Investigación en Entornos Tecnológicos en Educación Matemática (2020), No. 1, pp. 9–14
ISSN *Pendiente de asignación* – DOI *Pendiente de asignación*

TaskTimeTracker: A tool for temporal analysis of the problem solving process

Marta Pla-Castells[1] e Ignacio García-Fernández[2]

Abstract
El análisis del proceso de resolución de problemas en matemáticas permite obtener información sobre cómo se produce el aprendizaje. Sin embargo, el análisis de este proceso es una tarea complicada debido a la complejidad y no linealidad del propio proceso. En este trabajo presentamos la herramienta de software **TTT** diseñada para facilitar el registro y la representación gráfica de los pasos que sigue un alumno o un grupo de alumnos durante la resolución de un problema junto con la evolución y duración en el tiempo de estos pasos. Esta herramienta se basa y amplía los esquemas de representación presentados por Ärlebäck (2009) y puede ser aplicada a cualquier proceso de resolución de problemas (matemáticos o no) que pueden ser divididos en fases o categorías a lo largo del tiempo.

Abstract
The analysis of the problem solving process in mathematics can shed light on the learning process. However, the analysis of this process is a difficult task that has to face the complexity and non linearity of the process itself. In this work we present the **TTT** software tool aimed to facilitate the registration and graphical representation of the steps that are followed by a group of students during the resolution of a problem, together with the time extension of these steps. This tool is based upon, and extends, the representation schemes presented by Ärlebäck (2009), and can be applied to any problem resolution process (either mathematical or not) that can be divided into phases or categories along time.

Palabras clave
Resolución de problemas — Diagrama de actividad — Herramientas Informáticas
Keywords
Problem Solving Process — Time Activity Diagram — Software Tools

[1] *Departament de Didàctica de la Matemàtica. Universitat de València. España*
[2] *Departament d'Informàtica. Universitat de València. España*
*__Autor de correspondencia__: Marta.Pla@uv.es

1. Introduction

Cognitive and work processes of students when they face the resolution of a problem is a recurrent research topic in the field of mathematics education (Pólya, 1945; Schoenfeld, 1985, 1992). The analysis of these processes can be done by means of the final productions of the students. However, when children work in groups to solve mathematical problems, the interaction and exchange of ideas has an important effect on the learning process, as they share other students' ideas and perspectives (Gillies, 2003). As a consequence, the analysis of the discussions and interactions during the problem solving process within a work group can also be of great interest (Berry & Sahlberg, 2006; Gillies, 2003; Vygotsky, 1980).

This information can provide a powerful tool during the analysis of the process of solving a problem, specifically from a macroscopic perspective. The macroscopic analysis of the cognitive processes can be rather difficult to be performed due to the complexity of the process itself and, in addition, to the presence of an unstructured interaction among the students. As a consequence, many researchers have sought for a proper codification of the steps and cognitive processes produced during the resolution of the problem, as well as for tools that lead to a clear representation of this codification. In such a codification, the variable "elapsed time" has to play a relevant role to properly represent the process as a time evolving series of tasks or episodes (Ärlebäck, 2009; Schoenfeld, 1985).

In this work we present a tool aimed to facilitate the registration and graphical representation of the steps that are followed by a group of students during the resolution of a problem. The tool registers not only the different phases that the students go through but also the time extension of these steps. This tool is based upon, and extends, the representation schemes presented by Ärlebäck (2009), and can be applied to any problem resolution process (either mathematical or not) that can be divided into phases or categories along time.

2. Theoretical Framework

Studies on the problem solving process have focused mainly on mathematical problems from the beginning, although different studies can also be found in fields such as Psychology (Brandell, 2010, p. 189) or Computer Science (Pearl, 1984) among others.

The basis for most mathematics problem solving research in the past 30 years can be said to be founded in the writings of Pólya (1945) who considered an ideal problem solver and described the mental processes or phases she would follow to solve a problem in mathematics. The steps are: to understand the problem, to devise a plan for the solution, to carry out the plan and to look back on the solution.

Extending Polya's approach, Bransford and Stein (1984) developed a 5-step linear problem solving model. In both models these steps are followed sequentially, addressing one phase after the previous one has been finished.

However, the findings by different authors indicate that the problem resolution process does not need to be a linear one. On the contrary, the progress of students through the different phases of problem solving is a dynamic and cyclic process in which they can shift the task they are engaged in without completing the previous one, and repeat the whole process several times before the problem is completely solved.

In this direction Wilson, Fernandez, and Hadaway (1993) state that a student may combine reflection and activity at the beginning of the process or decide that she needs to understand the problem better while a plan is being devised. Or, when a devised plan fails during the execution phase, the student may jump back to the plan definition step again and re-elaborate it or, even, retake the task of understanding the problem.

As a consequence of this nonlinear essence of the problem solving process, the use of simple flow diagrams, such as the one shown in Figure 1, do not allow a clear vision of the cognitive process followed along time by students when they face the resolution of a mathematical problem. This makes it necessary to build a framework that emphasizes the dynamic an cyclic nature of the problem solving process for a successful analysis (Wilson et al., 1993).

In that direction, Schoenfeld (1985, 1992), observing the protocols of real solvers, analyzes the behaviors that they develop while solving problems that he calls "nonstandard". These problems are those that the students will not be able to solve by simply recalling and applying familiar solution patterns. In this way, Schoenfeld tries to categorize the behaviors of students and to describe the whole process as a set of *episodes*. An *episode* is "*a period of time during which and individual or a problem solving group is engaged in one large task or a closely related body of tasks in the service of the same goal*" (Schoenfeld, 1985, p. 292).

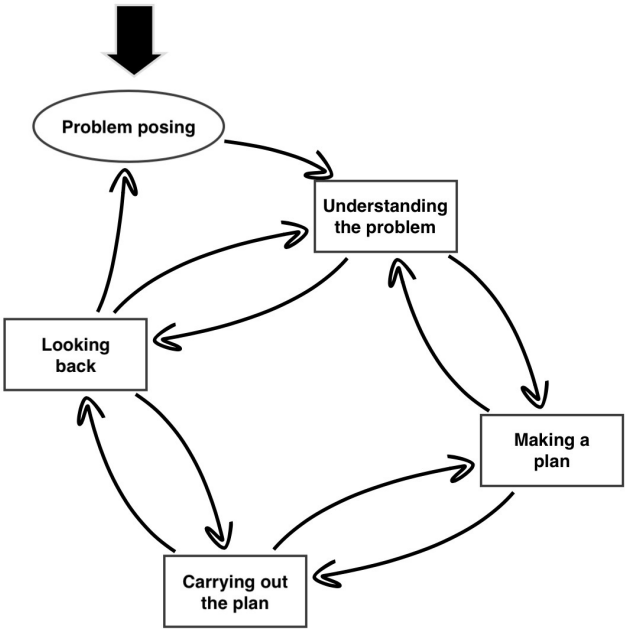

Figure 1. Possible phase changes that can be observed during the problem resolution process as described by Wilson et al. (1993).

By doing this, Schoenfeld develops an action model, and not a competency model, that also identifies the decisions of the solvers and how the resolution process evolves over time. The *episodes* (or categories) identified are: Reading or rereading the problem, analyzing the problem, exploring aspects of the problem, planning all or part of a solution, implementing a plan and verifying a solution.

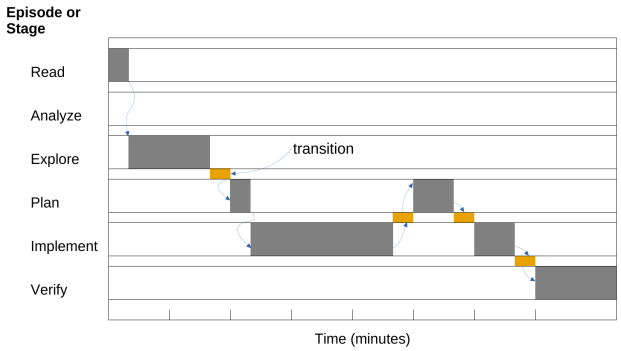

Figure 2. An scheme of the representation of Schoenfeld's *episodes* over time (Schoenfeld, 1985).

The introduction of the variable *time* together with the representation of the *episodes* helps to understand the nonlinear process of problem solving, although this representation must be interpreted depending on the macroscopic analysis

performed (see Figure 2 for an outline of Schoenfeld's idea).

Other authors have used classification into *episodes* of the problem solving process and their representation over time in literature. In this way, Puig (1996), introduces a new type of episode, called *formulation*. During a *formulation* episode what is done is to explicitly formulate a problem that has been generated by some procedure or that has been encountered by other causes.

Also, Goos and Galbraith (1996, p. 241) used "a selective extension of Schoenfeld's episode analysis" to study the nature of individual and interactive strategies used when students work together on solving problems and Scott and Stacey (2000, p. 123) followed Schoenfeld's methodology "as closely as circumstances permitted" to study how students use the problem solving strategies in a problem solving situation.

More recently Brown (2003) performs an analysis of the understanding of functions using graphical calculators using a time-line diagram that records the detail of the *episodes* and shows the progress of the solution as the students proceed. This diagram is also based on the scheme of Schoenfeld and allows the researcher to focus more globally on the *episodes* to find cognitive and metacognitive differences in the resolutions made by different students.

As indicated in the introduction to this work, our tool is based on the scheme developed by Ärlebäck (2009) who developed and used an adapted version of Schoenfeld's "graphs of problem solving" (Figure 2) to get a schematic picture of the problem solving process of students working on a Fermi problem. As defined in Ärlebäck (2009), Fermi problems are "open, non-standard problems that require students to make assumptions about the problem situation and estimates of certain quantities before they engage in, often, a series of simple calculations" and so, they are in line with those used by Schoenfeld (1985) in his work.

Comparing the phases and transitions in the modelling process according to Borromeo Ferri (2006), and the character of a realistic Fermi problem, Ärlebäck identifies the following six modelling sub-activities to be used as codes for the activities the students engage in when solving a Fermi problem.

Reading (R): reading the task and getting an initial understanding of the task.

Making model (M): simplifying and structuring the task and mathematizing.

Estimating (E): making estimates of a quantitative nature.

Calculating (C): doing maths, e.g. performing calculations and rewriting equations, drawing pictures or diagrams.

Validating (V): interpreting, verifying and validating results, calculations and the model itself.

Writing (W): summarizing the findings and results in a report, writing up the solution.

Just like Brown (2003), Ärlebäck (2009) modified and extended the number of Schoenfeld's *episodes* to better suit the characteristics of the problem solving situation studied in order to develop a framework adapted to his work. Ärlebäck uses the term *categories* or *activities* to refer to his extension of the Schoenfeld *episodes*. Figure 3 shows how Ärlebäck's diagrams can look like.

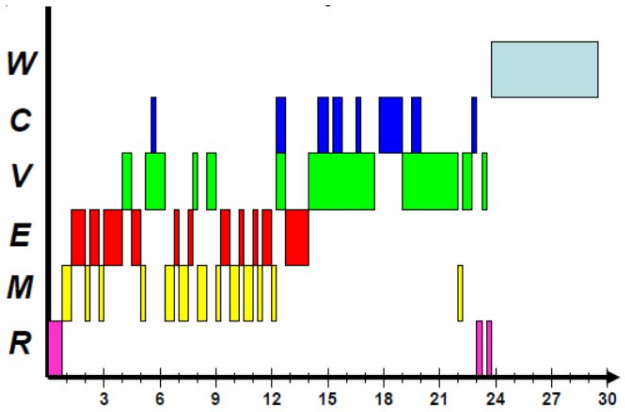

Figure 3. Example of a time-line diagram when students solve a Fermi problem. Extracted from Ärlebäck (2009, p. 345)[1]

More recently, Albarracín, Arleback, Civil, and Gorgorió (2019) further extended Ärlebäck's diagrams representing the sub-problems elaborated by the students to achieve their objective within the activity.

All these contributions show that there has been a considerable effort to build a proper representation capable of recording and representing the complex problem solving process. However, in the different works that have been cited, the authors have done the coding and graphical representation of the diagrams manually. This involves an investment of time and effort to find additional graphical representation tools when analyzing the student's behaviour and its study from an educational point of view.

For this reason, in this article we present a tool that will facilitate the registration of a problem solving session and the publication of the corresponding diagrams. The tool is a computer application which we have called *Task Time Tracker* (**TTT**). Once the *categories* present in the process of solving a problem have been defined, the application is able to create in real time a schema that represents this process over time. In the next section we describe the application and its usage.

3. Task Time Tracker

The **TTT** is a tool to create in real time a time-line diagram that represents the process and steps that are followed when solving a problem. The application has been developed using the programming environment Processing 3[2] (Reas & Fry, 2007) that features a programming language based on Java. A

[1] Reprinted with permission from the Editor of *The Mathematics Enthusiast*
[2] `https://processing.org/`

Processing application typically provides a drawing loop and a series of callback functions that are triggered on user events (e.g. mouse or keyboard events) and internal application events (e.g. the start of the drawing process for the application window).

This architecture has been used to capture user actions and store the beginning of each episode during the problem solving session. The start time and label of the task are stored in an object of the class Table. This object is used to generate the graphical view and to store the information on disk when this is requested by the user.

The **TTT** tool has a Graphical User Interface (GUI) consisting of two panels: an upper panel that shows the graphical representation of the registered *categories* and a lower panel that shows the information about the state of the registering process, together with indications on the available actions at each moment of the session. Figure 4 shows the GUI of the application when it is started. The application is controlled with the keyboard. Table 1 shows the main actions that can be done together with the associated keys.

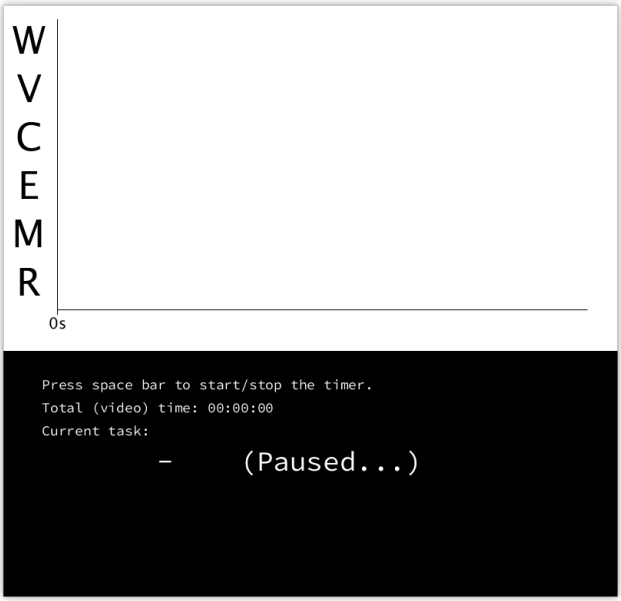

Figure 4. The initial state of **TTT** tool shows an empty time-line canvas and a control panel watinig for the user to start the session.

The application can be in two main states during a tracking session: registering tasks or paused. When it starts, the application is paused and, hence, task tracking is not active. This allows the configuration of the tracking session to be started.

By default, when the application starts, the task *category* labels are configured following the ones proposed by Ärlebäck (2009), RMECVW, as it was discussed in Section 2. However, it is possible to redefine this set of *category* labels before the task tracking is started.

Once the user has started tracking a session, the modification of the list of available labels would require the deletion of the stored information in the memory of the computer. As a consequence, the list of task labels can only be modified before the user has started registering any activity. Beyond this point, the number of different tasks and the associated labels cannot be modified.

The list of task labels can be defined by the user, so that the application can be used with the set of *categories* proposed by any representation. The program will ask for the set of letters to be used and will show in the information panel the typed keys, as shown in Figure 5. Since each tasks is associated to the key of the label during the registration process, keys cannot be repeated.

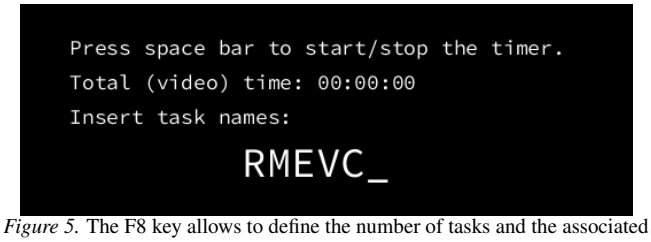

Figure 5. The F8 key allows to define the number of tasks and the associated labels that are going to be used during the tracking session.

In the resulting graphical time-line, the labels are ordered from bottom to top, so that if the labels RMECVW are typed, then 'R' will be at the bottom of the figure, followed by 'M', and so on. An example of a graph produced by **TTT** with Polya's steps with letters CEXM is shown in Figure 6.

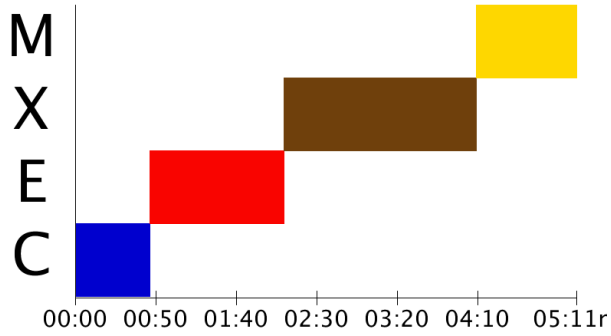

Figure 6. A graph produced by **TTT** tool using the problem solving steps defined by Polya.

Once the set of labels for the tasks have been defined, the user can start the registration of the *categories* present in the problem solving process. The clock that is placed in the lower panel of the application will start running to record and display the session time. During the session, whenever the group of students shifts from one *activity* to another, the key that represents the new *activity* must be pressed. When the key representing the *activity* is pressed, the system will register that moment as the instant of time when a new *category* started. From then on, the bar corresponding to the new *activity* will evolve in the upper graphical panel and the associated letter will be shown in the lower panel until a new key is pressed or the program stops tracking. Figure 7 shows the GUI during a tracking session. In the upper canvas the time-line is displayed,

Table 1
*List of main keys that control the usage of the Time Task Tracker (***TTT***) applications. The symbol ␣ in the first row stands for the Space Bar. The arrows stand for the arrow keys of the keyboard. The arrow keys can only be used when the tracking is paused.*

Key	Function
␣	Start/pause the timer and the registration of tasks
F8	Define the names of the tasks that will be tracked during the session. It can only be done before tracking has started
F9	Create a new session
F10	Load a CSV file containing a previously recorded session
F11	Save a CSV file that stores the activity stored so far
F12	Save a PNG image with the current state of the diagram
→	Advance the session time in 1 second
←	Step back the session time in 1 second
↑	Advance the session time in 10 seconds
↓	Step back the session time in 10 seconds

while the lower control panel shows the session time and the current *activity*.

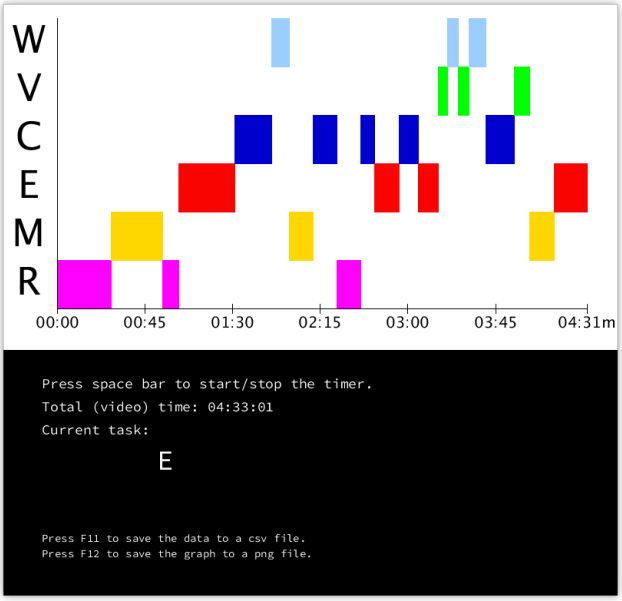

Figure 7. The state of TTT during a tracking session. In the figure, the user has pressed key E and that is the *activity* currently active.

In order to allow the correction of possible mistaken introduction of *activities*, or to make the clock step forward, when the application is paused it is possible to modify the time using the arrow keys. Left and right arrows make the clock advance or step back by one second, while the up and down arrows make the time change in ten second steps. It has to be taken into account that if the time is moved backward to a point before the last *activity* started, this *activity* is removed and the previous one is activated. This can be useful if a new *activity* was activated by mistake or if the *activity* was activated at an incorrect instant of time.

Once a problem solving session has ended, or during the session if necessary, it is possible to save both the upper graph and the data of the current tracking session. The system will ask for a path to save the files. The graph will be saved in PNG (*Portable Network Graphics*) format in the indicated path and the data of the session will be stored in CSV (*Comma Separated Values*) format. The CSV file can be opened from a spreadsheet application, allowing its inspection and analysis.

Moreover, a session that has been saved in a csv file can be loaded by the **TTT** application in order to resume the session at the point it was left. This allows the registration of a problem solving session in several video visualization tracking sessions.

4. Conclusions

In this paper we have presented the **TTT**, a tool that helps researchers in the task of creating a useful representation of the problem solving process. The application creates in real time a time-line diagram that represents the process and steps followed when a student or group of students face the resolution of a complex problem. The tool has been developed using the programming environment Processing and provides a drawing loop and a series of callback functions that captures user events. Both graphical and text data are produced when tracking a problem solving working session and automatizes the previous work that has to be done in a macroscopic analysis of the students processes.

Acknowledgements

Special thanks to L. Albarracín and I. Ferrando for their contributions and ideas in the design and usability of the tool.

References

Albarracín, L., Arleback, J., Civil, E., & Gorgorió, N. (2019). Extending modelling activity diagrams as a tool to char-

acterise mathematical modelling processes. *The Mathematics Enthusiast*, *16*(1), 211–230.

Ärlebäck, J. B. (2009). On the use of realistic fermi problems in introducing mathematical modelling in upper secondary mathematics. *The Mathematics Enthusiast*, *6*(3), 331-364.

Berry, J., & Sahlberg, P. (2006). Accountability affects the use of small group learning in school mathematics. *Nordic Studies in Mathematics Education*, *11*(1), 5–31.

Borromeo Ferri, R. (2006). Theoretical and empirical differentiations of phases in the modelling process. *ZDM*, *38*(2), 86–95.

Brandell, J. R. (2010). *Theory & practice in clinical social work*. Sage.

Bransford, J. D., & Stein, B. S. (1984). *The ideal problem solver. a guide for improving thinking, learning, and creativity*. Freeman.

Brown, J. P. (2003). An insight into student understanding of functions in a graphing calculator environment [Master's thesis]. *Department of Education, The University of Melbourne*.

Gillies, R. M. (2003). Structuring cooperative group work in classrooms. *International Journal of Educational Research*, *39*(1), 35–49.

Goos, M., & Galbraith, P. (1996). Do it this way! metacognitive strategies in collaborative mathematical problem solving. *Educational studies in mathematics*, *30*(3), 229–260.

Pearl, J. (1984). *Heuristics: intelligent search strategies for computer problem solving*. Addison-Wesley Pub. Co., Inc., Reading, MA.

Pólya, G. (1945). *How to solve it; a new aspect of mathematical method*. US: Princeton University Press.

Puig, L. (1996). *Elementos de resolución problemas*. Comares.

Reas, C., & Fry, B. (2007). *Processing: A programming handbook for visual designers and artists*. MIT Press.

Schoenfeld, A. H. (1985). *Mathematical problem solving*. Academic Press: Orlando, FL.

Schoenfeld, A. H. (1992). On paradigms and methods: What do you do when the ones you know don't do what you want them to? issues in the analysis of data in the form of videotapes. *Journal of the Learning Sciences*, *2*(2), 179-214. doi: DOI *Pendiente de asignación*10.1207/s15327809jls0202_3

Scott, N., & Stacey, K. (2000). *Orientation to deep structure when trying examples: a key to successful problem solving*. Hergué.

Vygotsky, L. S. (1980). *Mind in society: The development of higher psychological processes*. Harvard university press.

Wilson, J. W., Fernandez, M. L., & Hadaway, N. (1993). Mathematical problem solving. *Research ideas for the classroom: High school mathematics*, *57*, 78.