Supporting Semi-Structured Work using Higher Order Tasks

[Extended Abstract]

Bas Lijnse
b.lijnse@cs.ru.nl

Jan Martin Jansen
jm.jansen.04@nlda.nl

In today's society many kinds of work are supported by computer systems in some way or another. How and to what extent depends of course on the type of work. Some jobs are almost completely automated leaving the human in the loop only for monitoring and control purposes. Other types are primarily done without computer support, but computers may still be used for administrative and communication tasks. In general, one can view the extent to which work is supported by computer systems a spectrum in which on the one end work is completely automated, and on the other end work is performed entirely without computer support. The kind of work support systems at both ends of the spectrum are very different. On the highly automated side we find large integrated systems which embody much knowledge about the work that are capital investments for organizations. On the other side of the spectrum we find small "apps" for specific tasks and general purpose office and communication tools that aim to make overhead tasks, the so called paperwork, more efficient such that we have more time for our "real" work. For both ends of the spectrum are many options for support. Either tools and frameworks to build integrated work support systems , or many off-the-shelf apps to choose from to support unstructured types of work. The middle of the spectrum is less densely populated. Scripting languages, plugin mechanisms and applications with API's typically occupy this space. However, for the types of work that are too unstructured to warrant the investment in integrated systems, but are not so irregular that only task specific applications the possibilities for computer support are often sub-optimal.

In this paper we presents a framework that aims to make the development of work support systems that target the middle of the spectrum, semi-structurd work, easier. This framework is a library for the iTask System that enables the definition of tasks that are loosely structured on a high level, but structured on a low level. This is done by providing hierarchical todo lists on which the items are interactive sub task programs. Items can be added at will, but suggested items are computed continously to assist users.

The iTask System itself is a framework that facilitates Task-Oriented Programming (TOP) in the functional programming language Clean. In Task-Oriented Programming, interactive systems are programmed by defining compositions of subprograms that support a self-contained task. These task specifications abstract from many implementation details, allowing programmers to focus on the task they aim to support with their programs. Task compositions can either be rigidly structured, loosely structured or a mixture of both. In this paper we focus on the latter.

The enabling factor underlying our framework is the possibility of defining higher order tasks. Because TOP is grounded in functional programming, and task specifications are *just* functions, we can create task specifications that are parametrized with other task specifications, making it possible to define general frameworks such as this one.

The framework was originally developed for an iTask application that supports Coast guard officers. Their work is semi-structured yet guided by standard operating procedures. However, the framework's applications are more general as we show with an example using it to support the organization of small events.

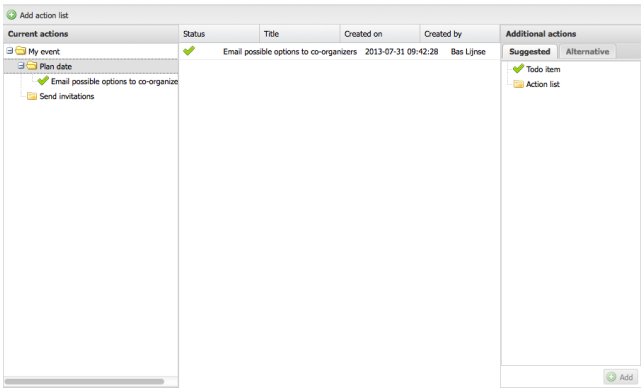

Figure 1: Screenshot of an example application using the framework