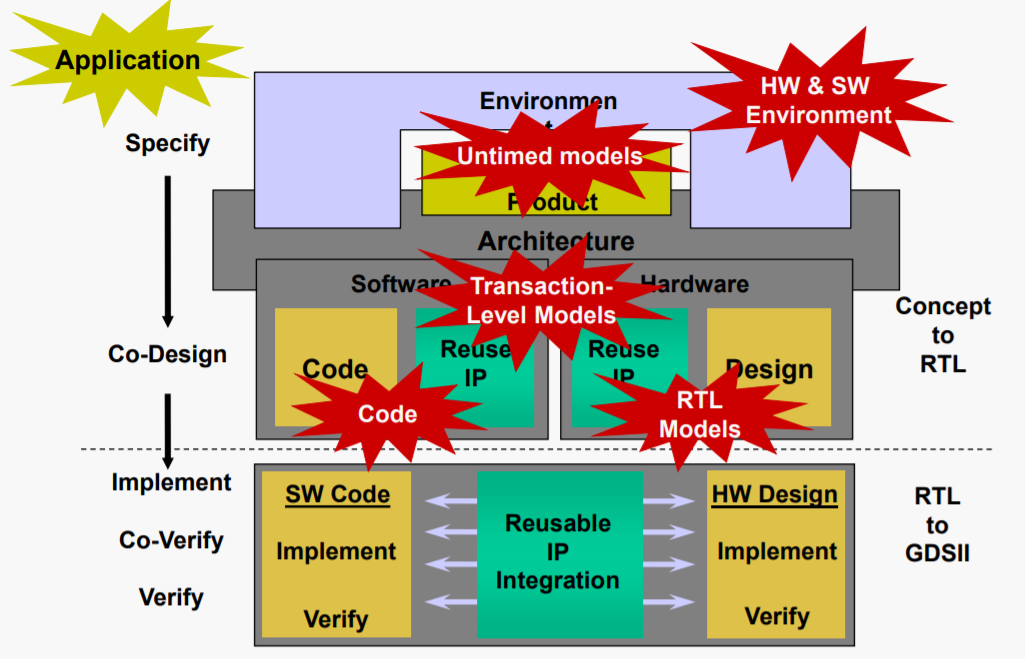
SOC vs. SIP

* SOC: system modules are integrated on the same die
  + Fabrication must be compatible with standard CMOS processes
* SIP: multi-die multi-process “stackable” module integration
  + More cost effective than single-die single-process approach

Hardware & Software Design



Verilog/VHDL not enough?

* Systems become more complex, pushing designers to design and verify at higher levels of abstraction
  + Software is playing an increasing role in system design
    - Need to support early software development, integration with existing C/C++ code, and HW/SW codesign
  + Need a design language that supports system-level IP delivery/integration
* Build on C/C++ so that:
  + Extensive C/C++ infrastructure (compilers, debuggers, language standards, etc) can be reused
  + Users’ existing knowledge of C/C++ can be leveraged
  + Integration with existing C/C++ code is easy

SystemC

* SystemC:
  + Enables early exploration of system-level design trade-offs
  + Enables early and fast system-level verification
* SystemC is a subset of C++ that model’s hardware
  + Language definition is standardized by IEEE
  + It provides a simple form of concurrency
    - Cooperative multitasking
  + SystemC simulation libraries are freely distributed
    - Commercial compilers available for translating SystemC programs into gate netlists
  + SystemC program consists of module definitions plus a top-level main function that starts the simulation

**Briefly State the benefits and drawbacks of SystemC?**