Contest Proposal: Smart Contract Debugger (Phase 1: Specs)

Dates

Start: November, 16, 2020, end of day UTC
End: November, 30, 2020, end of day UTC

Motivation and goal

Smart Contract Debugger is a highly anticipated toolchain component. Developer Experience with Debugger means visualization of code execution context, deeper developers' understanding of platform specifics, and better developer productivity.

The goal of the contest is to find the best principal design solutions and describe specifications for further smart contract debugger development.

Task

The best design solution for Smart Contract Debugger is combination of the following qualities:

- **Platform Context.** Consideration of TON Protocol unique features and restrictions.
- **Usability**. The design should take into account the most typical needs of a smart contract developer.
- **Composability**. Clear protocols of communication between compilers, the debugger and virtual machines. A new compiler or a new virtual machine if it implements the corresponding protocol should easily (ideally, automatically) integrate with the debugger.
- **Reuse of Ideas and Code.** Reuse of existing open-source codebases / standards / formats rather than reinventing the wheel is a merit. But the rationale for this reuse should be explained as well as its applicability.

Evaluation Criteria

Author should consider following structural points.
If submission lacks some of these points, submission may be rejected or its grade decreased.

1. **Description of use cases and restrictions.** Scenarios of usage of the debugger or the tasks you can solve with the debugger. Describe restrictions of proposed solution (if any).

2. **Functional Specifications**. Detailed description for each scenario from a programmatic point of view: i.e. what are inputs and outputs, what are the formats for the information exchanged between tools (debug information is usually generated by a compiler and the debugger only processes it), what components of the toolchain are involved in each scenario, what are the algorithms the debugger use.

3. **Reasoning for principal technical decisions**. Each author's technical decision should be supplied by a concise overview of existing solutions (in the scope of related Computer Science topic) and provide pros and cons for the proposed solution. ***Example:*** *It is not good to propose that exact debug info format should be used. Such decisions must be backed up by reasoning. To support his/her decision, the author may analyze pros / cons of the proposed format, highlight related components' interfaces / implementation parts, etc. In other words - provide all information by which the author was led while designing the Debugger Specs*

4. **Dependencies and Blocker Issues.** Identify, describe and prioritize what TON OS Components' specifics or TON Protocol implementation details should or should not be changed to use the proposed specs as basis for further Debugger development.

5. **Illustrations**. For ease of understanding one may supplement complex abstractions with graphics: component relation schemes, UML, data flow diagrams, etc. Each supplied visual should match the related part of proposed specs.

6. **References.** Every reference to facts, specifications, existing solutions / findings should be supported by the relevant document link (URL, page, chapter) where this information could be easily verified.

Optionally, author may cover additional important topics:

1. **Roadmap**. At least high-level view of further tasks (contests) to be done along with priorities + approximate timeline.

2. **API**. How 3rd party tools like visualizers, IDE, etc can interact with the debugger.

3. **Additional materials.** Implemented components / classes, algorithms or useful implementation details.

Reward Structure

1. 💎 20 000
2. 💎 15 000
3. 💎 10 000
4. 💎 7 000
5. 💎 5 000

6-10. 💎 1 000

Multi-stage Contest Notice

Please note that the results of this contest imply no restrictions on the further implementation contests, i.e. the final debugger specification for next contest phases might be a mix and match of the design solutions provided by the competitors.