Universität Zürich

Institut für Informatik

Binzmühlestr. 14
CH-8050 Zürich
Schweiz
Tel. +41 044 635 43 33
Fax +41 044 635 68 09
boehlen@ifi.uzh.ch
www.ifi.uzh.ch/dbtg

Prof. Dr. Michael Böhlen
Database Technology

Bachelorarbeit

Robert Dewor

Zürich, 30. August 2011

Design and Implementation of a workload generator for Oshiya

In the proposed project, an existing demo application for the *Oshiya* scheduling model shall be extended by at least the features listed below. These features shall be designed and implemented in a way that they emphasize the characteristics and advantages of Oshiya. This task includes requirements analysis, conceptual design, implementation and testing.

- A workload handler that is able to configure the incoming requests which have to be scheduled. This includes, e.g., the amount of clients and the type of transactions the clients execute (e.g. read-only).
- A statistics engine that collects runtime information such as the cardinalities of the scheduling relations or the execution times of the scheduling queries. This information has to be visualized in order to let the user understand the behavior of the application in a better way.
- A correctness checker that checks during runtime whether the protocol specific constraints hold for the generated history. The protocol-specific constraints have to be developed for the existing S2PL and SI implementations.
- A logger that records and displays status messages.

Background:

Modern systems have to schedule huge amounts of concurrent client requests efficiently while guaranteeing that the produced schedules fulfill certain correctness criteria and/or service-level agreements. The state of the art is to develop schedulers imperatively for a given application, which yields very complex scheduler implementations and inflexibility to adapt to changing scheduling requirements.

In Oshiya, the scheduling state is stored in *scheduling relations*. A protocol is formalized as a set of constraints called *protocol specification*. A *protocol specification* is implemented by three *scheduling queries*. And request scheduling is performed by repeatedly executing the *scheduling queries* over the *scheduling relations*.

Tasks:

- Literature research on declarative scheduling [1, 3, 2]
- Based on the existing basic version of the demo application, requirements analysis and conceptual design (towards high modularity) shall be done for the new features including at least a workload handler, a statistics engine, protocol-dependent correctness checkers and a logger.
- Finish implementation of the Oshiya scheduling algoritm
- Implementation and testing of the new features (leveraging our declarative 2PL and SI implementations)
- Writing a bachelor thesis presenting and discussing your results
- Presentation of results (15 minutes)

Requirements:

- Experience in programming Java (incl. thread programming), Java Swing
- Knowledge in concurrency control, transaction processing

Aim:

- A running demo application executing the Oshiya scheduling algorithm stepwise forward and backward and providing a workload handler, a statistics engine, protocol-dependent correctness checkers and a logger.

Conditions:

- A project meeting will take place periodically (typically every two weeks). At least 24 hours prior to a project meeting a report has to be sent to the supervisor including (a) the progress since the last meeting, (b) planned next steps, (c) problems to discuss and (d) ideas to solve those problems.

Literatur

[1] C. Tilgner. Declarative Scheduling in Highly Scalable Systems. In *Proceedings of the 2010 EDBT/ICDT Workshops*, EDBT '10, pages 41:1–41:6, 2010.

[2] C. Tilgner et al. Declarative Serializable Snapshot Isolation. In *ADBIS*, 2011.

[3] C. Tilgner et al. Smile: Enabling Easy and Fast Development of Domain-Specific Scheduling Protocols. In *BNCOD Posters*, 2011.

Task assignment and Supervisor: Christian Tilgner

Starting date: 25.07.2011 Ending date: 24.01.2012

Department of Informatics, University of Zurich

Prof. Dr. Michael Böhlen