A vital part of software development is defining a good architecture when planning a project. But during implementation, the intended structures are often neglected and the program’s architecture grows in a different way than planned – a problem called architectural decay. When only looking at source code, tracking the architecture and keeping a high-level view of dependencies between components is difficult. Commonly used graphical models such as domain or architecture diagrams are helpful, but they usually just describe the intended design, rather than the actual structure. PyStructure’s goal was to develop an analyser which reveals these internal structures and dependencies in a Python program by statically analysing its source code. Dependencies are identified by determining what types are used in a particular component. But as Python is a dynamically typed language, types are not known before the application is run. Therefore a major aspect of the project was to develop an algorithm, called type inferencer, which deduces types by mimicking the evaluation process of the Python interpreter. The following details are extracted by the structural analyser:
Dependency Visualisation for a small Python Project

- All involved components such as modules, classes and methods
- Dependencies between these components caused by:
  - Method calls
  - Variables, arguments or attributes of a certain type
  - Inheritance

The extracted data can be visualised by Structure 101g, a program developed by Headway Software. Structure 101g shows dependencies as graphs in various levels of detail and provides a wide variety of analysis tools, including ones to detect cyclic dependencies or validate the current architecture. This allows users to effectively monitor the progress of a project's architecture during development. The structural data itself is also useful in other areas of software development, such as enhanced accuracy for automatic refactoring or better support for code completion in integrated development environments.