

Ikarus::KirchhoffLoveShell
::calculateVectorImpl

Ikarus::LinearElastic
::calculateVectorImpl

Ikarus::NonLinearElastic
::calculateVectorImpl

Ikarus::Volume::calculate
VectorImpl

```
graph LR; A[Ikarus::KirchhoffLoveShell::calculateVectorImpl] --> D[Ikarus::Volume::calculateVectorImpl]; B[Ikarus::LinearElastic::calculateVectorImpl] --> D; C[Ikarus::NonLinearElastic::calculateVectorImpl] --> D;
```

The diagram illustrates a design pattern where three different classes (KirchhoffLoveShell, LinearElastic, and NonLinearElastic) implement a common interface or base class method, calculateVectorImpl. Each class has its own implementation, but they all delegate the calculation to a single, shared implementation located within the Volume class. This is a classic example of the Strategy design pattern, where the behavior of an object is determined by one of several strategies that implement the same interface.