Object Orient Analysis

Object-Oriented programming has become the most popular design paradigm in modern computer programming languages by no mistake. There are many advantages to OOP, which make it an obvious choice for writing programs that have easily maintainable code, enhance re-usability and reduce programming applications to the perspective humans have on the world. Dan Ingalls’ lecture on OOP makes a strong case for why OOP is a positive approach to building applications.

Object-Oriented programming benefits a team of programmers working on the same project. By separating code into logical objects, we reduce the repetition of code, increase the speed of production and simplify large and complex problems into many smaller parts. Ingalls states that Modularity attempts to reduce the quantifiable aspect of the following principle:

If any part of a system depends on the internals of another part, then the complexity increases as the square of the size of the system.

Ingalls use an example of Queue with Doctors, Nurses, and Patients, describing how procedural programming would have a more difficult time handling each case in every section of the code where it is used. This is error prone, since it gives a chance for code to be changed in one part of the application and left the same in another part of the application. This can lead to inconsistencies, even in very simple programs.

OOP introduced Inheritance to create general code that could be shared between more complex objects, allowing them to co-exist in data structures, reducing code maintainability, and speeding up production by eliminating the need to copy code into child objects. Inheritance shares code between the objects and can be updated across the application by editing a single file. Inheritance dramatically reduces the amount of code needed to repeat throughout the application.

The speed of development and repetition of code is not the only advantage of Modularity. Modularity also benefits the debugging phase of development, allowing the system to respond to errors and alert the programmer with Exceptions.

Ingall’s emphasizes the concept of messages. Messages are used between objects to react to things taking place in other objects. Messages enhance debugging because they allow compilers to respond when an object does not know how to interpret the message. This allows compilers/interpreters to throw exceptions without the programmers having to understand and handle all the types of errors that may occur when piecing together application. Basically, messages enable the system to intuitive react to errors, enabling them to through exceptions during compile and runtime.

Ingall’s believes that OOP has great benefit to application development because its ability to reduce clutter and handle error-prone operations such as Automatic Storage Management. He talks about how in procedural programs, Storage Management code is scattered throughout the code. This enhances the chance of have programs crash unexpectedly due to mistakes or changes in Storage Management.

Ingall’s states that OOP is a natural choice when designing Operating Systems and Graphical User Interfaces because these systems are structured in an Object Oriented way. Files and Directories are objects and are often called upon to repeat process. Graphical User Interfaces have components that are repeated throughout the entire system, but handle a generic principle in different ways. OOP resolves these issues by allowing programmers to create objects with different properties while handling similar process.

Ingall’s and the SmallTalk team were the founding fathers of Object Oriented Programming. There contribution to computer science laid a foundation for technology to grow exponentially. Without OOP principles our world would not exist and computers would not have the value they do today.