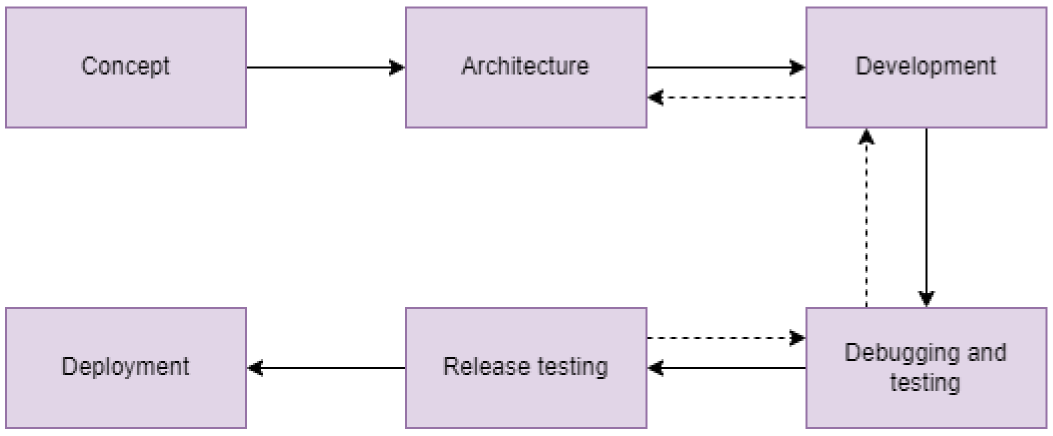
Formal verification inside the bigger picture

Prepared by Pruvendo at 05/23/23

The smart contract development teams often ask at what stage of development the formal verification should be started, when it is assumed to be completed and what is the overall overhead in terms of human efforts. The present document is intended to answer these questions.

Workflow assumption

Throughout the present document it is assumed that the development team follows some kind of development process. While the names and exact content of the stages are different for various teams the it's expected the following diagram is mostly related to the reality:



So, the following six stages are recognized[1]:

- Concept - The basic paradigm, set of features and logic of execution
- Architecture - the list of smart contracts, key methods and their interaction
- Development - the main part of development
- Debugging and testing - reaching the quality acceptable for production
- Release testing - final pre-production procedures
- Deployment - considered as a final point of the process. Post-production stages are off the consideration

[1] Again, they can be slightly different for the specific team

Injection of the formal verification

Pruvendo suggests three approaches of the formal verification injection: ideal, good and poor. Each of them is considered below. Please keep in mind the following terminology:

- High-level specification - halfly-independent from the implementation, should be understandable by the development team and their technical leaders
- Low-level specification - bound with the implementation and follows it
- Verification preparation - converts Solidity implementation into some form comfortable for the verification
- Quick Verification - the verification process based on the randomized testing (using QuickChick tool) that allows to quickly validate the project.
- Full Verification - the most advanced "deductive" verification process with strict mathematical proving

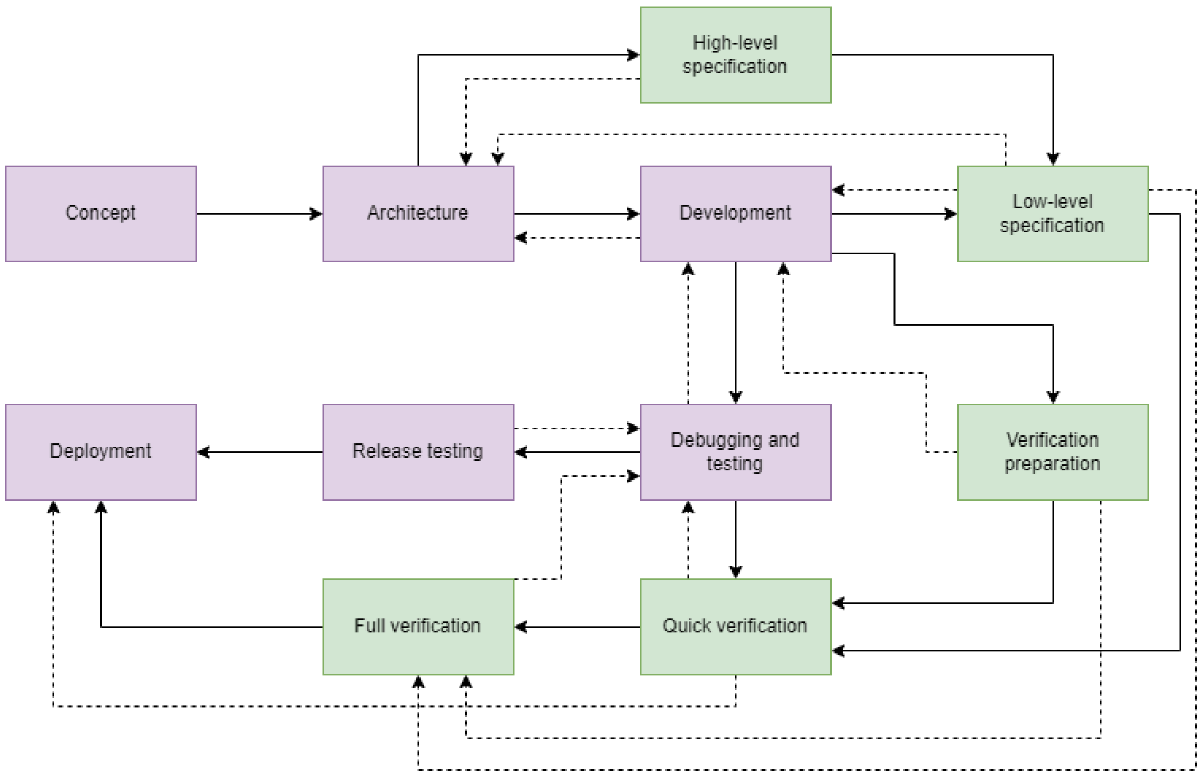
In some cases either Quick or Full verification can be justified to be skipped.

The efforts required for each of these five phases depends on the particular contract but roughly can be distributed evenly.

Please note, the specification efforts often find bugs in the architecture, so it's advised to start this activity as soon as possible. Also, it's useful to minimize the production delay due to the ongoing formal verification.

The recommended (ideal), acceptable (good) and not-recommended approaches are presented below.

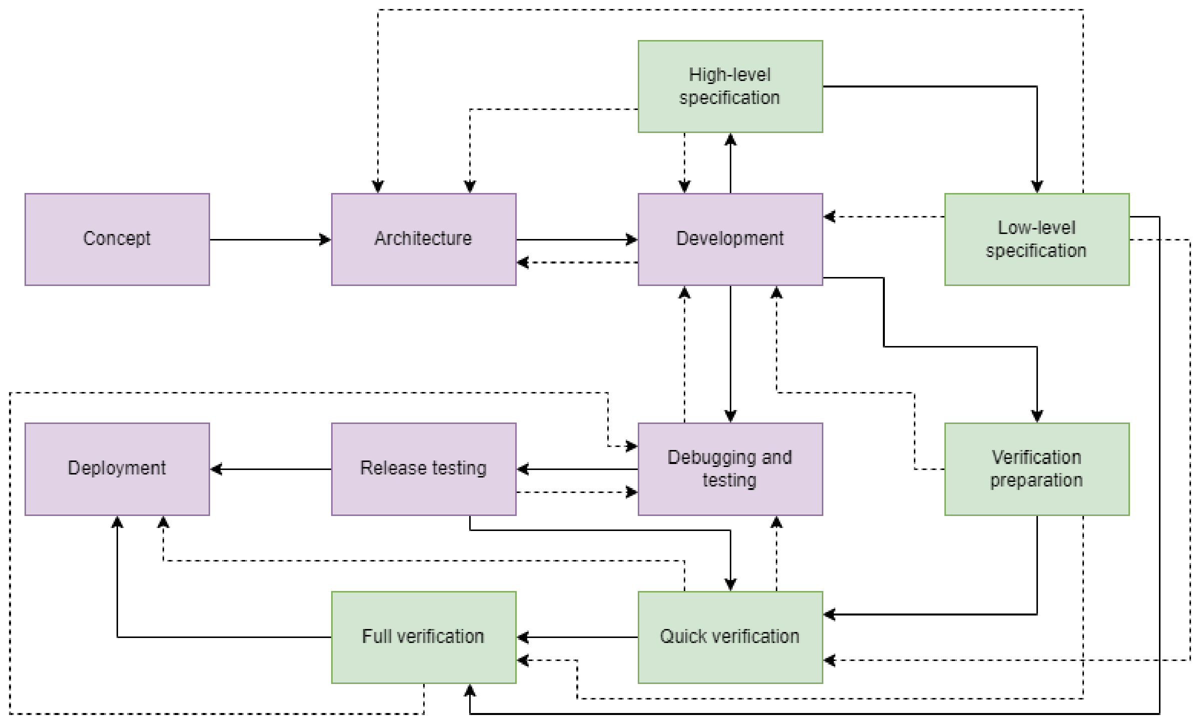
Ideal approach



In this approach the formal verification starts immediately after the architectural stage that allows to create high-level specification as well as to find potential critical architectural bugs during the main development.

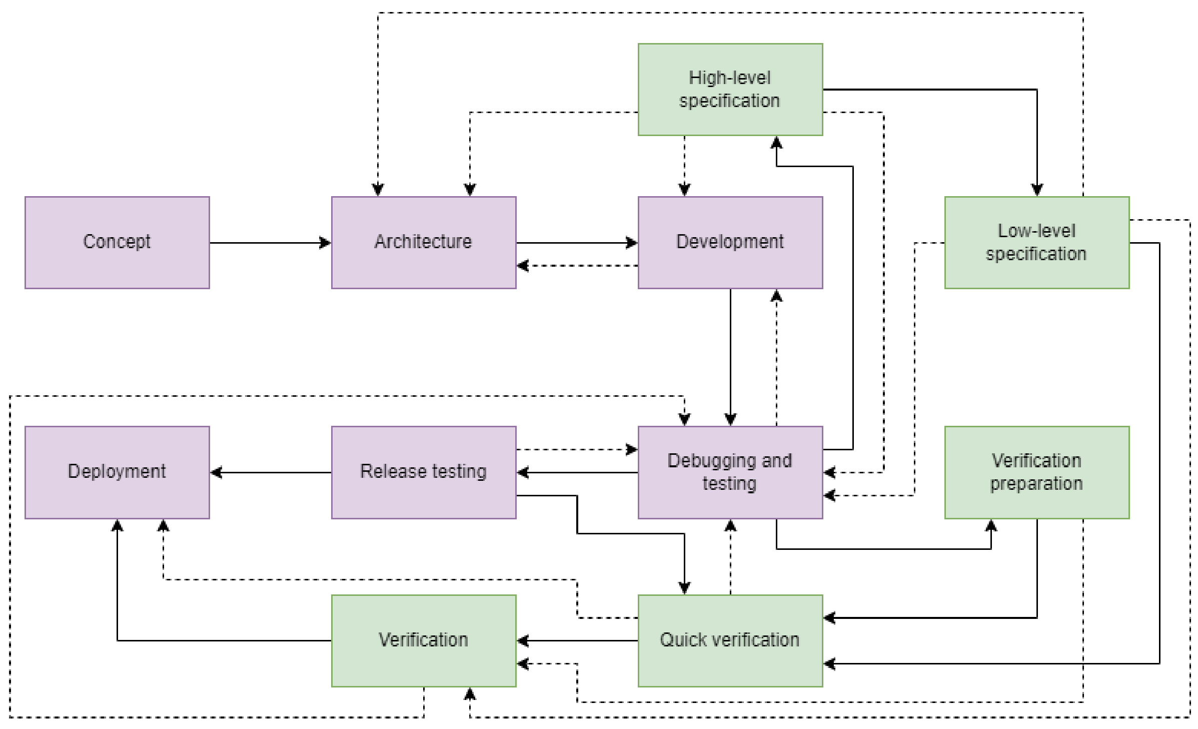
Generally speaking, this approach allows us to make all the preparations by the time of release testing and minimize the shift of the deployment (to 1-2 weeks in some cases).

Good approach



In this approach the formal verification starts when the main development is completed. Its usage can bring some problems in case of architectural bugs being found. Also the delay of deployment will be large (~1 month for the medium-sized contract).

Poor approach



At this approach the formal verification starts when almost everything is ready. While Pruvendo believes it is a bad idea to start so late, many customers prefer this approach. In this case bugs can be especially painful and the customer is expected to wait for the whole verification process (can be up to a few months). Pruvendo does not recommend this approach.

Human efforts

The human efforts required for the verification depend on many factors so please just see a few of examples:

- Elector - 5 man-months
- SafeMultisig2 - 1.5 man-month
- ERC20 (Ethereum) - 6 man-days

Constantly working on technology improvement Pruvendo strongly believes it should be able to dramatically decrease these numbers in the foreseeable future.