable by the defenders. The defenders had a light sensor allowing them to move in the direction of the attack.

Figure 16: First version of the scenario.

This first definition of the behaviours was such that, after colliding, the monsters and the defenders wandered over the playing field without a clear objective. A monster accidentally ended in the trap or succeeded to enter in the city; the defenders rambled without a clear strategy of how to block the monsters that they encountered. The children recognized these limits and proposed

alternatives, but the teachers could not implement the proposed mechanisms of attraction between the defenders and the monsters, and between the monsters and the city. So the experts were called in. They proposed two modifications to the project: to introduce tracks of a different colour showing to the monster the direction to arrive at the city doors, and a modification of the program of the defenders, activating a mechanism for seeking the monster (i.e., turn around and detect the direction of light). These proposals were discussed with the children, who modified the scenario (Fig. 17) and the robots so as to obtain the desired behaviours.

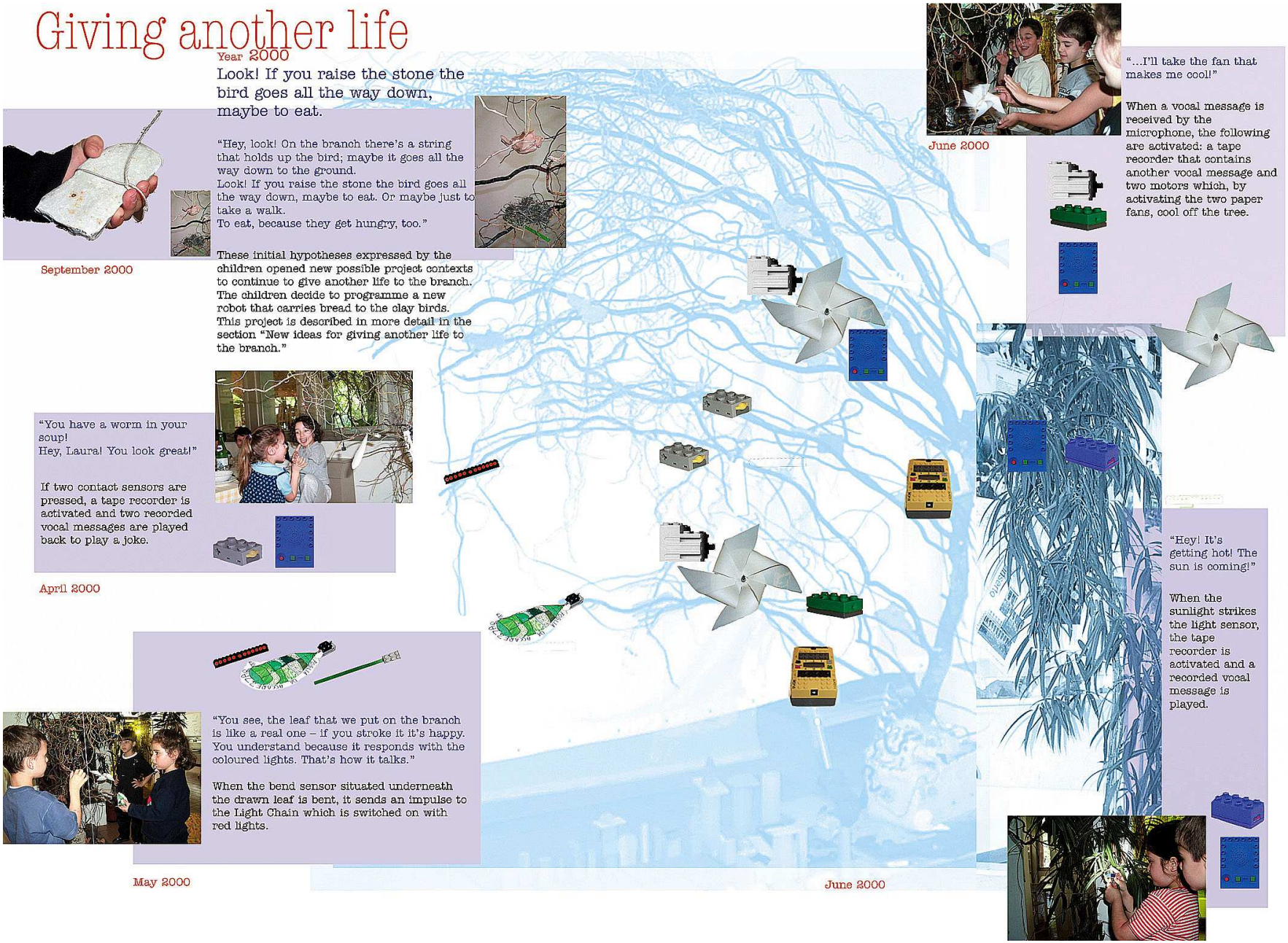
Figure 17: Modified scenario.

This project work has covered a long period over several phases: designing and realising the scenario, programming and experimentation, modifying the behaviours. By participating as design experts, the authors of this chapter ascertained the following:

- it is possible to capture the complexity of projects of this type in the proposed model;
- complex behaviours such as those exemplified by this project cannot be develop by children on their own, but they *can* become part of a repertoire of specialized components that children can evaluate and apply.

This style of interaction with cybernetic objects can be defined as “playing the psychologist” [Ackermann, 1991]. "Playing the psychologist" and "Playing the engineer" constitute the two ends of the spectrum of possible roles that children can adopt when they interact with a cybernetic construction kit. In the former, children observe and ask themselves questions on the nature of the object at hand (on its intentions, "intelligence", etc.), so as to understand its intimate nature; in the latter the construction and modification activities of the objects and their behaviours prevail. The children keep oscillating from one to another, and the psychologist and engineer components are calibrated and arranged as various degrees of breakdown occur.

Figure 18: First version of the "Giving another life" project.



Giving another life

This project originated in the 1999/2000 school year at the Villetta infant school (fig. 18):

> "… from a group of five and six-year-old children who wished to help a large branch that had broken off from a tree due to a heavy snowfall. The children were well aware that they could create 'another kind of life' for the plant, which had now been sheltered in the school piazza. The children placed the digital tools and materials in relation with the sensors and actuators which they thought were most suited to allow the different subjects to communicate, and with other languages and materials (paper, wire, clay, structures built with recycled materials, etc.) that are a common feature of school life. The children's narration was the bond that held together the different levels at which the research was being conducted, constructing meanings, even provisional ones, and identifying new questions to be investigated" [CRE, 2000b].

Figure 19: Children exploring the branch tree project developed the previous school year.

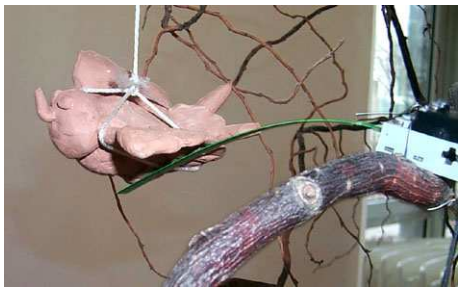
The following school year (2000/2001), another group of five and six-year-old children (Fig. 19) extended the project by adding a dialogue between a bird on the tree and its robot friend. During winter food supplies are scarce. The bird asks for the help of robot baker-boy that will bring crumbs to the tree. Once there, the robot will notify its friend, who will come down for the bread (Fig. 20).

Because of growing-up in a school where cybernetic constructions were just one of the things on offer, the second group of children found it natural to construct their robots and program them with the visual environment previously described. Here is how they summarized their understanding at the end of the project:

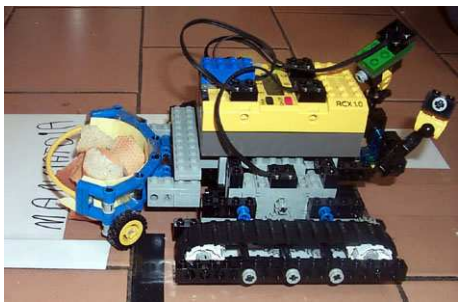
- "Now we're real robot programmers!"
- "It's true! This is a school of programmers! We can do all sorts of things!"
- "We discovered three secrets:

    1) two pieces of the measuring stick make one tile;
    2) if the bird touches the bend sensor, the recorder goes 'cheep, cheep';
    3) robots can talk to each other with the envelope and the letter box."

[CRE, 2001]



A bird touches the bend sensor...



...and immediately the “bread-carrier” robot starts off and goes to the tray.

Then the bird comes down from the nest and...

Figure20: The proposed project extension.

The children solved a number of sub-tasks generated by the evolution of their work. One was to determine the value for the “forward” command to let the robot move 6 tiles on the floor. To this end the children built a measuring stick, marking the distance travelled for various values of the parameter for the “forward” command. Experimenting with these values they discovered that “two pieces of the measuring stick make one tile” and “forward 12” was the solution to their problem (see Fig. 21). The magic number of “two pieces” did not come by accident. The programmable brick controls the amount of time the motors are on; how this time correlates to the distance travelled is a function of the actual vehicle details (tires, gears, weight, terrain, etc.). The software allows specifying a scale factor to tune the result, and the teachers customized it for this project.

The children built their robots in such a way that they could tell the story of the bird and his friend while the robots were playing it out. Synchronization points are essential to a good result. The robot should start moving when the bird calls it. An initial solution was to use a sound sensor to hear the voice of the bird. Unfortunately a sound sensor picks up noise and does not understand language. So, any noise was good enough to trigger the robot behaviour.

Figure21: Children building a measuring stick

By inspecting the software interface the children noticed the message icons, tried them out and discovered that robots “can talk to each other with the envelope and the letter box”. Exchanging messages provided a robust mechanism for synchronization, thus enabling the children to complete their project (see Fig. 22).

Figure22: The children discover that the RCX can exchange messages.

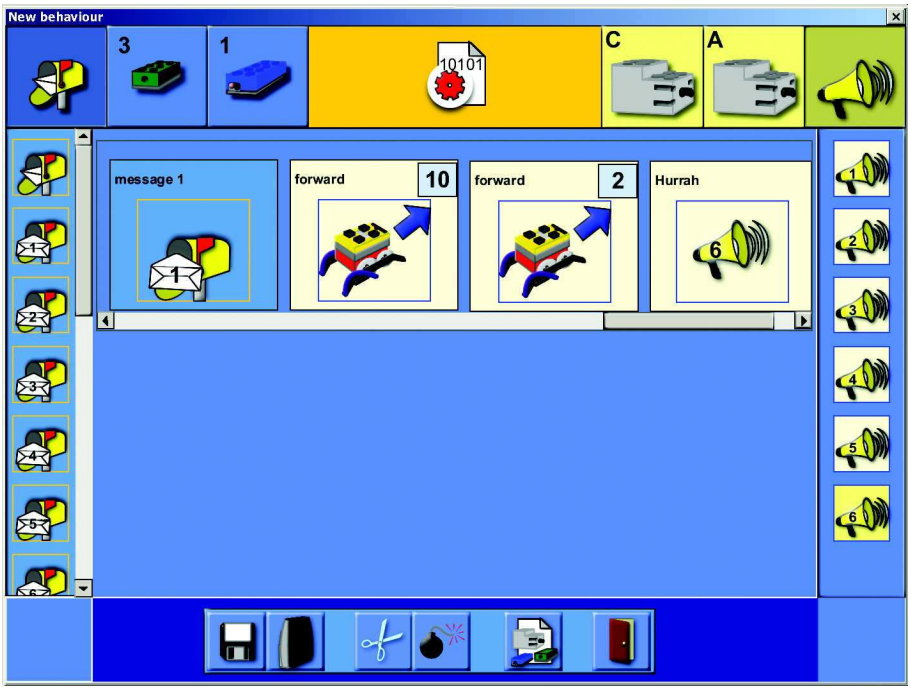


Figure23: The program for the bread carrier robot.

In the upper left corner of fig. 23 three sensors are shown: the message receiver (active), and a sound and a light sensor that are left over from previous design approaches. To move forward 12, the children used two commands (10 + 2) as the interface provides a slider (not shown in Fig. 23) that limits the parameter values to the range [0 .. 10].

The software interface does attempt to reveal what is possible with the programmable brick, by organizing the available features into boxes that contain components of the same type (Fig. 9). Furthermore, the structure of boxes reflects a concrete vs. virtual distinction: tangible sensors and actuators, built-in devices (i.e., sound and messages, which are not associated to a pluggable hardware component) and virtual devices (i.e., those implemented via software: timers, counters). This taxonomy is reflected into the operational structure of the interface that supports children while exploring, discovering and learning the available features.

Conclusions

We based our work on the notion of a competent child who can pursue difficult projects for extended periods of time in a supportive learning environment. We assumed that children would be interested in building their own animated constructions and programming their behaviours. The CAB project has shown that, in a supportive learning environment, children can and will design and build animated construction behaviours. We have proposed and prototyped a visual programming language that is not general purpose but rather strives for simplicity and power by incorporating knowledge of the construction types and the specificity of the project at hand.

A cybernetic construction kit endowed with a tangible programming interface and redesigned, to improve its mechanical system and the readability of active components, should enable children to explore the material freely and autonomously (without the need for external “experts”) while engaging in motivating projects.

Acknowledgments

This work has greatly benefited from continuous interaction with all the partners of the CAB project. We wish to thank all the adults and children that have accepted the challenge to use and experiment with the LEGO kit and our prototypes; the work described here could have not been done without their enthusiasm, creative input and support. The prototypes developed for the CAB project and cited in this paper do not imply any commitment from the LEGO company to incorporate them in future products.

References

Ackermann E. (1991), The "Agency" Model of Transactions: Toward an Understanding of Children's Theory of Control, in Harel I. & Papert S. (eds), *Constructionism*, Ablex Publisher, Norwood, NJ.

Ackermann. E. (2000), Relating to things that think: Animated toys, artificial creatures, avatars, in *I3 Magazine: The European Network for Intelligent Information Interfaces*, n. 8, July, pp. 2-5.

Ananny M. (2001), *Telling Tales: Supporting written literacy with computational toys*, Master Thesis, The MIT Media Laboratory, Cambridge, MA.

Askildsen T., Barchi P., Cagliari P., Chioccariello A., Giacopini E., Gustafsson B., Lindh J., Manca S., Rausch M., Sarti L. (2001a), *Construction kits made of Atoms and Bits. Research findings & perspectives,* CAB Del. 25

Askildsen T., Chioccariello A., Manca S., Munch G., Rausch M., Sarti L. (2001b), *A Cybernetic Construction Kit for Young Children,* CAB Del. 22

Barchi P., Cagliari P., Giacopini E. (2001), Encounters between children and robotics, in Askildsen T., Barchi P., Cagliari P., Chioccariello A., Giacopini E., Gustafsson B., Lindh J., Manca S., Rausch M., Sarti L. (2001), *Construction kits made of Atoms and Bits. Research findings & perspectives,* CAB Del. 25

Bers M. & Urrea C. (2000), Technological Prayers: Parents and Children Working with Robotics and Values, in Druin A. & Hendler J. (eds) *Robots for Kids: Exploring New Technologies for Learning Experiences*, Morgan Kaufman Academic Press, San Francisco.

Brooks R. A. (1991), Intelligence without representation, *Artificial Intelligence*, vol. 47, pp. 139–159.

CRE (2000a), *Approaches and suggestions for hardware and software development,* CAB Del. 12.

CRE (2000b), *Pedagogical results 2,* CAB Del. 17

CRE (2001), *Final pedagogical report*, CAB Del. 23

Dewey J. (1910), *How we think*, Health, Boston.

Edwards C., Gandini L., Forman G. (eds) (1998), *The Hundred Languages of Children: The Reggio Emilia Approach – Advanced Reflections*, 2nd Edition, Ablex Publishing Co., Norwood, NJ.

Ferrari M. & Ferrari F. (2001), *Building Robots With LEGO Mindstorms: The Ultimate Tool for Mindstorms Maniacs,* Syngress Media Inc., Rockland, MA.

Gorbet M., Orth M., Ishii H. (1998), Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography, in *Conference proceedings on Human factors in computing systems (CHI '98),* April 18-23, 1998, Los Angeles, USA, pp. 49-56.

Gustafsson B. & Lindh J. (2001), *Swedish field test. Final report***,** CAB Del. 24 http://web.archive.org/web/20031226200452/http://cab.itd.ge.cnr.it/cab/public/deliverables/del24.PDF

Harel I. & Papert S. (eds) (1991), *Constructionism*, Ablex Publishing, Norwood, NJ.

Maeda J. (2000), *Design by numbers*, The MIT Press, Cambridge, MA.

Malaguzzi L. (1998), History, Ideas, and Basic Philosophy: An Interview with Lella Gandini, in Edwards C., Gandini L., Forman G. (eds) *The Hundred Languages of Children: The Reggio Emilia Approach – Advanced Reflections*, 2nd Edition, Ablex Publishing Co., Norwood, NJ.

Martin F. (1995), The Art of LEGO Design, in *The Robotics Practitioner: The Journal for Robot Builders,* vol. 1, no. 2, Spring 1995.

Martin F., Mikhak B., Resnick M., Silverman B., Berg R. (2000), To Mindstorms and Beyond: Evolution of a Construction Kit for Magical Machines, in Druin A. & Hendler J. (eds) *Robots for Kids: Exploring New Technologies for Learning Experiences*, Morgan Kaufman Academic Press, San Francisco.

McNerney T. S. (1999), *Tangible Programming Bricks: An Approach to Making Programming Accessible to Everyone*, Master's thesis, MIT, Cambridge, MA.

Montemayor J., Druin A., Farber A., Simms S., Churaman W., D'Amour A. (2002), Physical programming: Designing tools for children to create physical interactive environments, in *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves (CHI 2002)*, ACM Press, pp. 299-306.

Nardi B. (1993), *A Small Matter of Programming: Perspectives on End User Computing*, The MIT Press, Cambridge, MA.

Papert S. (1980), *Mindstorms: Children, Computers, and Powerful Ideas*, Basic Books, New York.

Papert S. (1993), *The Children's Machine*: *Rethinking School in the Age of the Computer*, Basic Books, New York.

Papert S. (2000), What's the big Idea? Toward a pedagogy of idea power, in *IBM Systems Journal*, Volume 39, n. 3/4.

Resnick M. (1991), MultiLogo: A Study of Children and Concurrent Programming, in *Interactive Learning Environments*, vol. 1, no. 3, pp. 153-170.

Resnick M. (1996), Distributed Constructionism, in *Proceedings of the International Conference on the Learning Sciences*, July 1996, Evanston, USA.

Resnick M. (1998), Technologies for Lifelong Kindergarten, in *Educational Technology Research and Development*, vol. 46, n. 4.

Resnick, M., Berg, R., Eisenberg, M. (2000), Beyond Black Boxes: Bringing Transparency and Aesthetics Back to Scientific Investigation, in *Journal of the Learning Sciences*, vol. 9, no. 1, pp. 7-30.

Resnick M., Martin F., Berg R., Borovoy R., Colella V., Kramer K., Silverman B. (1998), Digital Manipulatives: New Toys to Think With, in *Conference proceedings on Human factors in computing systems (CHI '98)*, April 18-23, 1998, Los Angeles, USA, pp. 281-287.

Resnick M., Martin F., Sargent R., Silverman B. (1996), Programmable Bricks: Toys to think with, in *IBM Systems Journal*, vol. 35, n. 3/4.

Rinaldi C. (1998), Projected Curriculum Constructed Through Documentation – *Progettazione*: An Interview with Lella Gandini, in Edwards C., Gandini L., Forman G. (eds) *The Hundred Languages of Children: The Reggio Emilia Approach – Advanced Reflections*, 2nd Edition, Ablex Publishing Co., Norwood, NJ.

Suzuki H., Kato H. (1995), Interaction-Level Support for Collaborative Learning: AlgoBlock – An Open Programming Language, in *Proceedings of CSCL'95*, pp. 349-355.

Turkle S. (1995), *Life on the Screen: Identity in the Age of the Internet*, Simon and Schuster, New York.

Turkle S. & Papert S. (1992), Epistemological pluralism and the revaluation of the concrete, in *Journal of Mathematical Behaviour*, vol. 11, n. 1, pp. 3-33.

Wyeth P. & Wyeth G. (2001), Electronic Blocks: Tangible Programming Elements for Preschoolers, in *Proceedings of the Eighth IFIP TC13 Conference on Human-Computer Interaction (Interact 2001)*, Tokyo, Japan, July 9-13, 2001.