# Spiral Model

* **Definition:**The spiral model has four phases. Using this model, the SDLC goes thru these phases that are called spirals. The spirals are:
  + **Identification:** This phase gathers requirements and is the requirement engineering stage of the model. With the spiral model, requirements are frequently revisited to ensure the validity and integrity of the requirements.
  + **Design:** As the SDLC progresses, designs become less conceptual as they are in the beginning and evolves into logic design, architecture design, physical and logical designs.
  + **Construct or Build:** During this phase in the spiral model, actual development and implementation is happening. The early builds are prototypes, most likely prototypes to keep. They are kept because customer feedback is gathered thru the evaluation and risk analysis process and these requirements are implemented into the system more and more.
  + **Evaluation and Risk Analysis:** During this phase is when prototyping is being tested and given feedback by the stakeholder. Also, this phase is where the developer will state any problems or things worth mentioning to the stakeholder. Including but not limited to time constraints as well as financing concerns.
* **Advantages:**
  + Changing requirements can be implemented.
  + Allows for many prototypes and build tests.
  + Requirements are always revisited and validated throughout the SDLC
  + Stakeholders see the system early and can test it.
  + Development is incremental and risky code can be implemented and tested before the end user will see it.
* **Disadvantages:** 
  + Project Management is more complex and has higher chances of errors being implemented.
  + Project completion is hard to determine.
  + Generally directed at large expensive projects. Cheap small projects may go over budget.
  + It is a tedious and complex process.
  + Many phases require a lot of documentation.

# Incremental Model

* **Definition:**Similar to the waterfall model, this model begins with a process of obtaining requirements from the stakeholders, however, this model involves a prototype to keep that is implemented and enhanced thru the SDLC. This model does not start with all the requirements on hand, and instead implements a prototype for the stakeholder to test and give feedback on. This is a repeated process. A new prototype is introduced every increment or iteration; it builds on little by little. This happens thru the entire SDLC until the final product is deployed. It is detrimental to the project to fully understand the stakeholder’s requirements, and verification of each build released. Tests are integral to this model and must happen thru out the SDLC.
* **Advantages:**
  + Early testing and presentation to the stakeholder.
  + Early results and multiple variations of results are obtained.
  + Development can happen in parallel. I.e. Two concepts or subsystems can be implemented alongside the main development process.
  + Progress is measurable.
  + Scope Creep is less harmful and less expensive.
  + Small and incremental tests are relatively easy.
  + Each iteration or increment allows for easy risk management and analysis.
  + During the SDLC operational products are delivered.
  + Next increment/milestone can have code and feedback implemented easily.
  + Requirement change is manageable.
  + Low operation time. *What does this mean*?
  + Large and critical projects are better suited with this model.
* **Disadvantages:**
  + More resources may be needed. Time and money, team members etc.
  + Cost of change may be lesser, but this model does not suit changing requirements easily.
  + Project management is critical and must be done very carefully.
  + Because not all requirements are straightforward and known, design flaws and project infeasibilities may be encountered.
  + Some increments may require full understanding of the system.
  + Suited for large projects.
  + End of SDLC is hard to determine.
  + Expensive resources are required for risk analysis and management.
  + Risk management is highly important and is critical to success.

# Agile Model

* Definition:  
  The agile model is a mix of incremental, and iterative process models focusing on adaptability and stakeholder satisfaction. This model has rapid delivery of the product at hand. This model breaks the project up into small builds, these builds rely on the iterative model. This model allows for multiple teams to work on a single project, while focusing on different aspects. These aspects include but are not limited to: planning, requirements engineering, design, coding, unit testing and acceptance testing. This model allows for feedback from the stakeholders at the end of each iteration. This model allows for every project to be handled differently and apply custom methodology to each project. This model relies on**:**
  + Individual interactions between developers and stakeholders are very important as well as team work between the developers.
  + A working demo/prototype is a must and must show
  + The project must respond well to change and adaptability
* **Advantages:**
  + Well suited for real world development.
  + Enables good team work and cross skill sets
  + Rapid development with working prototypes
  + Resource requirements are minimal
  + High adaptability
  + Early prototypes and working demos.
  + Adaptable to changing environments.
  + Documentation easy to do
  + Concurrent development with overall system development.
  + Minimal planning
  + Easy project management
  + Flexible for developers
* **Disadvantages:**
  + Inept for complex dependencies/requirements.
  + High risk in maintenance and extensibility.
  + Needs a good project lead and a good development plan
  + Strict delivery of product to stakeholder is must.
  + Stakeholder interaction is critical to success.
  + If no documentation is left, a change in team is hard to manage.
  + Low documentation.