IceTT: A Responsive Visualization for Task Models

Lucio Davide Spano
Department of Mathematics and Computer Science, University of Cagliari
Via Ospedale 72, 09124, Cagliari, Italy
davide.spano@unica.it

Gianni Fenu
Department of Mathematics and Computer Science, University of Cagliari
Via Ospedale 72, 09124, Cagliari, Italy
fenu@unica.it

ABSTRACT
Task models are useful for designers and domain experts in order to describe sequences of actions that need to be completed for reaching a user’s goal. Their hierarchical structure is usually visualized through a tree representation that, for large models, is inclined to grow horizontally and reduces its readability. In this paper we introduce a visualization based on icicle graphs, which is able to adapt the tasks visualization to the screen width, suitable for displaying large models even on small screens.

Author Keywords
Task models; Visualization; Responsive Design;

ACM Classification Keywords
H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

INTRODUCTION AND MOTIVATION
Task models describe the sequence of actions that users need to complete in order to reach a specific goal. Given their simple and intuitive representation, task models have been used in multidisciplinary teams for communication purposes and in order to persist a shared view of an interactive system. The most common graphical representation of a task model is a tree: the root node represents the entire application, which is gradually detailed through its descendants, usually connected through a set of temporal operators (sequence, choice, concurrency etc.). Since the number of sub-tasks can be high in real world applications, task models often grow on the horizontal axis. In such cases, the model is difficult to read and understand, since designer has to move the model visualization back and forth through the horizontal scrolling.

In this work we propose IceTT, a responsive visualization of task models, which is able to adapt to a given window width, exploiting effectively the available space for including the model information. Such kind of visualization allows using different mobile devices, such as smartphones and tablets, in order to share and/or discuss task models. We provide a preliminary comparison between our visualization, CTTE [6], Hamsters [2] and K-MADe [3] on the same set of models, showing that IceTT is able to show more information on different small and medium standard screen resolutions.

RELATED WORK
Task models are usually built starting from a high-level description of the goal to accomplish (e.g. print a document), which is refined through different levels into a set of detailed actions. Such hierarchical description is suitable for being represented graphically with a tree. However, since task models describe not only the hierarchy among actions but also their temporal relationships, different techniques have been adopted for representing these aspects. In order to compare the proposed solution, we selected three public available representatives of three approaches, focusing on tools specifically designed for task modelling. We did not consider here more generic workflow or application flow visualizations (e.g. BPMN, Petri Nets, flow charts etc.).

The first group includes both the task description and the temporal relationships information inside the node representation, following the Hierarchical Task Analysis (HTA) [1] approach. This representation does not introduce additional nodes for the temporal operators, but the representation of the tasks needs more space. In this paper, the K-MADe [3] tool represents this group.

The second group we consider represents the temporal operators as nodes, including them as task siblings in the tree visualization. This group is represented by CTTE [6]. This kind of representation grows horizontally, since for task each sub-task we add to a given level, we have to add two nodes in the tree.

The third group represents the temporal operators through dedicated nodes, but they are on different levels with respect to the connected tasks. In this case, the tree is a bipartite graph, where tasks are always connected with operators and vice versa. In this case, the model grows vertically, since for refining a task, we need to add two depth levels to the tree representation. In this paper, the Hamster [2] tool represents this group.

THE PROPOSED VISUALIZATION
The IceTT task visualization has been designed in order to exploit the entire screen width without exceeding it.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the Owner/Author.

Copyright is held by the owner/author(s).

EICS’14, Jun 17-20 2014, Rome, Italy
ACM 978-1-4503-2725-1/14/06.
http://dx.doi.org/10.1145/2607023.2611452
At the intermediate levels, the task visualization has different levels of detail, in order to adapt the displayed information to the width available. We selected an icicle plot visualization [5], with a vertical layout for fulfilling these requirements. It consists in a table-like hierarchical representation of the task model tree structure. We can summarize its properties as follows:

1. Each row in the diagram corresponds to an entire level of depth in the task tree. Therefore, all nodes at the same depth are painted in the same row.
2. The representation of a task node has the same height for all levels.
3. The root node fills the entire screen width.
4. The width allocated for a given task node is divided among its children in the next level.

The properties 3 and 4 guarantee that the task model visualization fills the entire window (or container width). The properties 1 and 2 have an impact on the model visualization height, which depends linearly on the task tree maximum depth, as happens for other tree-like representations. We created the visualization extending the Javascript Infovis Toolkit (JIT) in [4].

Figure 1 shows the IceTT visualization of a task model representing the interaction with a mobile phone. On the left part, the model is rendered in a window 800 pixels wide. In the right part, the same model is shown with a width of 320 pixels. As it is possible to see from the figure 1, the visualization does not exceed the window width. We differentiated the task categories [7] through both icons and colors:

- The Abstract tasks (used for grouping together task of different categories at the intermediate levels) are identified by a cloud icon and a sky-blue background.
- The Interactive tasks (that involve both the system and the user) are identified by a human and a computer icon connected with an arrow. The background color is green.
- The System tasks (that involve only the system) are identified by a computer icon and a red background.
- the User tasks (that involve only the human user) are identified by a human icon and an dark-yellow background.

The composition operators are represented using the symbols defined in [7]. They express the temporal relationships between the different tasks (≫ for sequential enabling, [] for choice, ||| for concurrency, |= for order independence, > for disabling etc.). The tool allows changing the color palette for solving accessibility issues.

In order to effectively distribute the space among tasks at the same depth level, the available space is divided among the children considering the number of leafs of their sub-trees. In this layout computation, we considered as children also the temporal operators, since we put their graphical representation between a task and its sibling, as happens in CTTE [6]. More precisely we define a weight $w(n)$ for each node of the icicle graph (both tasks and temporal operators) in equation 1. $C_n$ contains the children of a node $n$, while $deg(n)$ is the node degree.

$$w(n) = \begin{cases} 
1 & \text{if } deg(n) = 0 \\
\sum_{m \in C_n} w(m) & \text{if } deg(n) > 0
\end{cases}$$

For instance, if we consider again the Use Phone task in figure 1, the layout function divides its space among five graphical elements: three tasks and two temporal operators. The first task (a user task represented with a dark yellow rectangle) and the two operators are leafs, therefore their weight is one. The Make Call and the Use other functions are the root of two different sub-trees, having respectively 16 and 5 leaf nodes. Therefore, the...
**Make Call** task is wider than the other task and temporal operator representations: 16 times more than the sibling user task and about 3 times more than the **Use other functions** abstract task.

Such organization of the space in the horizontal axis provides less space for representing tasks that are more deep in the hierarchy. Therefore, we used an incremental visualization of the task-related information, according to the available space. For each task, the visualized information in all tools is usually the category and the task name. The composition operators are usually represented through their symbols.

If we have very little space in width (less than 16 pixels), we show only a coloured rectangle for each task. This already provides information on the task existence and on its category. At higher widths, we show also the task name and icon. The visualization truncates the name if the available space is not enough for containing it entirely. In this way, even if the whole information on the task name is not present, the designer may recall it through its initial part. Finally, if there is enough space, the task name is visualized entirely and the icon is bigger.

For the composition operators we decrease the font size in order to draw its symbol in the available space. It the available space is not enough for displaying the symbol at the minimum font size (4pt), the operator is not shown.

Figure 1 shows an example of this incremental visualization technique. The **MobilePhone** task is the tree root, and it fits the entire visualization width. For this task, there is enough space and the name is visualized entirely, together with the icon and color assigned to the abstract category. For some tasks, such as the **Connect** children, it is not possible to include the entire name. The tool visualizes only the initial part (e.g. “Enter” for **Enter Pin**). For other tasks, the space is not enough even for displaying a small part of the task name. Therefore, the tool shows only a coloured rectangle indicating the presence of a task and its category. For instance, from the visualization in figure 1, we know that the **Number from List** abstract task has four children, three interactive tasks and one system task. However, we have no information on their names.

The composition operators are visualized using different font sizes. At the higher levels they are bigger and more readable, while at lower levels they are more compact and less distinguishable. In figure 1, the enabling symbol (≫) between the **Connect** and **Use Phone** task is more readable that the one between the children of the **Number from List** task.

Such layout technique has the drawback of hiding information for tasks at the lower levels, since the space is decomposed in width through the depth levels. Therefore, we added the possibility to select an intermediate task in the model and to set it as the root of the visualization, for focusing on a specific sub-tree. For instance, if we start from from the left part of figure 1, the user may click on the **Select Number** task. The iccTT tool enlarges its visualization and, through an animated transition, sets the selected task as the root of the icicle plot. The resulting visualization is shown in figure 2. In order to going back to the parent task, the tool interface provides a level up button that can be used for navigating the task tree.

**COMPARISON WITH EXISTING TOOLS**

In this section, we provide a preliminary comparison between our tool and the visualization of three publicly available environments for modelling tasks: CTTE [6], Hamsters [2] and K-MADe [3], as motivated in the related work section. For the comparison, we considered a set of examples shipped with CTTE (we did not consider collaborative task models), which consists of six models. We re-created all models with all tools. In order to compare the different visualizations we defined a score for quantitatively summarize the visualized information. For each task, we assigned one point if the category is visible, one point if the task name is visualized (or partially visualized) and one point for each readable temporal operator. Such information is shown by all considered tools for each task. The set of model is summarised in table 1. The “Task” column corresponds to the number of tasks in the model, the “Dpt” column to the maximum depth of the task tree, while “Pts” it the maximum visualization score.

We considered three different standard screen resolutions: QVGA (320x240), VGA (640x480), XGA (1024x768), aligning the root task to the top of the screen and centering it horizontally for calculating the scores. For each model and for each tool, we show in figure 3 the percentage of the model information shown at the considered resolution. The data shows that the proposed visualization is able to effectively exploit the screen area. With

<table>
<thead>
<tr>
<th>Model Description</th>
<th>Task</th>
<th>Dpt</th>
<th>Pts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access to student data</td>
<td>7</td>
<td>4</td>
<td>17</td>
</tr>
<tr>
<td>Content Management System</td>
<td>14</td>
<td>4</td>
<td>35</td>
</tr>
<tr>
<td>Virtual Museum</td>
<td>14</td>
<td>4</td>
<td>37</td>
</tr>
<tr>
<td>Automated Teller Machine</td>
<td>16</td>
<td>4</td>
<td>44</td>
</tr>
<tr>
<td>Mobile Phone</td>
<td>28</td>
<td>7</td>
<td>74</td>
</tr>
<tr>
<td>Nomadic Application</td>
<td>47</td>
<td>7</td>
<td>120</td>
</tr>
</tbody>
</table>

Table 1. Task models for the visualization comparison
small models it is able to visualize the entire model information already at low resolutions. With larger models, IceTT is able to visualize more information with respect to the other tools for all considered examples. In addition, IceTT consistently increases this advantage passing to higher resolutions. If we consider the QVGA resolution, IceTT is able to show the higher number of model attributes in all cases, except for the Student model, where CTTE performed better considering that it is particularly small. IceTT, Hamsters and K-MADe were able to maintain roughly constant the information visualized for all models. The percentages decrease with the growth of the models. In CTTE, the visualized information decreases with the growth of the model because of the horizontal expansion we discussed in the related work section. In particular for the Nomadic model, it was able to show only the root task. Passing from QVGA to VGA, IceTT is able to show all information for all models, except for the two largest models, outperforming the other tools on the entire test set. At this resolution, CTTE and Hamsters have a comparable performance, but CTTE is more efficient for the Museum and the ATM models.

Considering the XGA resolution, all tools are able to show most of the information for the first four models. K-MADe has a performance comparable with CTTE and Hamsters. IceTT is again able to consistently show more information with respect to the other tools. In particular, considering the largest model (Nomadic), it is possible to see that only IceTT and K-MADe were able to increase the visualized information with respect to the VGA resolution, while for CTTE and Hamsters the difference is not relevant. However, IceTT was able to show the 66% of the information, while K-MADe only the 22%.

Considering this preliminary results, we can conclude that the proposed visualization has a good potential for representing task models of different sizes even on small screen resolutions. In addition, passing to medium and large screen resolutions, the visualization is able to exploit the space effectively for showing more information.

**CONCLUSION AND FUTURE WORK**

In this paper we introduced a task model visualization based on icicle plots [5]. The visualization is able to exploit the screen width and to adapt the visualization in order to maintain the model readable, even with low resolutions. We reported on a preliminary comparison against three existing tools obtaining encouraging results. In the future, we aim to validate the visualization acceptance with the different stakeholders involved in task modelling (e.g. designers, domain experts etc.). In addition, we aim to extend the proposed approach also to other hierarchical models for user interface definition such as gestures and dialog models.

**ACKNOWLEDGMENTS**

We gratefully acknowledge Sardinia Regional Government for the financial sup-port (P.O.R. Sardegna F.S.E. Operati-