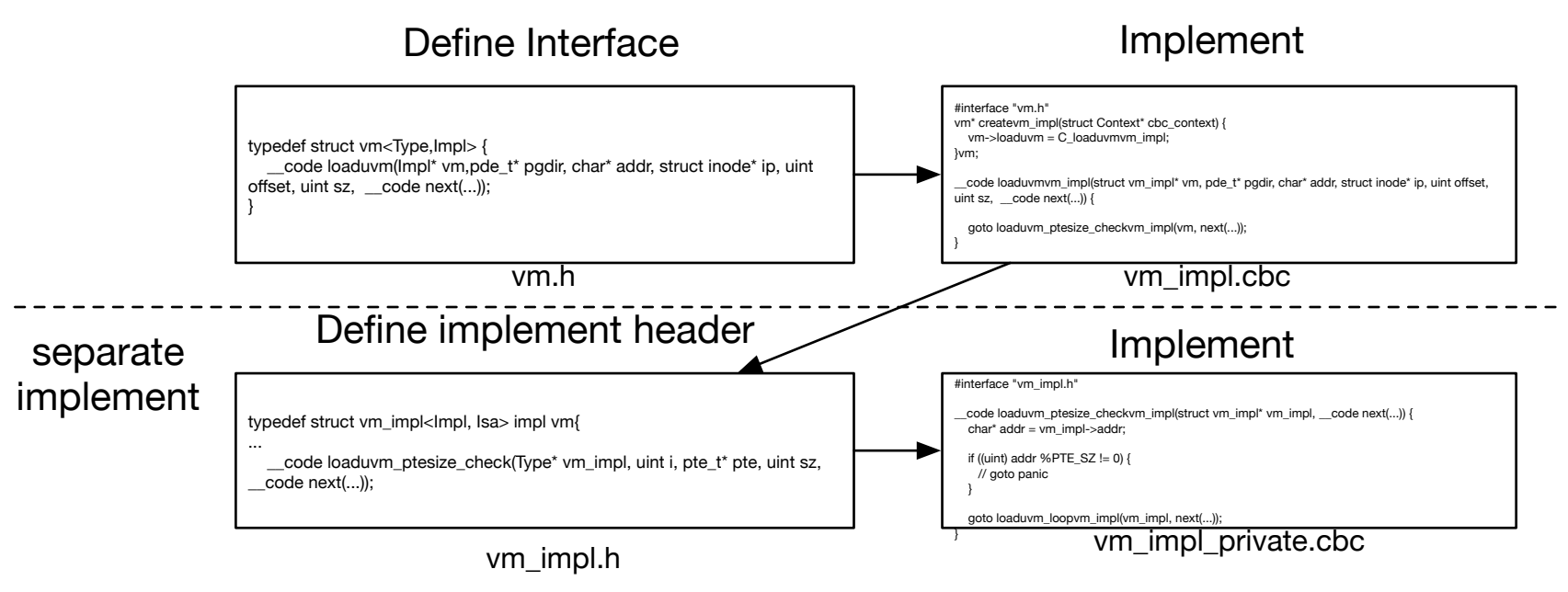
Define Interface
Implement
typedef struct vm<Type,Impl> {
__code loaduvm(Impl* vm,pde_t* pgdir, char* addr, struct inode* ip, uint offset, uint sz, __code next(...));
}
vm.h
#interface "vm.h"
vm* createvm_impl(struct Context* cbc_context) {
vm->loaduvm = C_loaduvmvm_impl;
}vm;
__code loaduvmvm_impl(struct vm_impl* vm, pde_t* pgdir, char* addr, struct inode* ip, uint offset, uint sz, __code next(...)) {
goto loaduvm_ptesize_checkvm_impl(vm, next(...));
}
vm_impl.cbc
separate implement
Define implement header
Implement
typedef struct vm_impl<Impl, Isa> impl vm{
...
__code loaduvm_ptesize_check(Type* vm_impl, uint i, pte_t* pte, uint sz, __code next(...));
vm_impl.h
#interface "vm_impl.h"
__code loaduvm_ptesize_checkvm_impl(struct vm_impl* vm_impl, __code next(...)) {
char* addr = vm_impl->addr;
if ((uint) addr %PTE_SZ != 0) {
// goto panic
}
goto loaduvm_loopvm_impl(vm_impl, next(...));
}
vm_impl_private.cbc