



institut für informatik



*Termexec* Component

# Termination Competition

Simon Bailey  
simon.bailey@uibk.ac.at

6<sup>th</sup> April 2009

## Version:

Id: content.tex 66 2009-04-01 15:19:23Z binabik

## Abstract

This document outlines the proposed functionality of the *termexec* component for the *termcomp* platform. *termexec* will offer users the possibility to submit experiments to the competition platform with a set of tools and problems they can determine themselves. *termexec* will implement scheduling algorithms to allow multi-user usage and to prevent experiments from interfering with the functionality of the *termcomp* platform.

# Contents

|          |  |          |
|----------|--|----------|
| <b>1</b> | <b>What is termexec?</b>                   | <b>1</b> |
| 1.1      | Terminology . . . . .                      | 1        |
| <b>2</b> | <b>Initial Component Structure</b>         | <b>1</b> |
| 2.1      | Software environment . . . . .             | 1        |
| <b>3</b> | <b>User Interface</b>                      | <b>2</b> |
| 3.1      | Selecting Tools . . . . .                  | 2        |
| 3.2      | Selecting Problems . . . . .               | 2        |
| 3.3      | Time-outs . . . . .                        | 2        |
| 3.4      | Comments . . . . .                         | 2        |
| 3.5      | Publicising Results . . . . .              | 2        |
| 3.6      | Superseding Previous Experiments . . . . . | 3        |
| <b>4</b> | <b>Admin Interface</b>                     | <b>3</b> |
| 4.1      | Time-outs . . . . .                        | 3        |
| 4.2      | Manage quotas . . . . .                    | 3        |
| 4.3      | Slice size . . . . .                       | 4        |
| 4.4      | Manage queue . . . . .                     | 4        |
| <b>5</b> | <b>Scheduler</b>                           | <b>4</b> |
| <b>6</b> | <b>Implementation</b>                      | <b>4</b> |
| 6.1      | Pre-requisites . . . . .                   | 4        |
| 6.2      | Preliminary schedule . . . . .             | 5        |
|          | <b>Bibliography</b>                        | <b>7</b> |

# 1 What is termexec?

There have been numerous requests in the course of the competition history to be able to compare different tools on reference hardware. This will enable experimental results published in papers to be easily comparable and should establish a standard amongst the termination community.

*termexec* will be an extension to the existing *termcomp* infrastructure allowing users to create custom experiments granting them exclusive run-time access to the reference hardware also used by the competition. In order to allow multiple users access, the existing scheduler will be adapted to support this usage scenario, thus guaranteeing fair access for all users.

## 1.1 Terminology

**experiment** An experiment is the complete set of tools and problems submitted to *termexec*.

**user** A human agent registered with the *termcomp* platform.

**team** A grouping of users within the *termcomp* platform with some shared rights.

**visitor** A human agent not registered with the *termcomp* platform who only has access to publicly available content.

**problem** The grouping of a single tool and a single input TRS.

**TPDB** The Termination Problem Database.

**XTC** The new XML based format proposed for the TPDB.

**tool** A tool is a software product submitted by a team.

**implementation** An implementation is the representation of a specific version of a tool.

## 2 Initial Component Structure

*termexec* will consist of three main components which will be integrated into the *termcomp* infrastructure:

- User Interface
- Admin Interface
- Scheduler

The first two components will be integrated into the currently existing web application and communicate with the scheduler. The scheduler will extend the functionality of the relatively simple scheduler which is currently used in *termcomp*.

### 2.1 Software environment

As this component will be integrated into *termcomp*, we will continue to use the JBoss Seam [1] application framework, which has matured considerably since implementation of *termcomp* was first started. The scheduler for *termcomp* is currently implemented with the Quartz [2] scheduling framework which has

some limitations—it is basically a time-based scheduling framework with no built-in support for batch processing. It should be possible to extend it for the functionality required by *termexec*, however we may have to investigate alternate solutions.

### 3 User Interface

The user interface of *termexec* should offer the following functionality:

- Select tools for the experiment
- Select problems
- Set time-out for proofs and certification
- Add comments to the results of the experiment
- Publicise results on demand
- Supersede previous experiments
- Visitor interface.

Further explanation of these items is offered below.

#### 3.1 Selecting Tools

A user wishing to submit an experiment will be able to select any version of any tool submitted to *termcomp* for comparison in his experiment. A user may wish to compare different versions of his own tool or compare the current version of his tool to the current or any historical version of a different tool. *termexec* will support this.

#### 3.2 Selecting Problems

The user will be able to search the TPDB based on certain criteria allowing him to create sets of problems for his experiment. Functionality for storing these searches will also be offered.

#### 3.3 Time-outs

The administration of *termexec* will set sensible default and maximum time-outs; a user submitting an experiment will be able to set his desired time-out between 1 second and the maximum as set by the administration.

Users will also be able to set a global time-out for the complete experiment so that the experiment is terminated after a fixed amount of computation time.

#### 3.4 Comments

When an experiment is first submitted, the user will be able to add free-form text to a comment field which will subsequently be displayed with the results of the experiment. Users will also be able to add comments to previously completed experiments.

### 3.5 Publishing Results

The results of an experiment are per default private and can only be seen by the user who submitted the experiment and members of the team he belongs to. If the user wishes, the results will be made public and will be visible to any visitor to *termcomp*. In order for the results of experiments to be more easily accessible, a URL schema will be implemented which will allow easy access to results.

### 3.6 Superseding Previous Experiments

As most termination provers are developed by humans, they are prone to errors which may or may not be found immediately. This functionality will offer users the possibility to mark certain experiments as superseding previously submitted experiments. If a visitor then accesses a superseded experiment, he will be informed of the fact by the *termexec* software and offered a link to jump to the most current experiment.

### 3.7 Visitor Interface

A public visitor interface will be available, enabling visitors to view the results of public experiments.

## 4 Admin Interface

The administration interface will offer functionality for setting certain default values and controlling the execution of experiments. The major functionality will cover the following:

- Set maximum available time-out
- Manage quotas
- Set slice size
- Manage queue

### 4.1 Time-outs

As the time-out used in the competition may not be sufficient for some tools to solve certain termination problems, the *termexec* platform will offer tool authors the possibility to execute proofs with longer time-outs. The administration will decide on a sensible maximum time-out and set it in this part of the administration interface. This will allow the maximum time-out to be changed on demand rather than hard-coding it into the *termexec* platform.

### 4.2 Manage quotas

As yet, there is still no consensus on how to implement the quota system. There are two viable possibilities which could be implemented:

- Time-based quota

- Problem-based quota

Using a time-based quota would enable the *termexec* platform to track the exact amount of system time used by a team for their experiments; if a tool is sufficiently fast, more experiments can be conducted with the allotted time. A problem-based quota, limiting the number of problems that can be included in experiments conducted by a team, may be of advantage to slower tools.

### 4.3 Slice size

In order to allow fair execution of experiments, each experiment submitted will be split into slices of a given size. Depending on the utilisation of *termexec*, the administration will be able to set the number of problems included in a slice. When an experiment is submitted, the slices are added to the queue and the queue will intelligently perform round-robin execution of the slices. This will also enable test-runs from the *termcomp* platform to be interleaved into the queue and executed after the current slice is finished; thus not forcing users to wait until a complete experiment has been completed before their tool is tested.

A suggestion made was to make the slice size time-dependent; this means the administration would set a maximum time a slice is allowed to take and depending on the time-out selected by the user, the number of problems contained in a slice will be determined dynamically.

### 4.4 Manage queue

This option will allow the administration to manage the jobs submitted to the scheduler. Detailed functionality still remains to be discussed, but the base functionality will allow the administration to pause or cancel complete experiments.

## 5 Scheduler

The existing scheduler used in *termcomp* will be extended by the following functionality; however, some of the points still require discussion and consensus, these are marked so: (\*).

- Support slices
- Prioritise queue: (\*)
  - FIFO?
  - Test-runs interleaved with experiment slices
  - Prioritise certain experiments
- Email notifications
- Web status panel
- Parallel execution of single-core jobs (\*)
- Before submitting, calculate if the maximum execution time will complete before the next competition starts

## 6 Implementation

### 6.1 Pre-requisites

Some of the components in *termexec* will also be affected by the implementation of *termexec*. Notably, these will include:

- Results display
- Problem search
- Convert the TPDB to the new XTC format

The results display will be improved and the underlying components will be re-factored to better support *termexec*'s functionality. With the TPDB converted to an XML-based format, the problem search facility will become easier to use; this is a requirement for *termexec* so that users can easily search for problems to use in their experiment.

### 6.2 Preliminary schedule

Programming will commence on the 6<sup>th</sup> of April 2009; we expect the effort to last about 4 weeks and a first prototype will be available for testing on May 4<sup>th</sup> 2009.



## References

- [1] Gavin King. Seam framework. Available from World Wide Web: <http://seamframework.org/>.
- [2] OpenSymphony. Quartz framework. Available from World Wide Web: <http://www.opensymphony.com/>.