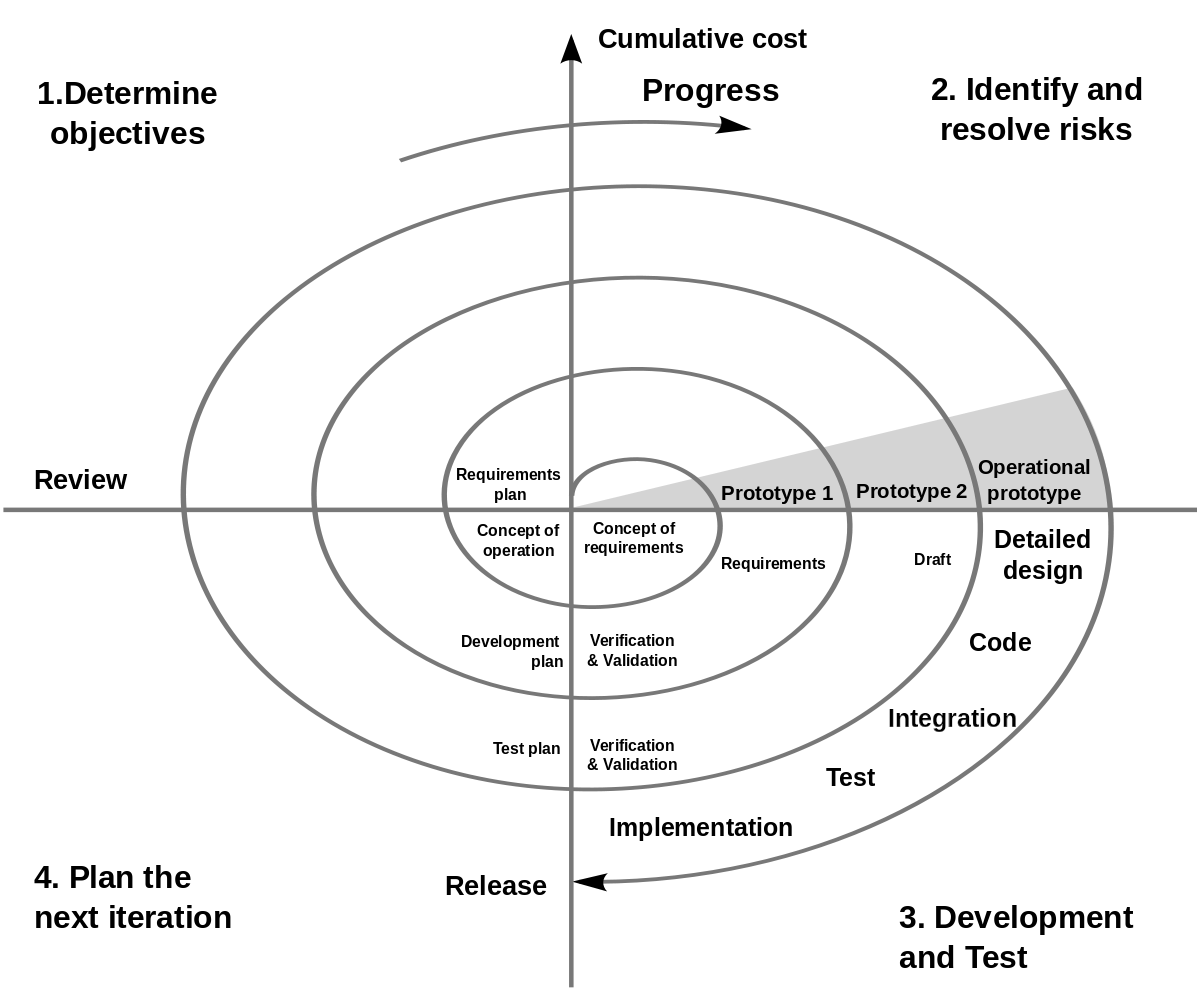
**Spiral**

The **spiral** model is a **risk-driven process model generator for software projects**. Based on the unique risk patterns of a given project, the spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall, or evolutionary prototyping. This model was first described by **Barry Boehm** in his **1986** paper "A Spiral Model of Software Development and Enhancement".

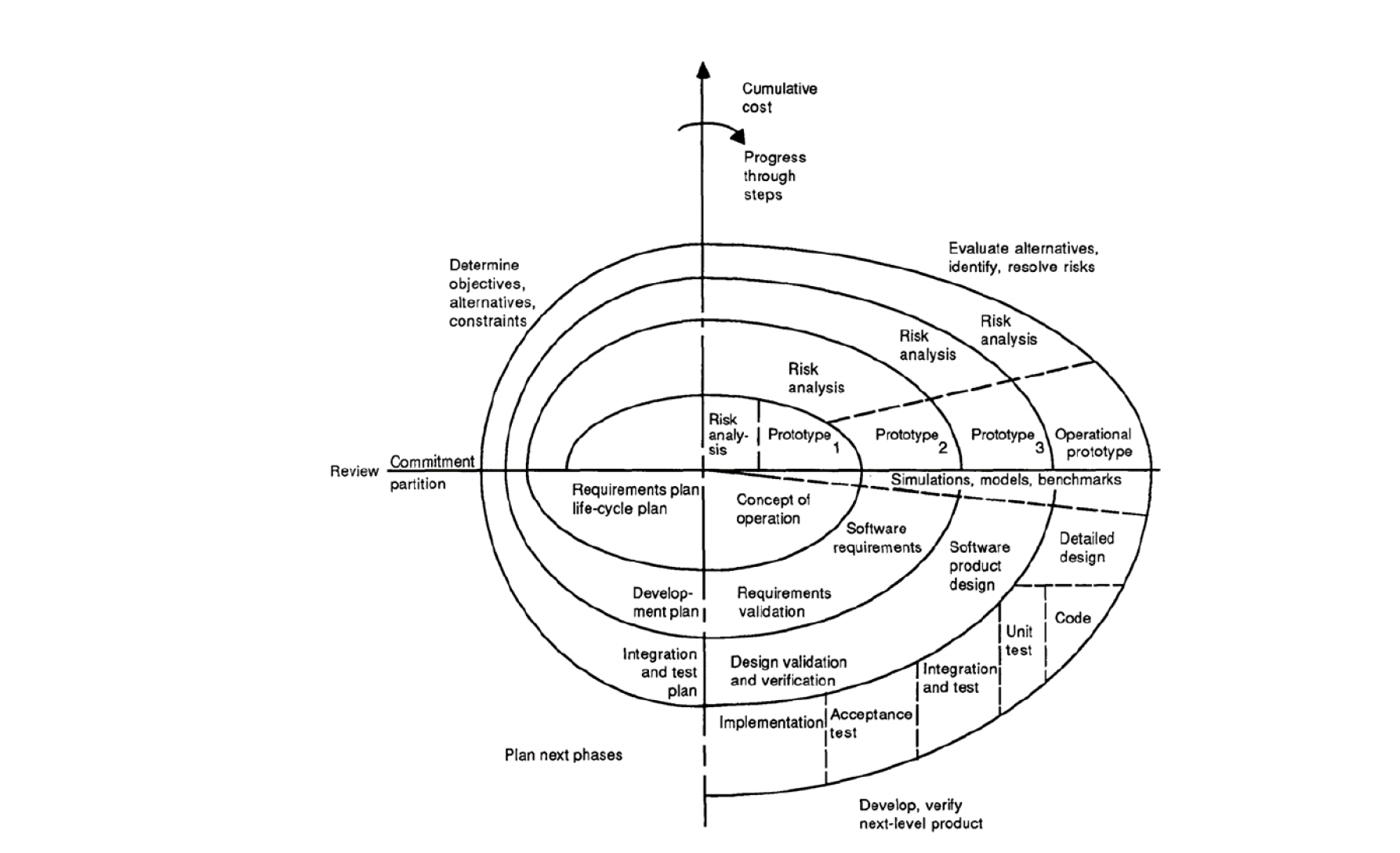


**Figure 1**: Spiral model (Boehm, 2000). A number of misconceptions stem from oversimplifications in this widely circulated diagram, with a few errors.

Boehm identified a **number of misconceptions arising from oversimplifications** in the original spiral model diagram, such as:

* The spiral is simply a sequence of waterfall increments.
* All project activities follow a single spiral sequence.
* Every activity in the diagram must be performed, and in the order shown.

**Authentic applications** of the spiral model are driven by cycles that always display **six characteristics** and **four phases**.



**Figure 2**: Spiral model of the software process.

**The Six Invariants**

* Define artefacts concurrently.
* Perform four basic activities in every cycle.
* Risk determines level of effort.
* Risk determines degree of details.
* Use anchor point milestones.
* Focus on the system and its life cycle.

As mentioned above, the spiral model has **four phases**. A software project repeatedly passes through these phases in iterations called Spirals.

* Phase 1: **Identification**
* Phase 2: **Risk Analysis**
* Phase 3: **Engineering and Evaluation**
* Phase 4: **Planning**

**When to use Spiral method**

* When costs and risk evaluation is important.
* For medium to high-risk projects.
* Long-term project commitment unwise because of potential changes to economic priorities.
* Users are unsure of their needs.
* Requirements are complex.
* New product line.
* Significant changes are expected (research and exploration).

**Pros**

* Changing requirements can be accommodated.
* Allows extensive use of prototypes.
* Requirements can be captured more accurately.
* Users see the system early.
* Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.

**Cons**

* Management is more complex.
* End of the project may not be known early.
* Not suitable for small or low risk projects and could be expensive for small projects.
* Process is complex
* Spiral may go on indefinitely.
* Large number of intermediate stages requires excessive documentation.

**Tutorial**

****

Youtube video: <https://youtu.be/mp22SDTnsQQ>

<https://en.wikipedia.org/wiki/Spiral_model>

<https://www.tutorialspoint.com/sdlc/sdlc_spiral_model.htm>

<http://istqbexamcertification.com/what-is-spiral-model-advantages-disadvantages-and-when-to-use-it/>