0.1 Używanie konceptów

Koncept to predykat czasu kompilacji (coś co zwraca wartość boolowską). Np. argument typu szablonu `T` mógłby mieć wymagania żeby być:

- iteratorem `Iterator<T>`
- iteratorem losowego dostępu `Random_access_iterator<T>`
- liczbą: `Number<T>`

Notacja `C<T>`, gdzie `C` to koncept a `T` to typ, to wyrażenie znaczące "prawda jeśli `T` spełnia wszystkie wymagania C, a nieprawda w przeciwnym wypadku."

Podobnie, możemy określić, że zestaw argumentów szablonu musi spełniać predykat, np. `Mergeable<In1, In2, Out>`. Taki predykaty wielu typów są niezbędne do opisywania biblioteki *STL* i wielu innych. Są bardzo ekspresywne i łatwo kompilowalne (tańsze niż obejścia metaprogramowania szablonów). Można oczywiście definiować własne koncepty i można tworzyć biblioteki konceptów. Koncepty pozwalają na przeciążanie i eliminują potrzebę wielokrotnego doraźnego metaprogramowania i kodu *scaffoldingu*[1] z metaprogramowania, co znacznie upraszcza metaprogramowanie, a także programowanie generyczne.

[1] metaprogramistyczna metoda budowania aplikacji bazodanowych. To technika wspierana przez niektóre frameworki MVC, w których programista może napisać specyfikację opisującą sposób wykorzystania bazy danych aplikacji. Kompilator używa tej specyfikacji, aby wygenerować kod, który aplikacja może wykorzystać do odczytu, tworzenia, aktualizacji i usuwania wpisów bazy danych