**Report:**

**Thread mentor : multithread processing**

Name: Hammad hussain (59490)

Rija Fatima (59631)

Areeba

**Abstract:**

Thread Mentor is a multiplatform pedagogical tool designed to ease the difficulty in teaching and learning multithreaded programming. It consists of a C++ class library and a visualization system. The class library supports many thread management functions and synchronization primitives in an object-oriented way, and the visualization system is activated automatically by a user program and shows the inner working of every thread and every synchronization primitive on-the-fly. Events can also be saved for playback. In this way, students will be able to visualize the dynamic behavior of a threaded program and the interaction among threads and synchronization primitives.

**INTRODUCTION:**

The capability of multithreaded programming was first commercially available in the late 60s when IBM added the task feature and completion event variables into its PL/I F compiler and made the same available in all IBM VS (i.e., virtual storage) operating systems as supervisor calls. Threading became popular in the Unix community in the early 80s. Today, virtually all operating systems have multithreaded capability, and the POSIX P threads standard is also popular. Many well-known operating systems textbooks added sections on threads in their newest editions] and numerous books about threads were published in recent years.

**SYSTEM OVERVIEW:**

The most important class in the class library is class Thread. A student defines a thread as a derived class of Thread and supplies a method Thread Func() as the thread body. Class Thread includes methods Begin() for executing a created thread; Exit() for terminating a running thread; Join() for joining with another thread; Yield() for relinquishing the execution to another thread;

**CONCLUSIONS:**

We have presented a detailed overview of the class library and visualization of Thread Mentor. Thread Mentor was used twice in the programming track of our “Introduction to Operating Systems” course [Shane 1998, 2002] to replace Sun Solaris threads, and at three workshops [Carr et al. 2001, 2002, 2003]. It was also site-tested at a number of schools. Reactions from site testers and participants of our workshops were very positive and encouraging. Typical comments include “the visualization is excellent” and “[Thread Mentor] is a useful tool for OS classes.” In the attitude surveys that were conducted at the end of the above-mentioned course, students indicated overwhelmingly that the visualization system of Thread Mentor “helps pinpoint errors quickly” and “helps to see what is happening with the threads.” The complete system is “wonderful,” “ease of use and straightforward,” “a good learning tool and very handy” and “[taking] a lot of trouble out of using threads,” and has “easy semantics and calling convention” and “a common interface between Linux and Solaris [and Windows].” Two students indicated that they never used the visualization because it is not their programming style. There were only a few negative comments, most due to unfamiliarity with the system. For example, a handful of students indicated that Thread Mentor does not perform properly on a remote machine (e.g., remote login). Since the GUI of the visualization system must transmit a large amount of graphical information, running Thread Mentor on a remote machine and displaying the windows locally cannot be very efficient. Thus, Thread Mentor is designed to run on a local machine, although it is possible to execute Thread Mentor on a remote machine. A few students mentioned that Thread Mentor’s behavior could be different on different machines and on different operating systems. This is normal, since the behavior of a threaded program cannot be identical across platforms. It is also interesting to mention that two students, one per year, criticized Thread Mentor for being proprietary and not used in the “real world.” They preferred P threads or Sun Solaris threads over Thread Mentor. Since the number of negative comments were very few, we believe that Thread Mentor is currently a reasonably mature system. A detailed analysis of the effectiveness of using Thread Mentor in the above-mentioned course with pre- and posttests will be published elsewhere.