**Packed Object Synchronization**

**Bing Yang** (UNB),
**Dr. Kenneth Kent** (UNB), **Dr. Eric Aubanel** (UNB), **Karl Taylor** (IBM)

University of New Brunswick, IBM Canada
Faculty of Computer Science

Bing.Yang@unb.ca, ken@unb.ca, aubanel@unb.ca, Karl_Taylor@ca.ibm.com

**Introduction**

Many Java applications and frameworks must deal directly with native data. Packed Objects provide an approach for users to deal with native data in a more convenient way.

Unlike the standard Java data model, the Packed Object data model uses the space more efficiently and reduces the overhead of each object.

However, due to this change in the object model, certain Java built-in mechanisms have to adapt to the new model. Since multiple packed objects could potentially refer to the same underlying data, and packed objects may have to deal with data shared with non-java code, a new approach to synchronize on Packed Objects is needed.

**Research Goal and Approach**

The primary research goal is to build a framework for providing flexible and scalable synchronization that could be used for concurrent programing of Java packed objects as well as non-java native code. In the current phase, our main focus is to make use of existing data structures and implement synchronization methods for packed objects and provide interfaces for users in Java and native code applications.

**Packed Object Data Model**

- **Packed object data model** is more compact.
- **Instance field of packed object** is inlined in the containing packed object.
- **Multiple packed objects** may refer to the same underlying data.
- **Users have more control** over the layout of Java objects.
- **Packed objects avoid copying and marshalling native data structure** into Java heap for manipulation.

**Research Goal and Approach**

- **Packed objects do not contain a Lock-word field.**
- **Only the object being synchronized** would be mapped to a monitor structure.
- **The address of the packed object data** would be the key to map with a monitor.
- **An interface method would be explicitly invoked** to acquire locking on the associated monitor before accessing the critical region of the packed object.
- **An interface method would be explicitly invoked** to release locking on the monitor after the thread finished its work.