Automated support of STL containers

V. Perevoztchikov
Brookhaven National Laboratory,USA

What is the PROBLEM?

Current ROOT can write/read STL containers, but differently than native ROOT containers, like TClonesArray or TObjArray. Why?

- For each instance of STL container special wrapper is generated by rootcint;
- To read STL container, the same library must be loaded, which was used for writing;
- There is no general way to split them, as it is possible for TClonesArray;
- There is no general way to navigate,increase,decrease,read and write STL containers in ROOT.

To improve this, special approach was developed. Only for two of. them, vector and list, it is possible. But these two are the most. important.

Direct access to STL containers

Direct access to STL containers is a method to add, remove, increase, decrease container without using templates. This is enough for a general I/O. In the current design only *vector* and *list are* implemented.

Details*:*

- Special helper classes TSTLCont and TSTLIter were developed;
- Initialization by text descriptor like “vector<TNamed>” and by void* or void** as a pointer to object;
- Methods Resize(int),At(int),… represent according STL methods. All constructors and destructors of objects inside container are called properly;
- Method Streamer(TBuffer &) provides ROOT I/O for container, not for helper class itself;
- Methods New() and Delete() for new and delete of container.

Implementation details.

- Inside of helper classes TSTLCont and TSTLIter real containers are casted to *vector<char> and list<char>* . All the differences between these simple containers and real ones is accounted inside of helpers.

- All work with the objects inside of container performed via TClass etc… company. So these classes should be in ROOT dictionary;

- For *list* container memory allocated for objects is bigger then it really needs. In average it is bigger in 1.5 times, which is not too big penalty.

TTree improvements.

TTRee family of classes was modified, to introduce new functionality:

- *vector* and *list* containers could be written into *TTree* in split mode, exactly as *TClonesArray*.
- For reading no need anymore to load library used in writing;
- These STL containers are more general then *TClonesArray*. They could contain not only *Tobject** pointers , but pointers to any classes, any classes itself, and even simple objects like int, float, double etc…;
- *TTree::Draw()* , *TTree::MakeClass* etc…can work with STL classes exactly as with *TClonesArray* but in more general way.

Conclusions

- The helper classes ***TSTLCont and TSTLIter*** to support ROOT I/O for STL containers were developed;
- There is no more need to generate multiple wrappers for each instance of STL container;
- No need to load special library with wrappers;
- STL containers could be splitted like *TClonesArray*
- This spliting mechanism works not only for TOBject* pointers but for pointers to any classes, classes itself and simple objects like int,float, etc…