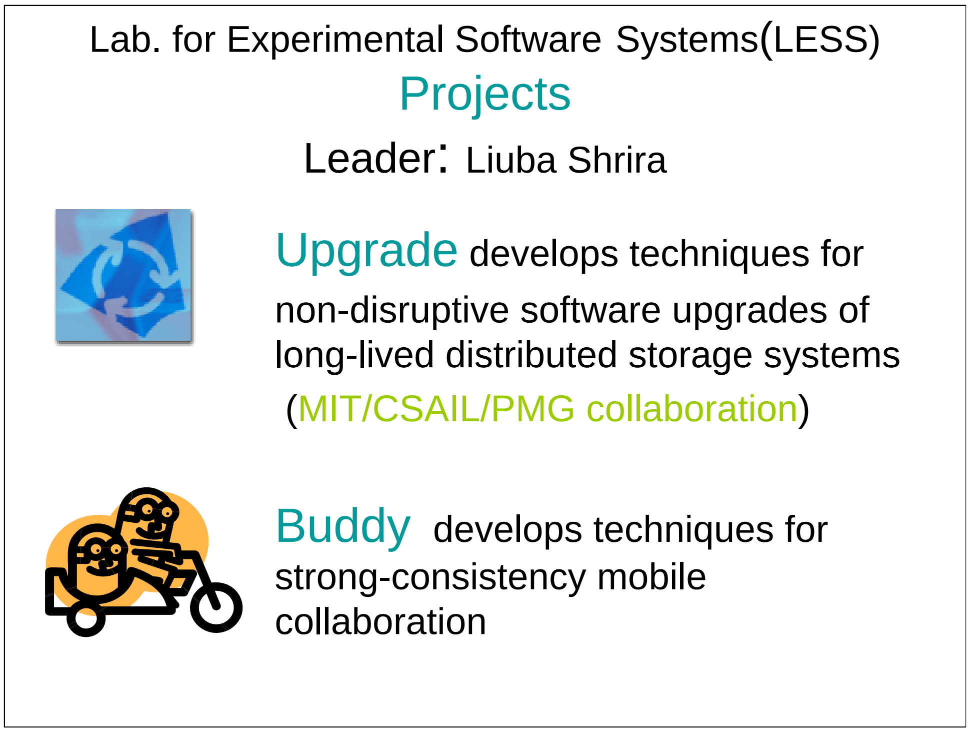

Lab. for Experimental Software Systems(LESS) Projects

Leader: Liuba Shrira

Upgrade develops techniques for non-disruptive software upgrades of long-lived distributed storage systems (MIT/CSAIL/PMG collaboration)

Buddy develops techniques for strong-consistency mobile collaboration

Upgrade:

Increasingly Internet systems are expected to be available 24-7: whenever a user requires stored information we want to be able to provide it. However, the systems are also intended to be long-lived;

therefore we must expect that its software will need to change during its lifetime.

The software of the system must be upgraded in a way that doesn't interfere with its availability. Bringing the entire system down while this happens is not an option. Furthermore, it can be expensive to

upgrade because upgrades can require transformation of persistent state, which can take a long time.

We are developing techniques that allow to upgrade large distributed systems In a non-disruptive way.

Buddy:

Improved connectivity for mobile devices allows collaborative computing to continue wherever collaborators go. This blurs the distinction between connected and disconnected work, rising the bar for disconnected collaboration

to resemble connected work. In spite of the connectivity improvements, wide-area connectivity remains an issue because of physical, economical and energy factors. It is important, therefore, to develop techniques that enable

to continue disconnected collaborative work without compromising the data consistency.

Byzy develops algorithms and implementation techniques to build practical scalable Byzantine fault-tolerant systems (MIT/CSAIL/PMG collaboration, supported by NSF)

TimeBox develops efficient high-performance storage system techniques for practical unlimited time travel (supported by NSF)

Byzy:

A very important problem today is how to make storage infrastructure robust in the face of failures and malicious attacks. Byzantine fault tolerance enhances the availability and reliability of

replicated services in which faulty nodes may behave arbitrarily (e.g. because they were subverted by a virus). Existing fault-tolerant techniques do not work well

in the Internet environment. We are developing techniques that provide high performance at Internet scale.

TimeBox:

Kurzweil says, computers will enable people to live forever and doctors will be doing backup of your memories by late 2030. We are not there yet. Instead, the remarkable drop in disk costs makes it possible and attractive to retain past application states and store them for a long time for mining or auditing.

A still open question is how to best organize the past state storage?

Our new snapshot-based approach to past state storage is attractive for several reasons. Snapshots are persistent, can be taken with high-frequency, and they are transactionally consistent. Unmodified database code can run against them. Like no other past state storage approach, they provide low-cost discriminated garbage collection

of snapshots, a useful capability in long-lived systems since since indiscriminately keeping all snapshots accessible becomes impractical over time even if raw disk storage is cheap, because administering such large-volume storage is expansive over long duration.