Title:

ConceptClang: An Implementation Model for C++ Concepts – An Update.

Speaker:

Larisse Voufo

Topic covered:

Generic programming with *concepts* is known to improve expressiveness, software components reusability, and safety in programming languages. The term "*concepts*" refers to a form of constraints-based polymorphism which, unlike subtype polymorphism (a la "Java generics" fashion), expresses algorithms and data structures in terms of shared properties of types rather than types. As such, *concepts* have been proposed to C++ as a much needed extension of its *templates* feature; which will improve its error detection and diagnosis. Despite a universal agreement for the said extension, design-specific details have been at the forefront of a two-decades debate that is continuing through another (current) decade, and is thereby becoming a notable example of what happens when well-founded theoretical ideas meet practical needs.

In 2010, we designed ConceptClang to aid the debate through implementation. The ConceptClang infrastructure is a generic and design-independent implementation of concepts for C++ in Clang—an LLVM frontend for the C family of languages. Ideally, given the nature of concepts, the implementation logic of ConceptClang could be ported to other compilers for other languages that support non-subtype constraints-based polymorphism. In fact, given our experience with ConceptClang, we have uncovered novel potential advances for programming languages theory. In particular, we have designed a framework for expressing and understanding name binding that introduces *weak hiding* as a new scoping rule and *two-stage name binding* as an associated mechanism. For object oriented programming languages (OOPL), our implementation unveils how C++ *concepts*, in particular, essentially bring open classes (or extensible structures) – a solution to the expression problem for OOPLs – for free in OOPLs with closed objects.

This talk will give an overview of generic programming with *concepts*, taking C++ *templates* as primary example, and relating it to ConceptClang. The talk will also highlight our ongoing theoretical findings and explore how ConceptClang can at least logically facilitate the implementation of *concepts* for other languages like Chapel.

Background and additional information can be found at `http://www.crest.iu.edu/projects/conceptcpp`. email: lvoufo@crest.iu.edu.