Subject: no true Iterator support in Upp???
Posted by kohait00 on Tue, 28 Sep 2010 20:56:05 GMT
View Forum Message <> Reply to Message

recently i came along to look closer at Iterator implementations in containers and i have to admit it is simpler that i expected..

generally i like simple, no doubt, it's most times the best option anyway. but the Iterator question somehow bugs me.. 2 things especially..

* Upp Iterator exist in 2 flavors, Iterator and ConstIterator, depending which one is able to be acessed, depending of the source state...(a const object can only return a ConstIterator). the speraration needs to be there to be able to correctly have mutator methods even on ConstIterator, like advancing the ii index..

* Iterator is *not* container independant, not even the template versions.., its not a 'sourceless' interface.. that you can store together with others from different container type but same element type i.e.

* Iterator interfaces are not coherent. the template version is not the same as the Vector version i.e. The template version is only used by the BiVector/BiArray containers, offering the rich interface..

* the virtual function less implementation of Iterator as per container, simpy beeing a pointer to an element is for speed reasons.., as well as the pure templated version..

what about a bit more flexible iterator interface?
i'm digging, as soon as done, i might post some ideas, but as i know mirek, he'll be up and done in no time with a cool solution anyway.

cheers

Subject: Re: no true Iterator support in Upp???
Posted by kohait00 on Tue, 28 Sep 2010 21:02:34 GMT
View Forum Message <> Reply to Message

a first idea to discuss would be:

* unify the Iterator and ConstIterator if possible
i think to make ii mutable would be no solution though, so this is probably not possible alltogether, the interfaces need to stay separated..but the 2 (const and non const) interfaces could be implemented by the same iterator imlementation, giving out the interface that is apropriate automatically..

* decouple the templated Iterator implementations for containers from the container dependancy, to be able to store the interfaces only, think of mixed containers environment, their interfaces in another common container to access the elements uniformely..

Subject: Re: no true Iterator support in Upp???
Posted by mirek on Wed, 29 Sep 2010 07:07:53 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Tue, 28 September 2010 16:56recently i came along to look closer at Iterator implementations in containers and i have to admit it is simpler that i expected..

generally i like simple, no doubt, it's most times the best option anyway. but the Iterator question somehow bugs me.. 2 things especially..

* Upp Iterator exist in 2 flavors, Iterator and ConstIterator, depending which one is able to be acessed, depending of the source state...(a const object can only return a ConstIterator). the speraration needs to be there to be able to correctly have mutator methods even on ConstIterator, like advancing the ii index..

No way around that. BTW, STL is the same.

Quote:
* Iterator is *not* container independant, not even the template versions.., its not a 'sourceless' interface.. that you can store together with others from different container type but same element type i.e.

Sure, but that is not even the point of iterators.

Quote:
* Iterator interfaces are not coherent. the template version is not the same as the Vector version i.e. The template version is only used by the BiVector/BiArray containers, offering the rich interface..

The reason there is template version is implementation only.

Quote:
what about a bit more flexible iterator interface?
i'm digging, as soon as done, i might post some ideas, but as i know mirek, he'll be up and done in no time with a cool solution anyway.

To tell the truth, iterators are modelled according to STL.

Gives at least one nice advantage: you can use STL algorithms with U++ containers

OTOH, I do not really like iterators - I appears to me that in many cases, you have to pass a reference to container with each iterator, while all algorithms use pair of iterators as range.

Perhaps the correct solution would be to provide some form Range interface... But whatever, in production code, iterators are almost never used outside Core package, so why bother... (and, they have that nice STL compatibility too, not that I ever used it).

Subject: Re: no true Iterator support in Upp???
Posted by kohait00 on Wed, 29 Sep 2010 09:28:00 GMT
View Forum Message <> Reply to Message

i'm not too experienced in that all as of now. so this was just state of current knowledge. i remember from c# time, that it has the IEnumerable / IEnumerator helpers that are pretty nice in terms of using foreach keywords on the arbitrary containers. could be a nice feature for upp as well...
this is also the kind of interface i had in mind that did not have eplicit container bonding, once you have obtained an instance of it, you can store it together with same iterators but from other container types..(MT stuff is other topic)

sth like this could be possible:

```
Vector<int> vi;
Array<int> ai;

Vector<Iterator<int> > iti;
iti << vi.GetIter();
iti << ai.GetIter();

for(int i = 0; i<iti.GetCount(); i++)
{
  Iter<int> & it = iti[i];
  while(it.MoveNext())
  {
  int& ii = it.Current();
  //do stuff with ii, dont care from where element came from
  }
}
```

```
//another thing: with a MACRO stuff like this should be possible:
//    C# syntax
//    foreach( string i in spp)
//            System.Console.WriteLine(i);
```

```
Vector<int> vi;
...

    FOREACH(int, i, vi)
      LOG(AsString(i));
```

see
http://msdn.microsoft.com/en-us/library/ttw7t8t6.aspx
http://msdn.microsoft.com/en-us/library/system.collections.i enumerable.getenumerator.aspx

donno if this is of initeres for upp at all, maybe if one is in need of things like this one should rethink it's design / model, if it's really correct..

upp states with a purpose: "... throw aside some old habbits"

Subject: Re: no true Iterator support in Upp???
Posted by kohait00 on Wed, 29 Sep 2010 10:12:46 GMT
View Forum Message <> Reply to Message

meanwhile:

```
#define FOREACH(type, var, cont) \
for(int __i = 0, __c = cont.GetCount(), __b; __i < __c; ++__i, __b = 1) \
 for(type& var = cont[__i]; __b; __b = 0)\


...


 Vector<int> vi;
 vi.SetCount(10);

 FOREACH(int, ii, vi)
  ii = Random();

 FOREACH(int, ii, vi)
  LOG(ii);
```

Subject: Re: no true Iterator support in Upp???
Posted by mirek on Wed, 29 Sep 2010 11:44:40 GMT
View Forum Message <> Reply to Message

[quote title=kohait00 wrote on Wed, 29 September 2010 05:28]

```
Vector<int> vi;
Array<int> ai;

Vector<Iterator<int> > iti;
iti << vi.GetIter();
iti << ai.GetIter();

for(int i = 0; i<iti.GetCount(); i++)
{
  Iter<int> & it = iti[i];
  while(it.MoveNext())
  {
  int& ii = it.Current();
  //do stuff with ii, dont care from where element came from
  }
}
```

Yes, but indicies are so simple and handy (and effective too).

Also, "enumerator" is not "iterator"

Subject: Re: no true Iterator support in Upp???
Posted by kohait00 on Wed, 29 Sep 2010 12:02:18 GMT
View Forum Message <> Reply to Message

i agree on effectiveness and all that.. no doubt.

but there is currently no possibility to have algorithms that can cope with arbitrary data structures.
i know that upp is handling much of it using template algorithms in Algo.h, etc. but it's not the same, though speed is an adavntage, no doubt.

it is not nesseccary to extend the containers with iterator / enumerator capabilities, such an interface could be returned by helperclasses, that are internally aware of the container, just as you sated.

BTW: wiki states that iterator / enumerator is same or am i missing some knowledge?

Quote:
C# and other .NET languages

Iterators in the .NET Framework are called "enumerators" and represented by the IEnumerator interface.

http://en.wikipedia.org/wiki/Iterator

Subject: PROPOSAL
Posted by kohait00 on Thu, 30 Sep 2010 14:16:41 GMT
View Forum Message <> Reply to Message

Here comes a proposal, meybe it is worth thinking about..

Iter<T>
ConstIter<T>

are interfaces to iterate through the containers (without beeing able to modify them)

currently supported containers:

ITER_CONT(Vector<T>)
ITER_CONT(Array<T>)
ITER_CONT(BiVector<T>)
ITER_CONT(BiArray<T>)
ITER_CONT(Index<T>)
ITER_CONT(ArrayIndex<T>)
ITER_CONT(Segtor<T>)

ITER_PTR(Ptr<T>)
ITER_PTR(One<T>)
ITER_PTR(T*)

Any
Value
Link<T>

the classes themselves are not changed at all.

```
Vector<int> vi;
vi.SetCount(10);
One<Iter<int> > ii = IterCreator::GetIter(vi);
while(ii->Next())
 ii->Get() = Random();

//or use macro
FOREACH(int, i, vi)
 i = Random();
```

there are also const variants, and respective macros

attached is a Test environment. it uses the Gen package from bazaar..for the Copyable interface.

a 'drawback' is the fact that the iter interfaces are all created on the heap to be able to clone them..here one can understand that i.e. C# enumerators rely on garbage collect features

nevertheless, i think it could be a helping package, when implementing abstraction layers..

please comment...

File Attachments

1) Iter.zip, downloaded 319 times

Subject: Re: no true Iterator support in Upp???
Posted by mirek on Thu, 30 Sep 2010 19:15:06 GMT
View Forum Message <> Reply to Message

kohait00 wrote on Wed, 29 September 2010 08:02i agree on effectiveness and all that.. no doubt.

but there is currently no possibility to have algorithms that can cope with arbitrary data structures. i know that upp is handling much of it using template algorithms in Algo.h, etc. but it's not the same, though speed is an adavntage, no doubt.

I guess the main reason is that there is no need for such thing....

Subject: Re: no true Iterator support in Upp???
Posted by kohait00 on Thu, 30 Sep 2010 19:25:15 GMT
View Forum Message <> Reply to Message

well, one should, ofcorse try to avoid polymorphism speed brakes but sometimes there is no other option..this one is just a tool among others, which fits somewhere as well. not everywhere though..
