Subject: Re: FairWriteoutBuffer::FillNewData and object ownership (memory leak)
Posted by Oliver Merle on Thu, 31 Jan 2013 13:24:56 GMT
View Forum Message <> Reply to Message

Thank you, Tobias.

Just as an opinion: the most elegant way - from the beginning - would have been to design FairWriteoutBuffer as a template FairWriteoutBuffer<t_digit, t_modify_functor > which is derived from base FairWriteoutBufferBase (<=> your current FairWriteoutBuffer). This way the compiler would autogenerate the boilerplate code which users currently have to implement. The only class the user would have to add is a custom modify functor in case he needs one.

The digit at the user side would be allocated on the stack and passed as const t_digit & (-> no new, no leak). This would decouple user and framework code so that you never run into these ownership problems.

Nevertheless, it doesn't hurt much to add the classes by hand. I mean ... you don't want to sell this package and scientists aren't too picky when it comes to copy and paste