# E-slate: A 'black-and-white box' approach to component computing

**C. Kynigos and M. Koutlis**

**University of Athens and Computer Technology Institute**

**kynigos@cti.gr, koutlis@cti.gr**

**Paper presented at the Annual Meeting of the American Educational Research Association,**

**New Orleans, 2002.**

## Introduction

In this paper, we review three aspects of the experience of E-slate development which are central to its design rationale and have played a crucial role in sustaining interest and funding since 1994. These are, the ways in which the user community may be empowered by building educational software through authoring, the adoption of a variety of strategies to generate user communities and the conceptualization of the E-slate project as an on-going sequence of broadly defined multi-organizational projects fitting into the European Community policy to bring know-how from member states, academia and industry closer together. We suggest that argumentation for component oriented technologies has focused too heavily on its potential large-scale economics of production and that a broader view of this technology is needed to understand and support its development and widespread use. We discuss these three aspects of user empowerment, socially grounded theory of use and long-term development, support and service sustainability as crucial for the component movement to progress, but also as its strength in relation to standalone technologies for computational media.

## History and principles of design and development

### Geography meets mathematics and then expands

Two of the three original founders of the E-slate project were experts in G.I.S. systems looking to develop software for geography education (Koutlis and Hatzilakos, 1996). The types of functionality, the sheer volume of content and the types of representations for geographical information systems required quite heavy development and technically complex pieces of software. The idea to build a platform making it possible to expand to exploratory software to do mathematics with and to explore possible combinations of functionality enabling inter-subjective exploration and constructions brought about the rationale for a variety of components, user authoring by building component configurations, scriptability and reconstructibility (Kynigos et. al., 1997). Having (just) survived the open-doc fiasco, the putting of user authoring to the test was only made possible two-three years ago after the E-slate desktop environment became reasonably stable and the number of generic components large enough (around 40). So, along with cheaper software production and proliferation through the re-usability angle, the rationale behind the E-slate project has been explicitly to investigate the idea that building component configurations can be creative and interesting as a process for epistemological and educational reflection and for learning about component functionalities. This is based on a 'black and white box', or' principled deep structure access' approach highlighting the idea of component connectivity. E-slate is thus designed as a platform for authoring educational software across subject domains by means of creating configurations amongst a set of generic components. The platform is also designed to be able to host any piece of software developed by third parties with only minor changes to allow for connectivity with other components.

**Series of multi-organizational projects since 1995**

E-slate has been essentially developed by two collaborating groups making use of the contemporary era in the European Community policy for funding multi-organizational R&D projects to enhance collaboration between member states and between industry and academia. This provided the opportunity for relatively long standing funding allowing for large labor development of both the E-slate desktop and 40 generic components and the generation of user communities at different levels including research on different aspects of teaching, learning and authoring with e-slate software.

It is important to note the Greek context, since the relatively late development of wide-spread university based research has resulted in the European Union being by far the most frequent source of R&D funds (the equivalent to the NSF in Greece has been created only recently). Research and development priorities therefore are influenced by a widely diverse cultural web of societies which only in the last 30 years or so have made explicit strides for co-development.

E-Slate has been financed by a series of European Community based R&D projects channeled through the EEC direct or through the Greek General Secretariat for R&D. Typically, the priority has been for these projects to explicitly support collaboration between a variety of organizations (i.e. companies, computer science university departments, and end user organizations such as schools and education university departments) in some cases on an international level. E-Slate has also received funding from the Greek Ministry of Education sources within the framework of "Odysseia", the policy for integrating new technologies in schools. This is important because the funding from this source is not exclusively for research, but rather for the implementation of Educational Policy.

**Two collaborating teams as core design, development and user community support**

The E-Slate project has so far been centered at C.T.I. and funded by a series of projects involving collaborations with partners from academia, industry and schools. One partner, the Educational Technology Lab at the University of Athens has had continual contribution in design, microworld development, teacher education and use in schools from the outset. It is thus reasonable to view the E-Slate project as carried through by a wider two – legged team of developers and education experts. The technical development team, lead by M. Koutlis at CTI, is comprised of more than six full-time programmers working at C.T.I. on a project basis. The education team, lead by C. Kynigos, is comprised of Ph.D. and post – doc students, who typically spend around 4 years on the team. Through the years, the teams have gone through different phases of close collaboration and more remote co-operative work. Continual feedback and exchange of ideas and know how has been a feature of this work either by means of co-design of E-slate functionalities and components or by means of beta testing and drawing experience from the user communities in the field. Although the collaboration began with emphasis on the notion of integrated (rather than fragmented) development, the breakdown of work necessary to cope with its rapid expansion, resulted in the emergence of the following activities (Kynigos, in press):

- component architecture design and development (desktop)
- software design and development (components)
- secondary development of component configuration (authoring with E-slate)
- activity design and development (documented microworlds)
- collaboration with schools and school support
- teacher education
- research involving classroom and teacher seminar observation, tests and interviews

The technical group have mainly worked in the first two types of activity, but have also had significant contribution in the third, fourth and fifth. The education group have respectively had significant contribution to design in the first two activities as well as their work in the others. This engagement in a mutually accountable larger project and integration of work has resulted in a considerable degree of hybrid know-how developed in two distinct organizations (for a discussion of this issue, see Kynigos, in press).

This process has not been without problems due to both external circumstances and to the nature of the work itself. In the initial stages of the project, for instance, there were problems with the development platform “Open Doc”, which was discontinued in November 97. This resulted in great difficulties in the early E-Slate projects, YDEES and IMEL, and a large lag in time for redesign and development in java. It also resulted in lag of communication between the education team and the development team since the former was tired with the problems created by “Open Doc” and an anti-macintosh climate in Greece and cautious with believing that E-Slate would reappear working adequately on a Windows O.S. Moreover, the 5 schools where E-Slate tried the “Open Doc” version experienced horrendous problems and the consequent loss of confidence by teachers and students. The development team on the other hand worked extra hard and focused on re-developing the E-Slate platform in Java, feeling - justifiably so - that there was no time to “waste” on collaborations. In fact, the whole issue about scriptability, which is now one of the main features of E-Slate, was almost abandoned since there was no funding to re-do it in Java (The project now uses Daniel Azuma’s Java Turtle Tracks). Not unlike others, the project has thus had to deal with the continuing tension between using state of the art development platforms and producing software that can operate in real life situations from the early stages of its development. It has also had to make choices between involving users in lab or real schools situations and has mainly opted for the latter in order to enhance the understandings of what it would take for the software to operate as a vehicle for educational change in the system.

**Large and continual expert labor to develop the system and sustain the core user communities**

Part of the rationale for re-use enabled by component-oriented architectures is that it makes it possible to develop a low-labor technological solution to proliferating educational software. The argument, central to the ESCOT and EoE projects, is that this software is produced in an incrementally cheaper way taking advantage of the growing library of applets and then made available to users on existing platforms like the classical browsers and editors via an internet service such as a portal. Boxer and E-slate projects are on the other hand large – labor processes focusing on the idea of a custom desktop environment enabling users to hook up components and access their functionality in differing degrees. We suggest that these are not contradictory approaches but that each one makes use of a different strength of the component architecture, which can support both low-labor production and distribution and distributed user engagement and deep structure access. Large labor desktop services, however, help considerably in the support of the latter strength.

**Black-and-white box approach**

E-slate has adopted a black and white box approach in that it provides technically efficient black box components as higher – order building blocks to build software consisting of component configurations. These components are designed to be as generic as possible. E-slate authoring is not only based on the constructionist paradigm through building component configurations, but also on the connectivity metaphor, providing authors with multiple metaphors for connecting and thinking about component connections. We are investigating how the constructing – connecting combination can support creativity in building software. In this sense, E-slate is based on a ‘principled deep structure access’ design involving decisions on where to draw the access line in favor of technical efficiency and higher – order functionality constructions.

There is interesting debate on the white box versus black box issue. The malleability, the poking and tweaking and the ability to build things and customize them to meet personal requirements (diSessa, 1997) are all important when this technology is perceived as computational media and the arguments for this kind of deep structure access to software have been important in domain – related education fields (Mathematics and Science are the most celebrated) as well as in applications design (Eisenberg, 1995, Harvey, 2001). So, why has white box technology not yet spread in the culture as a computational medium? The main arguments of the component architecture rationale, accessibility, distributability and production of high quality software for education from a community generation, systemic, large-scale perspective seem to reduce the importance of the white box design rationale. Or is it just interpreted to be doing so? In any case, we suggest that these two are just different frameworks for design and it’s not necessary for one approach to have direct converse effects on the other. Black box approaches may be better in generating computational cultures at large scale in that, in the times that we live, they are easily understood and fit into peoples’ habits. We do not seem to be going through an era where bricolage is considered as a cultural habit in education and in the wider society, or as a habit, which despite the converse arguments from the field of educational researchers, is important to cultivate and spread.

The absence of this type of technological culture may have been one of the factors inhibiting the wide spread of white box technology. Pure white box approaches may result in eloquent ideas and arguments but need such a mindset shift that they are only useable in tightly controlled and supported situations and this might remain so for some time. On the other hand, black box cultures may get used to passive technology use which might create further problems. There are many factors influencing this of course, with industry and marketing of computational products being one of them. Its not the focus of this paper however. What's in focus is the idea of black and white box approaches.

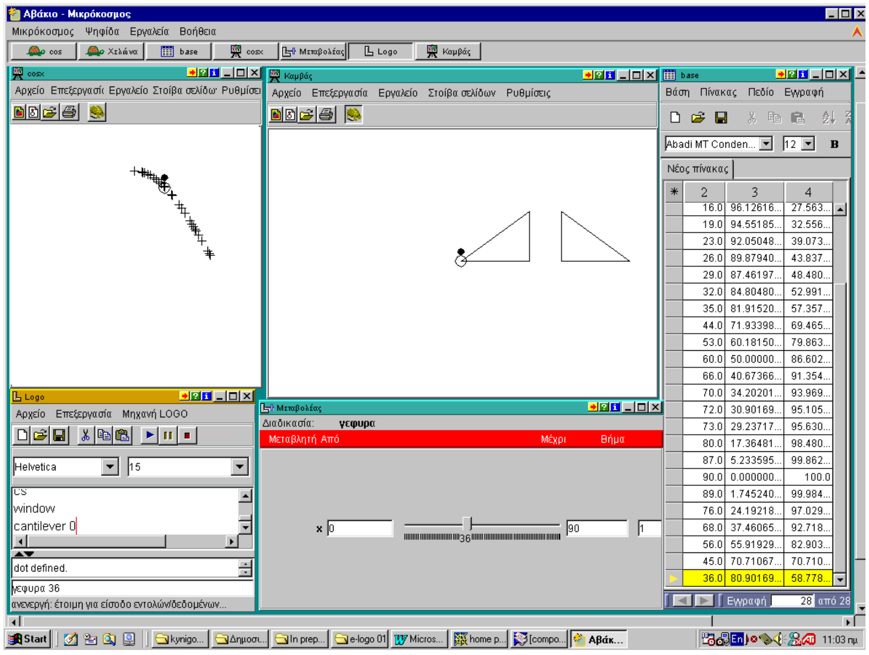
An example outside technology is the story of Lego bricks. In the old days, packages of Lego bricks contained generic bricks, which could be used in any construction and posed no restrictions on creativity in building various models. Recently, Lego is adopting a different approach where each package has a specific picture of a model to be built with it and special custom made bricks for that model. There are a few generic looking bricks but only enough to build the specific model. The scope for creativity is thus considerably reduced, if not killed. On the contrary, the black and white box approach is about packaging sets of combinations of custom and generic bricks with only some suggestions as to what to build.

So, we've taken a black and white box approach in designing E-slate. Components are black boxes in that the user cannot alter their main functionality and in that they are developed primarily to be technically efficient. However, each component is designed so as to be as generic as possible, in the sense that it can be used for a family of activities and not just a discrete set of activities. Example is the database component, the agent component, the map-G.I.S. component, the canvas component and of course the language-scripting component. This of course leaves E-slate open to criticism on the level of specificity which can possibly be reached in developing software through authoring. However, we argue that this compromise is not without gain at another level. This level is about meeting authors half way. Providing them with some technically efficient ready made building blocks with which they can build as complex software as they like (figs. 1, 4, 5). The whole issue is therefore where to draw the white box – black box line. We call this issue 'principled deep structure access', in the sense that the designer makes decisions on what is not so important for the user to be able to break down into further constituent parts in order to gain higher order building blocks with which they can create interesting efficient software in a more focused way. To what extent is constructionism compromised? Well, the author can still build component configurations without limits focusing not only on their composite functionality but also on the issue of connectivity. This is a new forum for creativity and learning, since the types of connections between components, if designed appropriately, may provide another field for constructions.

## Creativity in connecting and constructing component configurations

There is some criticism of component systems that despite the claims for authorability and re-use, there is not much creativity or interest in hooking up ready-made general - purpose pieces of software. Unless there is deep structure access and specificity, users will always find something they want and cannot do, to an extent making the re-use factor problematic. This would imply that even if component architectures could reduce the cost of educational software development and enhance access through new distribution mechanisms, the software itself would be of questionable quality and poor malleability.

In E-slate design, we are attempting to question this criticism, by creating a desktop environment to build component configurations, by developing generic components and a set of metaphors to connect components together. In that sense, we suggest that there might be a lot of creativity in the choice and configuration of components working as a whole piece of software and in the ways in which these components are connected. E-slate emphasizes the idea of component connectivity in the sense that it provides more than one connection metaphor, i.e. prefabricated connections through a plug icon-driven interface, a nesting metaphor through a drag and drop interface and the means for users to define their own connections through a scripting language and event handling (Logo, Javascript). We have had little time to work with our user communities in authoring software by those means in a teacher professional development setting (project E42, figs. 1 and 2) and in a research project aiming to support the generation of an institutionally distributed community of practice based on integrated expertise (project SEED, figs. 4 and 5). There are two points of interest emerging from this experience with respect to the produced software and to the process of authoring.


Αβάκιο - Μικρόκοσμος
Μικρόκοσμος Ψηφίδα Εργαλεία Βοήθεια
cos
Χελώνα
base
cosx
Μεταβολέας
Logo
Καμβάς
Αρχείο Επεξεργασία Μηχανή LOGO
Μεταβολέας
Βάση Πίνακας Πεδίο Εγγραφή
Νέος πίνακας

```
cs
window
cantilever 0
```

```
dot defined.
γεφυρα 36
ανενεργή: έτοιμη για είσοδο εντολών/δεδομένων...
```

Διαδικασία: γεφυρα

| Μεταβλητή | Από | | Μέχρι | Βήμα |
|---|---|---|---|---|
| x | 0 | 36 | 90 | 1 |

| * | 2 | 3 | 4 |
|---|---|---|---|
| | 16.0 | 96.12616... | 27.563... |
| | 19.0 | 94.55185... | 32.556... |
| | 23.0 | 92.05048... | 39.073... |
| | 26.0 | 89.87940... | 43.837... |
| | 29.0 | 87.46197... | 48.480... |
| | 32.0 | 84.80480... | 52.991... |
| | 35.0 | 81.91520... | 57.357... |
| | 44.0 | 71.93398... | 69.465... |
| | 53.0 | 60.18150... | 79.863... |
| | 60.0 | 50.00000... | 86.602... |
| | 66.0 | 40.67366... | 91.354... |
| | 70.0 | 34.20201... | 93.969... |
| | 72.0 | 30.90169... | 95.105... |
| | 73.0 | 29.23717... | 95.630... |
| | 80.0 | 17.36481... | 98.480... |
| | 87.0 | 5.233595... | 99.862... |
| | 90.0 | 0.000000... | 100.0 |
| | 89.0 | 1.745240... | 99.984... |
| | 76.0 | 24.19218... | 97.029... |
| | 68.0 | 37.46065... | 92.718... |
| | 56.0 | 55.91929... | 82.903... |
| | 45.0 | 70.71067... | 70.710... |
| ▶ | 36.0 | 80.90169... | 58.778... |

Εγγραφή 28 από 28

*Figure 1. The "Cantilever Bridge" microworld. It was developed by the E42 teachers and instructors. It consists of a model animated by the variation tool. It is an investigation on the width and height of the opening of a cantilever bridge, and the student needs to work with relationships of the sin function. In the software, as each model changes by dragging the variation tool, the results of measurements of interesting features are sent to a database. These values are then plotted on to a graph. At a first glance, the software resembles classical applications of the type, dynamically manipulating a geometrical figure and observing a graph of the relationship between chosen features being formed. However, the teachers designed this as a half – baked microworld for students. In their written activity plans, the student is meant to start off by observing what goes on when you drag the variation tool and from then on change the model by working with the code. Some potential changes for the model might be to generalize it further by including more variables, to select other features and plot their relationships in order to investigate patterns or rules, to superimpose the curve of a mathematical formula in order to observe how its fits the plotted points from the database.*

The software itself, or 'E-slate microworlds' as we call user authored component configurations, can easily involve composite functionality, which would be very costly to develop from scratch in standalone architectures. For example, our users have built a piece of software combining mathematical modeling through Logo constructions with direct manipulation of these models, data collection through measurements in the ways the models change and graphing of relationships of parameters (fig. 1). In standalone terms this would mean a composite functionality of software like Function Probe, Cabri and Logo. Another example is a piece of software for historical reasoning through excavations by finding artifacts on a layered map, making hypotheses about their use, their time e.t.c. and looking up on a database to find possible similarities to other artifacts (a version of the one in fig. 5). This would involve a composite functionality of a G.I.S. system and a Tabletop like piece of software. Component connections can determine the ways and direction in which data is passed from one component to the other and the interdependency between components.

The process of joining up components, of discussing their functionality and determining the ways in which they might be connected to do something interesting have been excellent opportunity to generate reflection and discussion amongst our users on issues such as epistemology (what is mathematics, science, history), one what it means to learn and on teaching methods. In our project to train teacher educators, for example (project E42), we observed our mathematics teachers progress:

- from a perception of using the computer to create a model built with unquestioned mathematics
- to discussing the mathematics in order to design a tool for others to experiment with
- to deciding on "taking away" some of the mathematics in order to give the students the opportunity to carry out an experiment and complete the building of the model by inserting a mathematical relation the worked out for themselves.

We thus suggest that perspectives on authoring do not need to be restricted to just the easiest means to democratize the production of software but can be of enough interest to build professional development programs based on this kind of activity.

## Technology

### The desktop environment

The rationale for developing a component desktop environment for E-slate was two-fold. Firstly, in order to avoid dependency on browsers, operating systems and hardware platforms in order to view, connect and manipulate components (a convincing discussion of the problems inherent in such a case is made by J. Roschelle and the ESCOT project). Secondly, to make available E-slate's authoring functionalities and semantics, so that using components as building blocks to create complex configurations (rather than perceiving authoring as just joining up specific sets of components in more or less standard ways), placing, ordering, moving and sizing them on the workspace and thinking about different ways to connect them can be made accessible to a wide user community. In this sense, we aim to meet the authoring community half way with respect to the Boxer and ESCOT/EOE rationale, asking the question of whether our 'black and white box' compromise on deep structure access by providing prefabricated higher – order building blocks can be compensated by the ability for authors to create computationally efficient complex functionalities.

### Generic components

There is of course interesting debate on whether it's possible for a component to be generic enough so that it is re-usable in a broad enough variety of situations without loosing out to the specificity and malleability authors might want when they configure it to match their exact plans for use. Although we certainly do not want to break loose from the community of programmability puritans, or dissociate E-slate from the family of programmable software (we believe that it has more chances of being strengthened if criticized as such), we have tried to grapple with the idea of just how generic components might be and how much is actually lost by the inevitable shortcomings on the specificity and malleability issues. We thus intentionally tried to design components to be as generic as possible having in mind a well defined but as broad as possible family of uses authors might find for them. Some such examples are the data-base component, the agent component, the variation tool component and the vector component (Hoyles, Newman and Noss, 1997). We have thus developed around 40 generic components trying to keep as open minded as possible the configurations we might see them in later on. The process of design has inevitably been bottom up and cyclical, experience with the user communities has made us go back more than once to re-think about the granularity issue and re-design some of the components so that they have more clearly defined generic behaviors. So, at this moment, the components we've developed can be categorized in the following groups:

- Data handling
- Modeling (simulations and representations)
- U.I.
- Media handling

As it stands, E-Slate components have to be Java Beans. It is however, technically easy to enhance the E-Slate Logo component to become a Logo-based E-Slate component (i.e. the author writes the component's behavior exclusively in Logo). In this case, E-Slate would still regard this component as an E-Slate-Logo Java Bean (so far only Brian Harvey has asked whether this could be done). In fact, we have had third party developers create components (with Java) and use them in conjunction with our family of generic ones. This would imply that E-slate could host any piece of software as such, but for it to gain by the possibility to be connected to E-slate components some (not a lot) changes have to be made so that connectivity is made possible and available through the E-slate semantics and metaphors.

## Component connectivity

E-slate emphasizes connectivity by adopting a variety of ways to connect components including a scripting language with which users may define their own connections or associations between components through event handling. The rationale is that there may be creativity in the ways components are connected and the types of connections available or sensible. Furthermore, connections can be made at two levels, i.e. by hooking up pre-fabricated connections through an icon-driven interface (plugs and drag drop for component nesting) and by defining connections through scripts. The icon driven connections may in some cases be an easier way into E-slate authoring even if its at the expense of deep structure access which may be made possible at a later stage. In other words, scripting is not necessary in order to be able to do interesting things with E-slate but provides deeper access and a larger variety of things authors can do by defining their own connections (Kynigos, 2001).

### Prefabricated types of live-link-connections, manifested by "Plugs" (color, shape, direction, multiplicity)

These are manifested by 'plug' icons with a set of connectivity semantics. Color means type of connection. Some connection types are hosting, projecting, spatial coordination, time coordination and others. There's two kinds of shapes, ones which denote joint behavior and ones which denote data-passing. Another semantic is cardinality, i.e. whether a connection can be made from one component to many. Finally, there's the direction of connections denoted by a positive – negative semantic. The author can think about and combine these four different sets of semantics when joining up components, provided that joining up is possible (e.g. different colors or negative to negative don't join).

### Scripts for user-defined connections

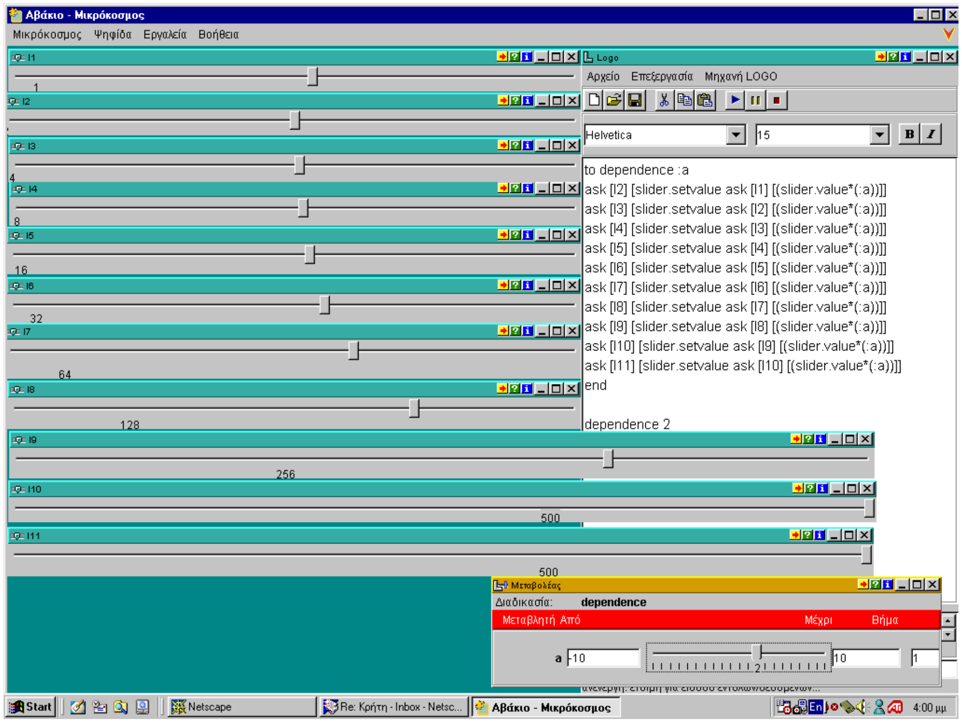
Each component carries its own domain – specific primitives and dynamically adds them on to the scripting component when inserted onto the desktop and into a microworld. This means that the user has a full programming language available to write scripts defining component associations through e.g. the TELL, ASK, EACH commands taking component names as inputs. Through these scripts, for instance, data may be passed from one component to the other, an event can be triggered in one as a result of another event triggered in another e.t.c. In this sense, users can define their own types of associations between components, which are different than the prefabricated ones available through plugs (see for example, fig. 2).

### Event-scripts for user-defined connections

Associations through scripting can be enhanced by means of event handling. There is a choice of events like change of state or clicking permutations. The author can thus create some dependency between components which will be fired every time the event takes place. A trivial example is to make a slider take twice the value of another one each time the cursor is slid in any direction. This would create a function of 2*x between the values of the two sliders, something which cannot be done by joining the sliders with plugs, in which case they take the same value, as in figure 2.

### Parent-child connections for implementing "nesting"

This connection is made by drag and drop and co-exists with the other types of connections. It enables the author to 'group' components together and thus make the system treat them as entities or building blocks which can then be used to build more complex configurations.



```
to dependence :a
ask [I2] [slider.setvalue ask [I1] [(slider.value*(:a))]]
ask [I3] [slider.setvalue ask [I2] [(slider.value*(:a))]]
ask [I4] [slider.setvalue ask [I3] [(slider.value*(:a))]]
ask [I5] [slider.setvalue ask [I4] [(slider.value*(:a))]]
ask [I6] [slider.setvalue ask [I5] [(slider.value*(:a))]]
ask [I7] [slider.setvalue ask [I6] [(slider.value*(:a))]]
ask [I8] [slider.setvalue ask [I7] [(slider.value*(:a))]]
ask [I9] [slider.setvalue ask [I8] [(slider.value*(:a))]]
ask [I10] [slider.setvalue ask [I9] [(slider.value*(:a))]]
ask [I11] [slider.setvalue ask [I10] [(slider.value*(:a))]]
end

dependence 2
```

*Figure 2. The $ax^n$ function microworld. The E42 teachers represented the curve of $ax^n$ by means of the positions of the cursors in consecutive sliders, each taking the value of the previous and multiplying it by 2.*

## Social Niches

### Social and systemic Context

Centralized system, uniform practice, encyclopedic educational paradigm, minimal teacher education, e-commerce lagging, internet use lagging.

A major theme in the rhetoric for and criticism on component architectures for education is its potential with respect to large – scale economy at the level of production, distribution, access and use of educational software. Our feeling is that we need to understand more about the social and systemic factors influencing the spread of a technology using culture for learning and how these factors may be different in different societies. We borrow diSessa's term of 'social niches' (diSessa, 2000), to think about strategies to infuse the use of component based systems in already existing mechanisms of educational support and activity, as in the case of ESCOT's collaboration with the Math Forum. The idea of social niches is also useful, however, in addressing the converse case where in the absence of compatible mechanisms, these niches are hard to find and are thus by default a weak case for replicability of component software proliferation. In such cases, which seem to be much more in line with the experience in the E.O.E. and the E-slate projects, component architectures may be considered as tools for creating mechanisms to generate educational communities engaged and supported in educational innovations. In the context of the Greek education system, for example, we have a centralized nation-wide administration coupled with a single national curriculum placing the teacher in the role of the technical implementer of this curriculum. It is thus very hard to distinguish innovation from systemic reform and to perceive of teachers involved in curriculum design and in trying out alternative teaching methods, both of which are central to the use of educational software, component based or not. Furthermore, there is almost non-existent teacher education at both pre-service and in-service levels, resulting in a teacher practitioner community of a static nature. This is enhanced by the teacher intake age (the average age of Mathematics teachers in secondary education in Greece is 49 years), which is a result of a queuing system which was only recently substituted by an examination-based one.

Another issue of importance across societies is that of the educational paradigm, i.e. the cultural heritage of what education is about and what type of service the educational system is designed to provide and preserve (Kontogiannopoulou - Polidorides, 1996). In the case of Greece, for instance, the educational paradigm is a descendant of the central European one, where the role of schooling is to provide all students with all human knowledge. Schooling is thus seen as the medium to bring information to students in the goal to provide them with a wider comprehensive education as part of their cultural heritage. In Greece, the prevailing educational paradigm has been termed 'humanistic' and 'encyclopedic' by McLean, (1990) and is characterized by a revelatory approach emphasizing a religious and a national identity culture (Kontogiannopoulou - Polidorides, 1996). In contrast, in the United States, the educational paradigm has been characterized as 'pragmatist' with a main emphasis on preparing the student for realistic adult life with emphasis on employment (Holmes and McLean, 1992). Designing a use of technology at the level of educational policy, promoting learning by construction, experimentation, collaboration, critical appraisal of information and dealing with uncertainty and ambiguity, would thus constitute an attempt to implement quite a radical innovation into the Greek system, not only at the level of new academic knowledge about the way we learn (as may be the case in the States), but more deeply at the epistemological level of what education is about.

So, our attempts to infuse E-slate use in Greece was to try out a variety of strategies. One was at the school as organization level, i.e. to support school-wide innovations, which were at the fringes of the education system, which engaged teachers and students in creative project work and which provided profile enhancement to the schools' administration. Another was to try out anchoring technology infusion to an attempt to institutionalize systemic change by means of the Odysseia project designed to establish a system of in-service education in technology use for all curriculum subjects. A third was to go the other way and circumvent systemic and organizational restrictions by generating an institutionally distributed community of integrated expertise constructing software and innovative activity plans for student projects. Finally, we used the Odysseia project as a forum through which to collaborate with third party organizations to develop components and component configurations using E-slate.

## Use for educational innovation

<u>Learning process, collaborative learning, professional development, implementing school – based innovation.</u>

An important part of the issue of user community engagement is the extent to which educational software use is perceived as trying out some educational innovation and the extent to which this is a norm or an exception in the surrounding social context. As outlined above, in the specific context E-slate has been tried out so far, designing and implementing innovation is not part of the norm. The type of innovation we tried to associate E-slate use with was to do with emphasis on student learning process, on investigational project work, on collaborative learning. With respect to the teacher community, our seminars, which were based on reflection, software authoring, teacher education practice, were considered innovative in the sense that they were not just informative as was the norm. Finally, the idea of school wide innovations (i.e. all teachers trying out alternative activities with their classes) was relatively new and only adopted by some private schools with extended daily programs.

## Authoring education and support

Despite the advent of exploratory software and the numerous projects and case studies involving authoring and microworld design by teachers, we have not yet overall seen much spread of such self – evolving cultures constructing microworlds and pedagogies. The turbulence created in teachers' perceptions about their profession and their role, their epistemological and pedagogical beliefs and the pragmatics of their professional situations has to date proved rather a tall order with respect to our earlier expectations of change. Moreover, using powerful metaphors such as the ones in the Boxer project, has not gained much in popularity alongside wide – use technological developments that are often advertised as an extension to the television, the encyclopedia, the public services and the phone rather than media for expression and construction.

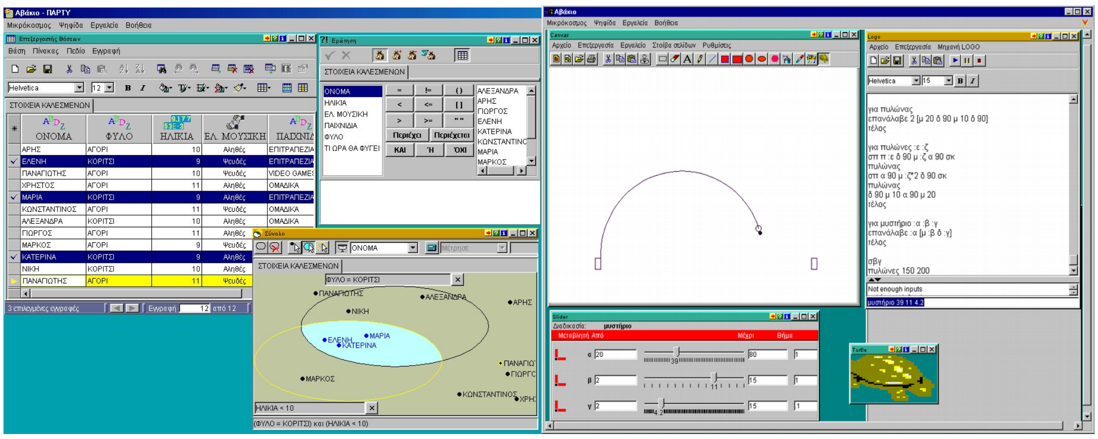
In accordance with the Boxer project, we perceive as an important means of proliferating the use of the E-slate system, the design and implementation of user professional development courses in the authoring of software. E-slate is not as easy to learn as Boxer at some levels, since it has a large variety of ready - made generic components whose functionality needs to be understood and a set of connectivity metaphors which do different things and have different interfaces.

A main feature of the teacher education courses has been work with E-slate which begins with what we call “half – baked” microworlds. These are microworlds designed so that the teachers would want to build on them, change them or de-compose parts of them in order to carry out some mathematics for themselves or to build microworlds for students. They are meant to operate as starting points, as idea generators and as resources for learning how to define connections between components and to customize their behaviors. In a sense, they operate like diSessa’s toolsets (1997) in that they are not built and presented as ready - made environments to be understood by the teachers and then used by students. Instead, the point is to change and customize them and thus to gain ownership of the techniques and the ideas behind microworld construction as outlined earlier.

## Different Strategies for user engagement

### <u>Level of School</u>

In the early stages of E-slate development, the development platform we used was buggy and created huge technical problems (OpenDoc), the desktop environment was at the stage of a very first prototype and we could only start by developing a small number of components. Furthermore, we had already started out with our collaboration with five schools helping them implement a project based school wide course using exploratory software. By using E-slate, we thus wanted to also further our research agenda to understand the kinds of turbulence generated in this type of classroom activity and whether such an innovation was feasible at the level of school. This fitted with our agenda to use E-slate for geography and Mathematics, so we developed two microworlds, resembling the functionality of software already existing in the schools, but extending it so that teachers and students would see the point in substituting it with the old software. This strategy was used so that the E-slate software would be used as early as possible in school conditions that would be as close to normal mainstream school activities as possible. For example, the schools were using Logo, a word processor and graphics package for their computer projects, so the education research team decided to develop the corresponding components and the desktop environment within which they could inter-operate, which we called ‘Turtleworlds’. The added value for using ‘Turtleworlds’, however, was that it provided a mechanism to dynamically manipulate the values of procedure variables and observe how the graphics evolve as the value changes. With corresponding rationale, we built the ‘d-base’ microworld, the functionality of which resembles that of the Tabletop software (Hanckock, 1995). Teachers could thus significantly expand their project repertoire to data handling investigations (fig. 3).


Αβάκιο - ΠΑΡΤΥ
Μικρόκοσμος Ψηφίδα Εργαλεία Βοήθεια
Επεξεργαστής Βάσεων
Βάση Πίνακας Πεδίο Εγγραφή
ΣΤΟΙΧΕΙΑ ΚΑΛΕΣΜΕΝΩΝ
Ερώτηση
Σύνολο
Canvas
Slider
Logo
Αρχείο Επεξεργασία Μηχανή LOGO
Turtle

| | ΟΝΟΜΑ | ΦΥΛΟ | ΗΛΙΚΙΑ | ΕΛ. ΜΟΥΣΙΚΗ | ΠΑΙΧΝΙΔ |
|---|---|---|---|---|---|
| | ΑΡΗΣ | ΑΓΟΡΙ | 10 | Αληθές | ΕΠΙΤΡΑΠΕΖΙΑ |
| ✓ | ΕΛΕΝΗ | ΚΟΡΙΤΣΙ | 9 | Ψευδές | ΕΠΙΤΡΑΠΕΖΙΑ |
| | ΠΑΝΑΓΙΩΤΗΣ | ΑΓΟΡΙ | 10 | Ψευδές | VIDEO GAMES |
| | ΧΡΗΣΤΟΣ | ΑΓΟΡΙ | 11 | Αληθές | ΟΜΑΔΙΚΑ |
| ✓ | ΜΑΡΙΑ | ΚΟΡΙΤΣΙ | 9 | Αληθές | ΕΠΙΤΡΑΠΕΖΙΑ |
| | ΚΩΝΣΤΑΝΤΙΝΟΣ | ΑΓΟΡΙ | 11 | Ψευδές | ΟΜΑΔΙΚΑ |
| | ΑΛΕΞΑΝΔΡΑ | ΚΟΡΙΤΣΙ | 10 | Αληθές | ΟΜΑΔΙΚΑ |
| | ΓΙΩΡΓΟΣ | ΑΓΟΡΙ | 11 | Αληθές | |
| | ΜΑΡΚΟΣ | ΑΓΟΡΙ | 9 | Ψευδές | |
| ✓ | ΚΑΤΕΡΙΝΑ | ΚΟΡΙΤΣΙ | 9 | Αληθές | |
| | ΝΙΚΗ | ΚΟΡΙΤΣΙ | 10 | Αληθές | |
| | ΠΑΝΑΓΙΩΤΗΣ | ΑΓΟΡΙ | 11 | Ψευδές | |

3 επιλεγμένες εγγραφές   Εγγραφή 12 από 12

ΦΥΛΟ = ΚΟΡΙΤΣΙ
ΗΛΙΚΙΑ < 10
(ΦΥΛΟ = ΚΟΡΙΤΣΙ) και (ΗΛΙΚΙΑ < 10)

```
για πυλώνας
επανάλαβε 2 [μ 20 δ 90 μ 10 δ 90]
τέλος

για πυλώνες :ε :ζ
σπ π :ε δ 90 μ :ζ α 90 σκ
πυλώνας
σπ α 90 μ :ζ*2 δ 90 σκ
πυλώνας
δ 90 μ 10 α 90 μ 20
τέλος

για μυστήριο :α :β :γ
επανάλαβε :α [μ :β δ :γ]
τέλος

σβγ
πυλώνες 150 200
```

```
Not enough inputs
μυστήριο 39 11 4.2
```

Figure 3. *The ‘Party project’ developed with the database microworld and the ‘Bridge project’ developed with ‘Turtleworlds’.*

So far the teachers have used these microworlds in their classrooms extensively but have engaged far less in designing or modifying any activities or software. These have been designed by the education research team but have subsequently been renegotiated many times with the teachers.

The schools project, which began with one school in September 1986 and had the others join in later in 1995, has been based on classroom activity from the outset[1]. It has involved teacher education, curriculum development and research on teacher strategies and children's learning. From year 3 to 9 inclusive (i.e., children aged 8 to 15), we have tried to help generate a school policy of all teachers working with their own class. In primary, this has been pretty straight forward, providing teachers with an extra two hour period a week to establish small group project work with their class. In secondary, we take the computer science hour, join it with one hour from another subject alternately and have teachers design inter-subjective projects for their students in a joint two-hour period. In this context, technology has been used to set up an unconventional classroom practice; all the teachers have been developing strategies for a pedagogy encouraging collaborative investigations centered around constructions and experiments with 'Turtleworlds' and the 'd-base' microworld. Pupils work in groups of two or three; they are encouraged to discuss ideas, to negotiate their mutual cooperation, to persist on a problem and deal with it in depth, to develop autonomy and responsibility towards themselves and their group peers, to present and discuss the results of their projects. Studies of related issues in school settings can be found in Hoyles et al. (1992), Hoyles and Sutherland, (1989) and Kynigos and Theodossopoulou, 2002.

So, trying out buggy versions of E-slate and E-slate microworlds was not very flexible, as in the ESCOT project, since a large part of it involved a school - wide implementation. The scheme was to substitute software used by the school at the time with E-Slate microworlds of the same functionality as they came out. The "Open Doc" story was a disaster since the software non-deterministically destroyed system files and operations (you can imagine what that does to a school). However, research was carried out on different aspects of classroom life, not focusing on the functioning of E-Slate per se (e.g. teacher beliefs and strategies see Kynigos and Argiris 1999, Kynigos 1995), apart from a couple of attempts (Kynigos 1997). This research, amongst others, has been operational in designing parts of the educational policy for integrating New Technologies in the system.

In spite of this experience, the E-slate Project was quite bold in using the Java version at a very early and buggy stage across the schools in 98-99. That year was quite dangerous to the project since people were already cautious from the previous experiences and the version was very buggy to survive in school settings - but somehow it did (project NETLogo). This had the consequence that there was very intense and urgent need for debugging and it resulted in having a version, which could survive in the classroom from the beginning of 99-00. The climate now is much better in the schools.

This is why the end user community did not operate from the beginning as software authors but more as designers, users and debuggers, coupled with the fact that there were not many components available so that configuring them in different ways could be meaningful. Also some were not very accessible to users like the map component with which, in the current version, the user can build their own G.I.S. - like map. During that phase of the Project, some component configurations were made by the unavoidably necessary direct support from the development team. These resulted in some "classical" exploratory software – type microworlds developed through a systemic project, 'Odysseas', such as Mycenae, Xenios, a Physics simulation. Apart from their functionality being pretty typical of such software, these microworlds had little authoring features showing some added value to using component architecture, apart from a straightforward connection of components.

However, things have turned around in the past three years. E-Slate has been more operational and stable and a large number of components have been developed (more than 40), some of which are "requests" from other organizations within the framework of collaborating for the projects. The project went through a phase experiencing development, which was too fast and varied for the user community to digest and give feedback. There were educational researchers who felt a loss of grip in contributing to priorities and timing for component development and developers (many of them new so they don't know the history) frustrated from the lack of immediate user feedback. Through the types of projects described next, however, we have established a process of re-energizing the user communities to respond to a more stable and rapidly developing E-Slate. The project has picked up momentum by means of the following two strategies for community generation and support. The first is a the level of

[1]School projects in Greece are few and usually restricted to an intensive teacher - training course outside classroom activity and with no follow-up related to it. Alternatively, the researchers take over classroom teaching without the presence of the teachers. There is great resistance to "outsiders" participating or influencing normal classroom activity.

educational policy for teacher education, by organizing the generation of teacher authoring communities and the development of microworlds incorporating specific educational ideas. The second is at the level of an integrated institutionally distributed community developing microworlds and corresponding activity plans. Among the goals is to make use of the strengths of component architecture by incorporating original combinations of functionalities.

## Level of system

In 1996, the organization where E-slate development was based, undertook the management of a large project to infuse new technologies in secondary schools on behalf of the Ministry of Education, called 'Odysseia[1]'. Odysseia was the first integrated project with this kind of objective in Greece and consisted of projects for telecommunications' infrastructure, equipping schools with labs, portal development, teacher education and software production. E-slate was used in the latter two types of projects amongst many other pieces of software developed in Greece or adapted from internationally acclaimed exploratory software. In that sense, it gained from the funding, the relatively wide use and the wide publicity. It also suffered, however, from the corrupting influence the system has on innovations, eloquently articulated by Hoyles, 1997.

The design framework for the 'Odysseia' project was that educational use of CICT and the innovations it entailed, would not happen by itself merely by equipping the educational system with the necessary technical infrastructure, hardware and software and by subsequently implementing a 'Total Quality Management' style systemic reform (Prawat, 1996, Papert, 1993). Educational systems are conservative, rigid institutions by nature and thus inflexible to the rate and quality of change feasible within the Information Society. In this sense, it would not be reasonable to expect rapid developments constituting innovations with respect to respective educational paradigms. Papert (1993) and Hoyles (1993), for instance, have argued that the school and the educational system seem to fight back against innovation by transforming it so that it finally fits in traditional practice, thus neutralizing its impact. From the point of view of reform initiatives, the methods usually employed are within the T.Q.M. paradigm where detailed and precise expectations are mediated to educational staff. The reform is then assessed by means of the extent to which these expectations are met by measuring performances. In education, this paradigm for reform does not work. Prawat, (1996) and Grossman et. al. (2000), for instance, have argued for a 'Learning Community' approach to educational reform, where expectations are articulated in broader terms, leaving initiative for teachers to take the role of participators and shapers of the reform by means of being actively engaged in reflective pedagogy through interaction in teacher communities of practice. Evaluating the reform requires new methods of looking at social processes rather than results and needs to allow for arduous, messy procedures eventually leading to change, rather than linear ones.

Up until 1996, large-scale initiatives in Computational, Information and Communication Technologies (CICT) in Greece had mainly adopted the technical paradigm of adding a computer science hour in the curriculum and teaching about this technology as a new subject area (Kontogiannopoulou – Polidorides, 1996). CICT was thus an add-on to the existing educational system, allowing for actions like providing schools with technical infrastructure, hardware, educational software, computer science curricula and teaching staff to be fragmented and independent of each other. Only at the small-scale level were there some scarce school-based research–led initiatives for using CICT within innovative constructionist and social paradigms (Kynigos et. al, 1993, YDEES project).

The Odysseia project (1996-2001) however, incorporated an integrated strategy for a bottom – up, scaleable infusion of the use of CICT (starting from 10% of schools) and a paradigm where small-group collaborative project work using exploratory software was encouraged as an alternative way of teaching all subject areas (Maritsas et. al., 1992). An essential aspect of the Odysseia project was its integrated nature, i.e. that these actions would develop concurrently with considerable effort for co-ordination between them so that use of the school lab would be initiated in a reasonable and relevant way. It is important to note that none of the above actions had previously been initiated except at the level of equipping schools with under – used labs for the computer science course and respective teaching staff consisting at best of computer science engineers with no teacher training.

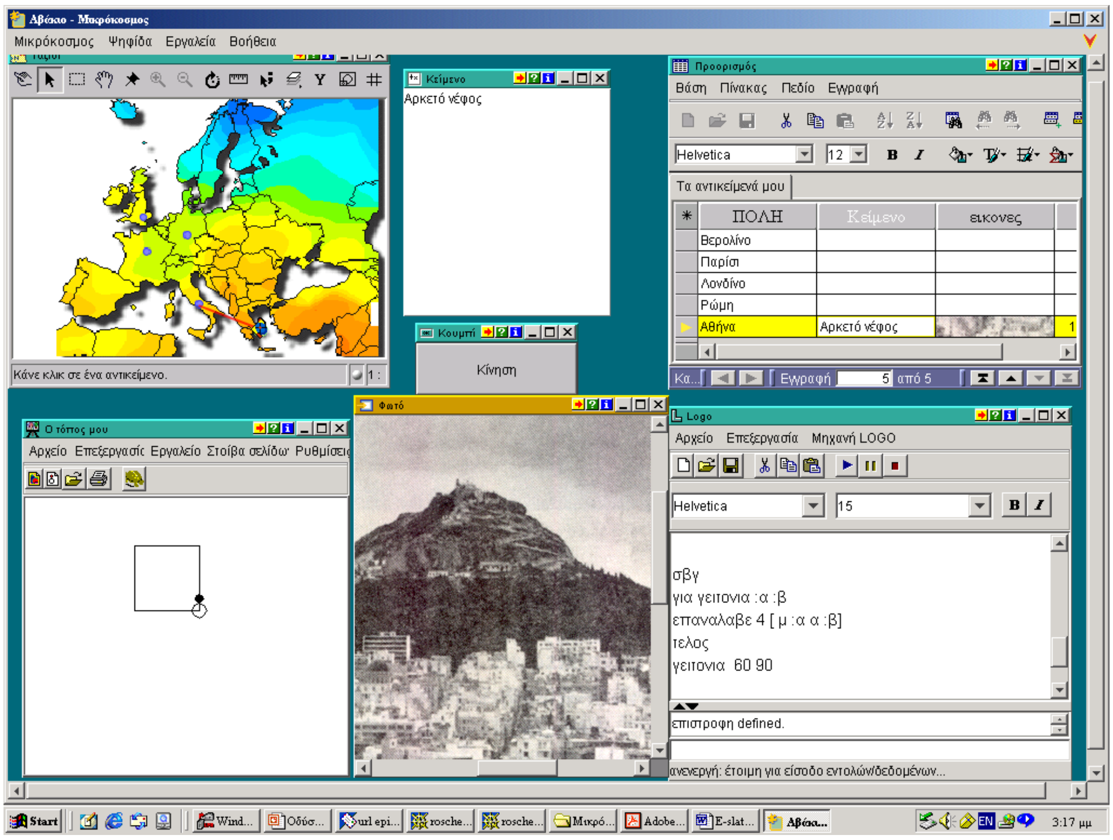
E-slate was used a) in a project to train teacher educators in the use of technology in their respective subjects, b) in a pilot integrated project to develop exploratory software for History, Learning of Foreign Language and Physics (E-slate team developers co-operated with technically inexperienced teachers) and c) in the software development project where third party institutions developed pieces of software by authoring and developing their own components supported by the E-slate technical team.

The objective of the teacher education course was to train experienced teachers, co-selected by the Ministry of Education, to become teacher educators in the use of CICT for teaching and learning in their respective subject. It was one out of three such courses funded by the Odysseia project and was carried out by the University of Athens group. During and after completion of the course, these teacher educators were relieved from their school duties and given the task to engage in in-service teacher education programs in 3-5 schools neighboring their own. Apart from the underlying innovation with respect to the educational paradigm, launching such an initiative from the perspective of educational policy, provided the Ministry with a new means to support teachers' in-service professional development. In Greece, pre-service teacher education is almost non-existent (only some optional courses are provided in undergraduate programs) and in-service education is ad–hoc and of an informing nature, rather than that of supporting the development of educational practice. Inevitably, teachers are seen as conveyors of the single national curriculum emphasizing information retention and drill rather than as reflective practitioners developing their own pedagogical styles and methods. The initiative was thus innovative in the sense that it meant to generate the need for institutionalization of teacher professional development support through the Ministry of Education. In the large-scale design, this would mean that the Ministry would invest in training around 1% of its staff (I take each school to approximately consist of 20 staff members) and subsequently employing them to support the infusion of CICT use in all schools as a steady state of affairs. A course to train teacher educators would thus need to perceive teacher education as a systematic, life–long professional development activity addressing teachers' epistemologies, practices, pedagogies and subject–related knowledge, rather than large-scale intensive, short-term seminars, which aim at providing information about technology itself to the teachers. It would entail teachers supported to adopt the role of reflective practitioners personally engaged in co-implementing the innovation. It would thus involve teacher educators in communicating and supporting this new role not only through the in-service seminars, but more broadly by generating learning communities in school (in the sense of Prawat, 1996 and Grossman et. al., 2000). This role for teacher educators cannot be prescribed and handed to them by their administration when it constitutes such an innovation. Rather, it needs to be actively claimed and shaped through practice. The course would thus need to support teacher educators in that role by taking a broad perspective in designing the syllabus. After the end of the course, a number of teacher educators are authoring pieces of software using E-slate among others as a platform and giving equal if not more importance to the escorting description of activity plans. They mainly do this in order to use their 'scenarios' in their teacher education courses and to distribute them to their colleagues through a teacher educator mailing list, usually referring to their personal web pages (see e.g. http://www.geocities.com/kampranis). However, in the wider teacher community and at the level of the administration at the Ministry E-slate is perceived as a piece of software rather than a platform for proliferating educational software or a computational medium. These ideas are difficult to understand at these levels. At the educational policy level, proliferating educational software more effectively by making use component oriented architecture is often considered as contrasting a policy to give the existing software industry a chance. E-slate was evaluated by the Pedagogical Institute as one single piece of software (it passed, so now its officially recognized).

The Odysseia project has been completed far too recently to be able to properly understand its impact and the way it has influenced systemic reform. However, the basic features of the picture have already begun to emerge. Not a lot of innovative classroom practice has been observed yet (Hadzilakos et. al., 2001). The phenomenon of 'term decadence' (teacher education can mean anything with respect to process, content and educator expertise, almost all pieces of software are characterized as exploratory software e.t.c.) is in abundance. Policies for nation wide uniform projects are underway (e.g. a 30 hour computer literacy course give to all 70.000 in-service teaches in Greece). Criticism that Odysseia was a luxury we can't afford, addressing only a select few, is being articulated to justify these huge wide coverage projects. On the positive side, authoring with E-slate has become a habit of a group of teacher educators, a set of E-slate microworlds resembling rather classical CAL software has been recognized and is used alongside with others. Like the EOE project however, the E-slate group is reflecting more widely on strategies which may make component based style efficiency understood and made use of at the level of systemic reform in centralized education systems. This is one of the objectives of our current project, SEED, which is described next.

## Institutionally distributed integrated communities

In both our school and systemic level strategies, we felt the difficulties inherent in developing, authoring and using E-slate and E-slate microworlds deriving from the restrictions posed by people working within the confinement of their organizations which are by definition a part of the system. For instance, small software houses often opt for lower labor solutions of customizing authoring tools rather than developing applications for scratch. In such a case, it would seem as rather arduous to develop a component and then only use it in one single project. In school, teachers have well defined schedules and are working in a context perceiving the teaching profession as static. For the school and the system, it seems as external activity to participate in authoring and 'scenario' development. At the level of policy, things which can have some impact in the political life cycle are of primary importance, so creating the resonance required for an integrated stepwise infusion of CICT use in the classroom look like a luxurious and inefficient (because of its longitudinal mature) enterprise.



Αβάκιο - Μικρόκοσμος
Μικρόκοσμος Ψηφίδα Εργαλεία Βοήθεια
Κάνε κλικ σε ένα αντικείμενο.
Κείμενο
Αρκετό νέφος
Κουμπί
Κίνηση
Προορισμός
Βάση Πίνακας Πεδίο Εγγραφή
Helvetica
12
Τα αντικείμενά μου
Εγγραφή 5 από 5
Ο τόπος μου
Αρχείο Επεξεργασία Εργαλείο Στοίβα σελίδων Ρυθμίσεις
Φωτό
Logo
Αρχείο Επεξεργασία Μηχανή LOGO
Helvetica
15
Start
3:17 μμ

| * | ΠΟΛΗ | Κείμενο | εικονες | |
|---|---|---|---|---|
| | Βερολίνο | | | |
| | Παρίσι | | | |
| | Λονδίνο | | | |
| | Ρώμη | | | |
| ▶ | Αθήνα | Αρκετό νέφος | | 1 |

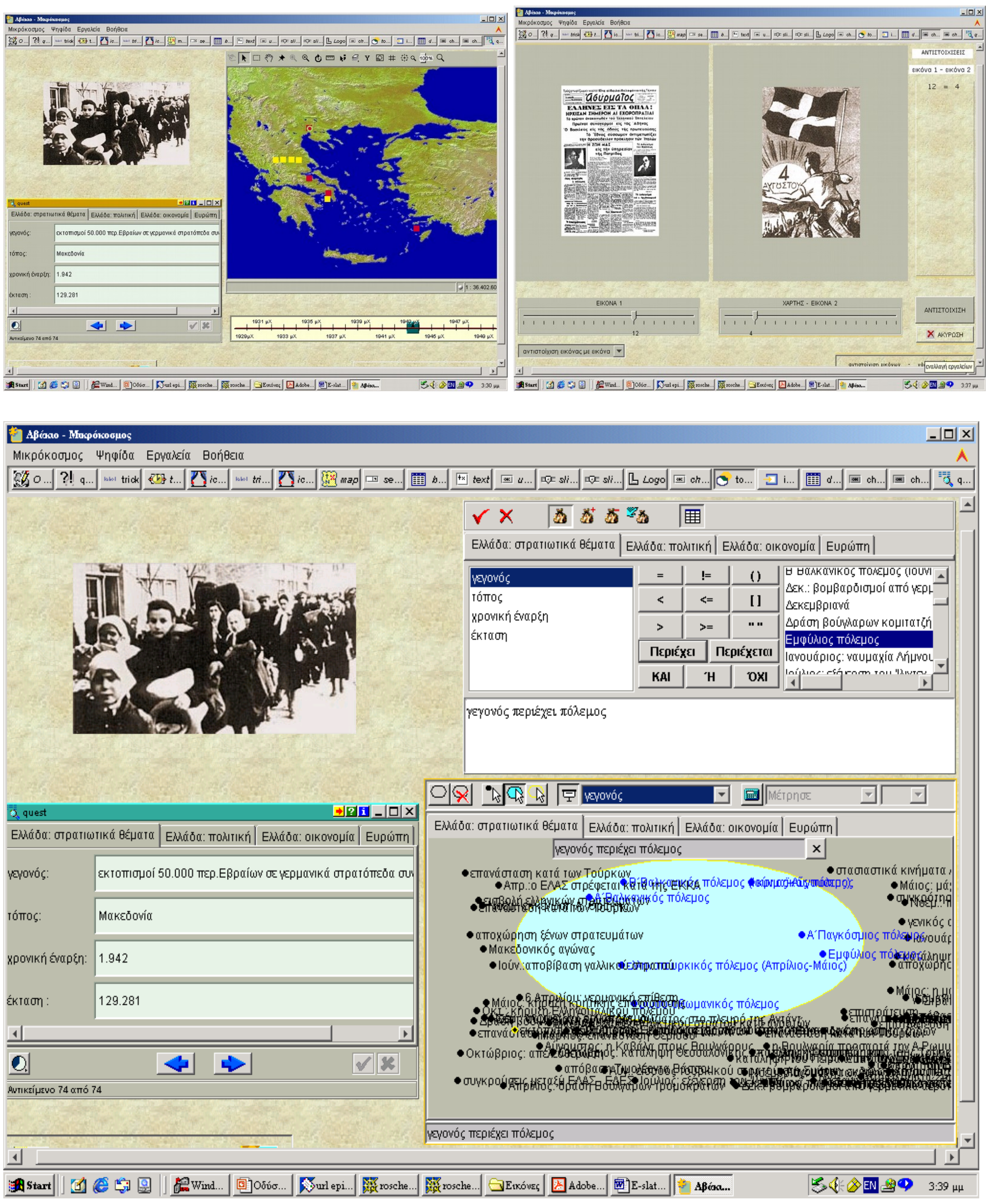
```
σβγ
για γειτονια :α :β
επαναλαβε 4 [ μ :α α :β]
τελος
γειτονια  60 90
```

```
επιστροφη defined.
```

```
ανενεργή: έτοιμη για είσοδο εντολών/δεδομένων...
```

*Figure 4. The Geography microworld, developed by two primary teachers, a computer science teacher and a member of the research team. Students can insert map icons, fill the data base with information, link the records to specific points on the map, command the agent to walk from place to place on the ma, either addressing names of places or coordinates, insert photos of places and write programs to make a graphical model of their map and think about scaling by using the variation tool. The microworld design decisions were discussed by the team.*

We thus tried an alternative strategy of using European Community funds to help generate and support an institutionally distributed community of complementary expertise. This project (SEED) has just completed the first out of three years. Our community of teachers from the school sites, teacher educators, e-slate group developers and educational researchers has produced over 40 E-slate authored microworlds. Our strategy is based on designing and supporting the generation of this community based on the notion of institutional distribution, constructionism, complementary expertise, mutual accountability, professional development support, productivity (we aim to develop a set of professional quality pieces of software and escorting documentation and jointly publish it).



γεγονός: εκτοπισμοί 50.000 περ.Εβραίων σε γερμανικά στρατόπεδα συ

τόπος: Μακεδονία

χρονική έναρξη: 1.942

έκταση : 129.281

Αντικείμενο 74 από 74

γεγονός περιέχει πόλεμος

*Figure 5. The 'Historical Facts' microworld was designed for the student to carry out historical investigations. The same microworld allows investigations of information in time layers (slider) and theme layers (different tables in the database). The student can make Boolean queries and observe the results on the table and on the set microworld (the picture shows the results to the 'facts contains war' question. Also, it has a match the pictures game, where two pairs of thematic pictures are shuffled and then the student can guess match the ones with associated themes. The pictures show three flip sides of the microworld. The microworld was developed by a History teacher educator who learned E-slate authoring during the SEED project and collaborated by email with the technical team.*

The function of the community is to co-develop a set of suggestions for innovative use of technology each one consisting of a carefully thought out activity plan and rationale and the corresponding software. The learning domains which seem to be emerging as preference are, Kinematics, Co-variation, Orientation and concept of space, Programming, Control technology, Language and History. We work by means of holding a series of presence workshops, by discussing educational, technical and collaboration issues with the use of an online forum and by working in small groups and collectively. Some of the dynamics emerging from our experience as part of this community will be discussed during the presentation, using data from our communications and our constructions. The questions we ask in this project are to do with the potential large-scale impact of supporting the generation of a number of such communities and making the experience and products available widely. Are the construction of tools, design of innovations and reflection strong enough community motivators for such communities to grow despite the turbulence created by institutional distribution?

## Development Strategy

### Multi-organizational context

The two E-slate groups mostly acted as two partner organizations collaborating with other occasional partner organizations in wide-scope R&D projects.

To deal with the need for composite expertise, organizations mostly employ *complementary actors*, e.g. either a programmer or an educationalist, depending on the expertise missing at the respective department. In many cases, these people would become *hybrid* actors, in the sense that they would acquire enough complementary understanding so as to effectively act as part of a team based on alien expertise. It was through this type of uni – organizational setting that the community working on the conception of technologies supporting the generation of mathematical meanings and the development of our understanding of this process as essential for learning mathematics was developed (for a discussion, see Kynigos, in press).

In the case of the E.E.C., the agenda for multi-organizational projects is *not* educational in the narrow sense, but more socio-political and developmental in nature: to create intercultural sensitivity and to generate more synergy between academia and industry. Researchers are thus faced with explicit encouragement to engage in R&D activity participating in multi-organizational consortia, rather than working through their organization and hiring hybrid actors. Not only this, but the directive is that these organizations should be a mix of academic, private sector and systemic (schools, local authority and ministry departments) institutions and that they should be situated in a variety of countries.

The interest shown by new communities, coupled with the encouragement of multi-organization projects, provide new potential and at the same time new complexity and challenge to the constructionist – exploratory educational software community. Given that organizations incorporate sustained and institutionalized expertise, there is the potential to have unprecedented support in all aspects of the development, production and infusion of exploratory software use in educational settings. There is the challenge, however, that the educational purpose of exploratory software has to be communicated and integrated with powerful communities with different perspectives and priority systems with respect to education. The traditional processes and methods of developing exploratory software for education seem very idiosyncratic when seen through perspectives held in the wider communities involved in educational reform and in software development in general. At the same time, however, it is this community that has the expertise and deep understanding of integrated software development for educational change.

### Large and diverse projects

The consequence of E-slate development through multi-organizational projects has been that each project "carried" E-Slate development and use a little further, but also that there did not appear to be outstanding success in each single project, due to the effort spent in setting up a variety of collaborations and focusing on the production of readily useable results. Also, although the objective in each case was typically to develop a specific piece of software, breaking it down to components and developing the desktop environment took a large part of the available resources, especially at the early stages. This resulted in a rather large overhead of bureaucracy, communication on peripheral issues and distributed focus. For instance, there were times when the technical developers had to focus on the desktop environment (i.e. the development platform) while the education team had to focus on improving component functionalities and uses in schools, with the result of one team being of little direct use to the other. The tight and continuing collaboration between the technical and education teams, however, was not only due to external pressures for organizational collaborations. From the start, a cyclical development method was adopted where the software was co-designed by the two teams and tried and tested as early as possible in situations which were realistic and which could be scaled up in the future.

**Institutionally distributed teams**

In the case of e-slate, the developers' work has been long term and on-going, since within the persistent goal of continual development of the 'desktop' and the construction and re-construction of components, there are all kinds of new technological developments which can be created and used. Within this framework, it made sense, for instance, to develop the researchers' community request for the variation tool components, i.e. for an idiosyncratic environment integrating symbolic expression and direct manipulation features. The key here is that this environment has survived many hardware, operating systems and technological changes and has evolved in itself through different ideas for links with other components – hooking it onto a database, for instance, was not part of the original functional specifications. On the other hand, the interest of school-based research was essential for e-slate use to survive school staff and researcher turnovers, changing labs, different phases in school priorities, not to mention the subsequent systemic and institutionally distributed levels of E-slate development and use. A crucial issue therefore, is to *set the right level of generality in the task* to preserve the interest of all the communities involved. What is needed is thus a careful *architecture of objectives*. This is hard to do and is a function of discriminating the rate with which different features of activity and technology are changing. I suggest that this can only be done through the development and preservation of *hybrid communities across organizations*. Uni – organizational or isolated hybrid actor situations are bound to fall short of the demands posed by the complexity of multi – organizational contexts.

## Conclusion: Component Architectures and the 'social empowerment versus instant fit' paradox

The experience with E-slate is showing what we've seen before with user empowering software, i.e. that it is quite a tall order to attempt to bring about serious societal change in a time frame of the order of magnitude of a few years (and not, for instance a few generations which would be more sensible given the history of proliferation of expressive media, see Kaput, Hoyles and Noss, 2001). In the context in which we have tried to place it so far, we've been faced with challenging the system, the school as organization, socially resident ideas about teaching and learning and about the epistemology of education. However, we do not believe that social change can come about in a few years, even if technology and the information society considerably reduce the time frame from the order of close to 10 generations, which seems to have been the norm so far. Quite frankly, we'd like to express some concern for criticism impatiently complaining that computational media style software like Logo, Boxer, Knowledge Forum, object oriented and declarative programming languages (e.g. Smalltalk and Squeak) and others have not changed society so far and are thus defunct as design rationales and as technologies (for a discussion of these issues in the context of the Logo phenomenon, see Hoyles and Noss, 1996 and Agalianos, 1997). Since we now have the technological means to create environments which have many faces, there is more grounds to argue for Trojan horse style design architectures (such as the one behind component rationales) which can play the role of instant societal fit, allowing for rapid spread of use, but have the potential for deep structure access through the preservation (rather than throwing them in a scrap heap) of empowering technologies and semantics. This would imply that the development of such software and the consecutive attempts to generate user communities should not be perceived as one-project long efforts or as the development of a single piece of technology, but as a continuing effort to build social change architectures. We suggest that saying that the ESCOT, E-slate, Boxer and EOE projects have been completed since we now have a set of technologies and user community generation experiments falls into the technocentric trap of perceiving scientific progress as the manufacturing of an endless series of gadgets.

## References

Agalianos A. S. (1997) A Cultural Studies Analysis Of Logo in Education, unpublished doctoral thesis, Institute of Education, Policy Studies & Mathematical Sciences, London.

DiSessa, A. (1997) Open toolsets: new ends and new means in learning mathematics and science with computers, Proceedings of the 21st Psychology of Mathematics Education Conference, Finland, Lahti, Pekhonenen (Ed).

DiSessa, A. (2000). Changing minds, computers, learning and literacy. Cambridge, MA: MIT Press.

Edwards, L. (1998). Embodying mathematics and science: Microworlds as representations. *Journal of Mathematical Behavior* 17 (1), 53 – 78.

Edwards, L. (1998). Embodying mathematics and science: Microworlds as representations. *Journal of Mathematical Behavior* 17 (1), 53 – 78.

Eisenberg, M. (1995), Creating Software Applications for Children: Some thoughts about Design, in *Computers and Exploratory Learning,* A. diSessa, C. Hoyles and R. Noss (eds.), Springer Verlag NATO ASI Series, pp.175-196.

Grossman P., Wineburg, S. and Woolworth, S. (2000) In Pursuit of Teacher Community, paper presented at the Annual Meeting of the American Educational Research Association, New Orleans.

Hadzilakos, Th., Kynigos, C., Ioannides, C., Vavouraki, A., Psicharis, G., Papaioannou T. (2001) Case Study of ICT and School Improvement, The Case of Greece. OECD/CERI ICT PROGRAMME, Center for Educational Research, Athens, Greece.

Harvey, B. (2001) Harmful to Children? The Alliance for Childhood Report, Proceedings of the Ninth Eurologo Conference, Lintz, Austria,

Hancock, C. (1995). The medium and the curriculum: reflections on transparent tools and tacid mathematics. In DiSessa, A.A., Hoyles, C. & Noss, R. (eds) Computers and Exploratory Learning. Berlin Heidelberg: Springer-Verlag. pp. 221-241.

Holmes B., and Mc Lean M. (1992) The Curriculum: a Comparative Perspective, London, Routliedge.

Hoyles, C. (1993). Microworlds/Schoolworlds: The Transformation of an Innovation. In C. Keitel & K Ruthven (eds) Learning From Computers: Mathematics Education and Technology, 1-17. Berlin/Heidelberg: Springer-Verlag, pp. 1-17.

Hoyles, C., Noss, R. (1996) *Windows on Mathematical Meanings*, Kluwer Academic Publishers, Dordrecht/Boston/London.

Hoyles, C., Newman, K. and Noss, R. (1997) Report on the FLYVER microworld and on components for education generally inspired by the FLYVER microworld.

Kafai, Y., Resnick, M., Eds. (1996) Constructionism in practice. Designing, thinking and learning in a digital world, Lawrence Earlbaum Associates, New Jersey.

Kaput, J., Noss, R. and Hoyles, C. (in press) Developing New Notations for a Learnable Mathematics in the Computational Era

Kontogiannopoulou - Polidorides, G. (1996) Educational paradigms and models of computer use: does technology change educational practice?, In Cross National Policies and Practices on Computers in Education, Plomp T., Anderson, R., E. and Kontogiannopoulou – Polidorises G. (Eds). Kluwer Academic Press, Dordrecht, 49-84.

Kontogiannopoulou - Polydorides, G. & Kynigos, C. (1993) An Educational Perspective of the Socio-cultural Prerequisites for Logo-like Education in Greece. Proceedings of the 4th European Logo Conference, Georgiadis P., Gyftodimos, G., Kotsanis, Y. and Kynigos C. (Eds), Doukas School Publication, 377-389.

Kynigos C. (2001) E-Slate Logo as a Basis for Constructing Microworlds with Mathematics Teachers, Proceedings of the Ninth Eurologo Conference, Lintz, Austria, 65-74.

Kynigos C. (in press) Generating cultures for mathematical microworld development in a multi-organisational context, Journal of Educational Computing Research.

Kynigos C. and Theodosopoulou V. (in press) Synthesizing Personal, Interactionist and Social Norms Perspectives to Analyze Student Communication in a Computer - Based Mathematical Activity in the Classroom, Journal of Classroom Research.

Kynigos, C. (1997) Dynamic Representations of Angle with a Logo - Based variation tool: a case study, *Proceedings of the Sixth European Logo Conference,* Budapest, Hungary, M. Turcsanyi-Szabo (Ed), pp. 104-112.

Kynigos, C. (in press) New Practices with New Tools in the Classroom: Educating Teacher Trainers in Greece, to Generate a 'School Community' use of New Technologies, Themes in Education.

Kynigos, C. and Argyris, M. (1999) Two Teachers' Beliefs and Practices with Computer Based Exploratory Mathematics in the Classroom, Proceedings of the 23rd Psychology of Mathematics Education Conference, Haifa, Israel, O. Zazlavsky ed., 3, 177-184.

Kynigos, C., Giftodimos, G. and Geordiadis, P. (1993) Empowering A Society Of Future Users Of Information Technology: A.Longitudinal Study Of An Application In Early Education. European Journal Of Inf. Sustems, Vol. 2, No. 2, Pp 139-148

Kynigos, C., Koutlis, M. and Hatzilacos. T., (1997). Mathematics with Component-Oriented exploratory software. *International Journal of Computers for Mathematical Learning 2*: pp.229-250.

M. Koutlis, Th. Hadzilacos, (1996) "Avakeeo: the construction kit of computerised microworlds for teaching and learning Geography", Second International Symposium on GIS in Higher Education, NCGIA, Baltimore, September 1996.

Maritsas, M., Hatzilakos, Th., Kynigos, C,. Koutlis M., Christodoulakis D. (1992) "Astrolavos": A Program to Utilize Computing Technologies to Support Education in the Hellenic Secondary Schools, Proposal submitted to the Ministry of Education under the supervision of Prof. D. Maritsas, Director of the Computer Technology Institute.

Noss, R. (1995) Computers as Commodities, in *Computers and Exploratory Learning,* A. diSessa, C. Hoyles and R. Noss (eds.), Springer Verlag NATO ASI Series, pp.363-381.

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

Prawat R.S. (1996) Learning community, commitment and school reform, *Jnl. of Curriculum Studies,* vol. 28, No. 1, pp. 91-110.

Solloway E. (1998) No one is Making Money in Educational Software*, Communications of the ACM,* 41, 2, pp. 11-15.

## Projects

YDEES: Development of Popular Computational Tools for General Education: The Computer as Medium for Investigation, Expression and Communication for All in the School, #726, E.P.E.T. II, General Secretariat for Research and Technology, 1995-1998.

I.M.E.L.: Intercultural Microworld courseware for Exploratory Learning, European Commission, Socrates, Open and Distance Learning, 1996-1998.

NETLogo: The European Educational Interactive Site. European Community, Educational Multimedia Taskforce, #MM1020, Joint Call in Educational Multimedia, 1998-2000.

KALIPSO: Management of the Computer and Communication Technologies in Secondary Education Project, Ministry of Education (Odysseia), Operational Programme for Education and Initial Training EPEAEK, Fourth Support Framework, European Community, 1996-2000.

ODYSSEAS: Integrated Network of schools Educational Regeneration in Achaia, Thrace and the Aegean, Ministry of Education , E11, Operational Programme for Education and Initial Training EPEAEK, Fourth Support Framework, European Community, 1996-1998.

E42: "Post Graduate Specialization of Teacher Educators in the Educational Use of Computational and Network Technologies in Secondary Education, University of Athens, School of Philosophy, Faculty of Philosophy, Education and Psychology, Dept of Education. Collaborating partners: Dept. of Informatics and Dept. of Methodology, History and Theory of Science. Funded by the Ministry of Education, 2nd and 3rd Community Support Framework, Odysseia Project, 1999-2001.

Project 'SEED': Seeding cultural change in the School System through the Generation of Communities Engaged in Integrated Educational and Technological Innovation, European Community, 5th Support framework, Information Society Technologies, IST – 2000 – 25214, 2001-2004.

[1] Odysseia is a long-term program of the Ministry of Education (1996-2001). It is funded by the Operational Program for Education and Initial Vocational Training (2nd Community Support Framework) with 27 M Euro and by the 3rd Community Support Framework with 11.3 M Euro. Odysseia was designed by the Academic Research Institute on Computer Technology (CTI), in co-operation with the appropriate departments of the Ministry of Education (Directorate for Secondary Education Studies, Directorate for the Community Support Framework and the Pedagogical Institute). A large number of organizations were involved in its implementation: 57 University departments, sectors, laboratories and research institutes, 71 software, hardware and telecommunications companies, publishers and museums. CTI's Educational Technology Sector coordinated the program in collaboration with the Directorate for Secondary Education Studies and the Pedagogical Institute. CTI's Networking Technology Sector was responsible for the School Network, while the Computer and Networking Services Lab was responsible for the school labs. CTI's Administrative and Financial Services undertook the Financial Management of the program. Over 1,000 scientists, educators, engineers, technicians and executive personnel were involved in the design, development and support of the technical and educational infrastructure that enables 385 schools throughout the entire country to take advantage of Information and Communication Technologies as a daily learning tool for all. Th. Hatzilakos was the director of the Odysseia project.