Queen's School of Computing
Queen's University
Kingston, Ontario, Canada

Queen's University

I am therefore I compute

# Model-level, Platform-independent Debugging in the Context of the Model-driven Development of Real-time Systems

Mojtaba Bagherzadeh, Nicolas Hili, Juergen Dingel

# Outline

- Motivation
- Approach
- Illustrative example
- MDebugger
- Conclusion

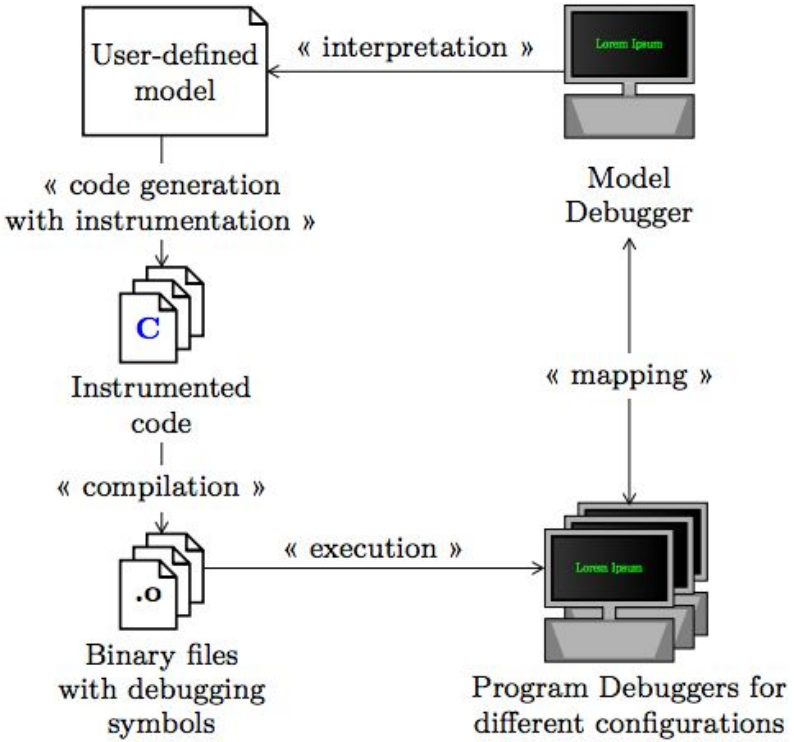
# Motivation

- Debugging support is one of the central barriers to a broader adoption of Model Driven Development (MDD).

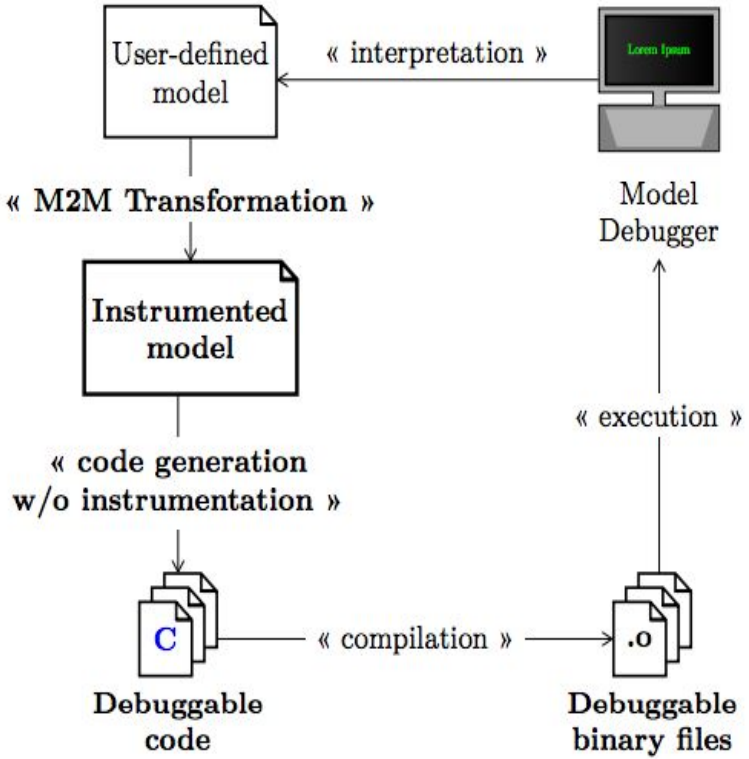
- MDD tools should provide debugging services to help user to debug their application at model-level.

# Existing Solutions

- Simulation
- Trace and reply
- Interactive Debugging on target platform



# Our Approach

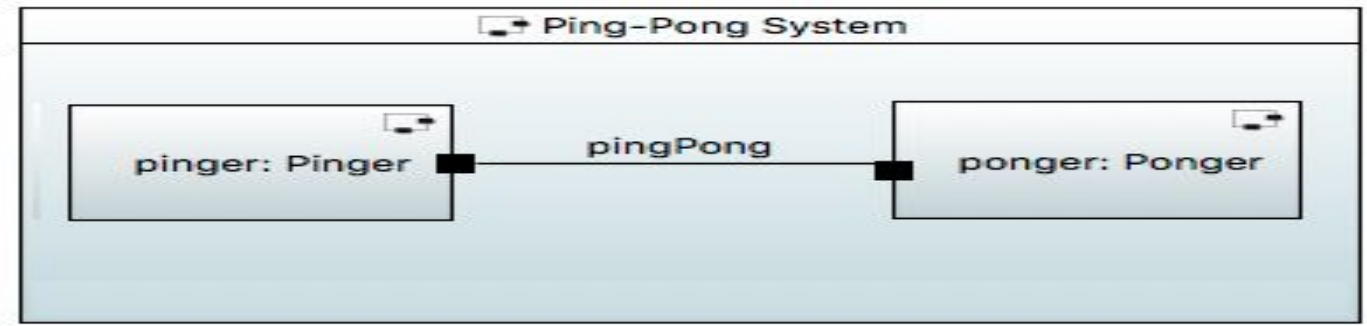


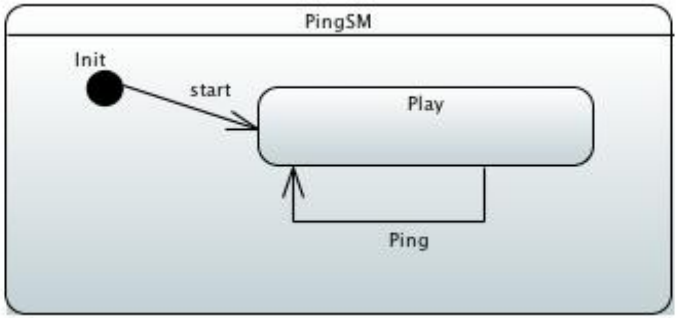

- Relies on a model instrumentation process which is platform-independent.
- Using model transformation communication between the debuggable system and the model debugger, and debugging capabilities are implemented.

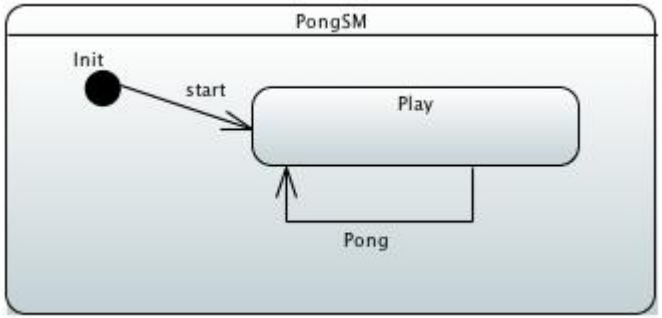
# Applied Model Transformations

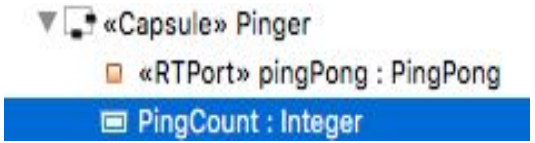
- Communication
- Stop and Resume Operation
- Variable View and Change
- Generate monitoring events

# Illustrative Example - User Defined Model



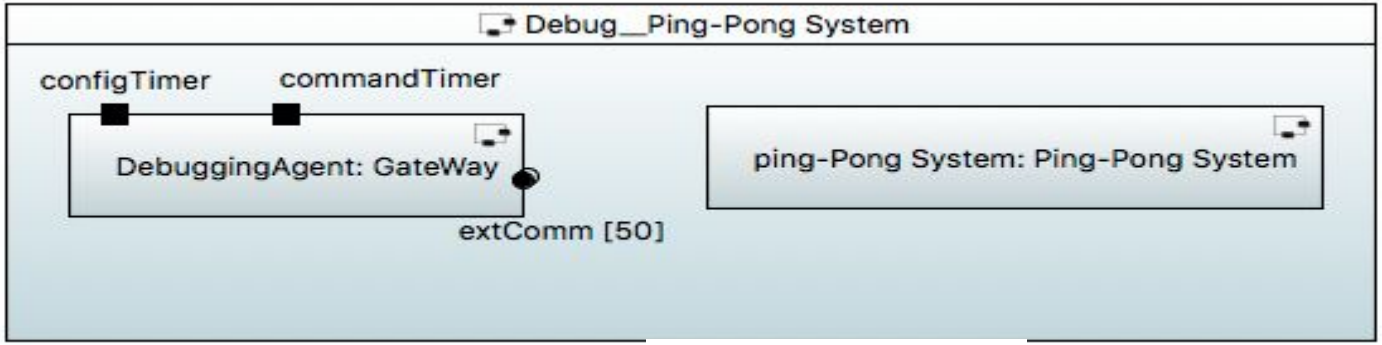



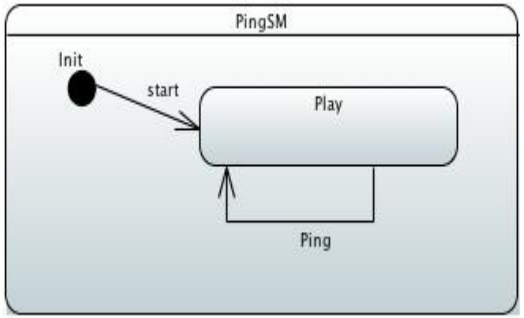
# Illustrative Example - Transformed Model

Communication Layer and Reflection:

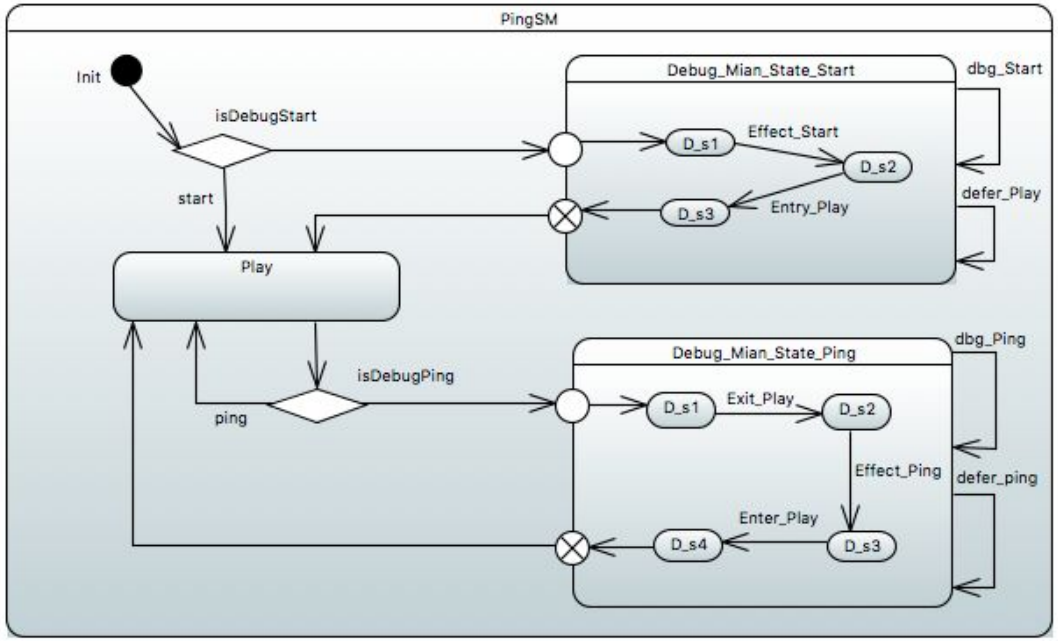


# Illustrative Example - Transformed Model

Stop and Resume (Pinger):




**User Defined Model**




**Instrumented Model**

# MDebugger

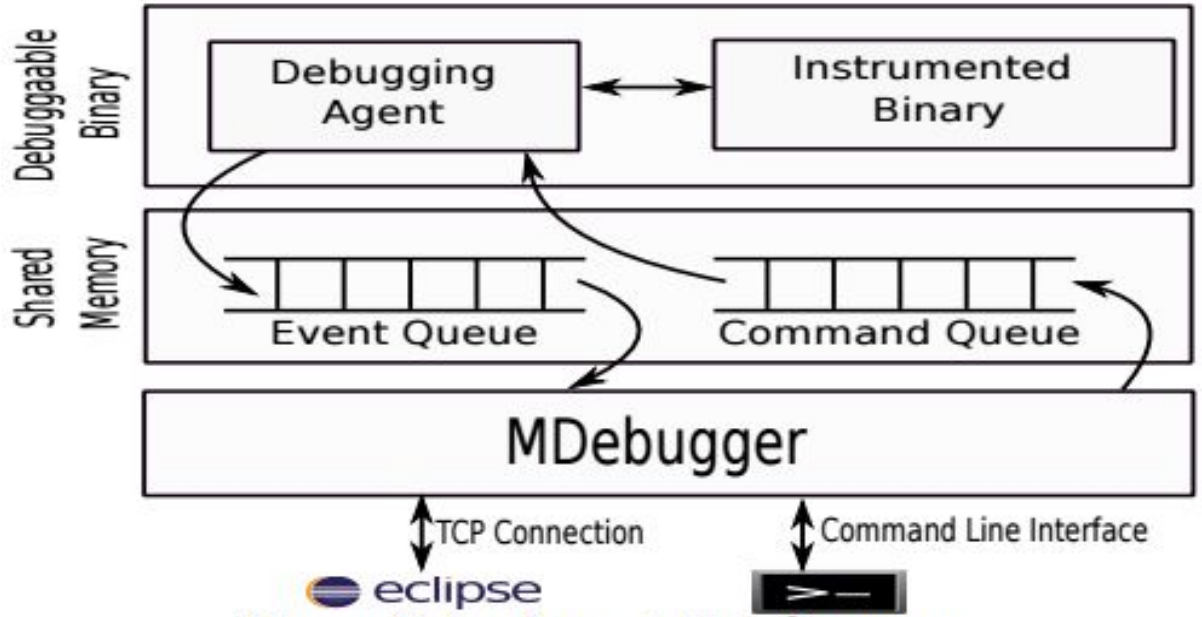
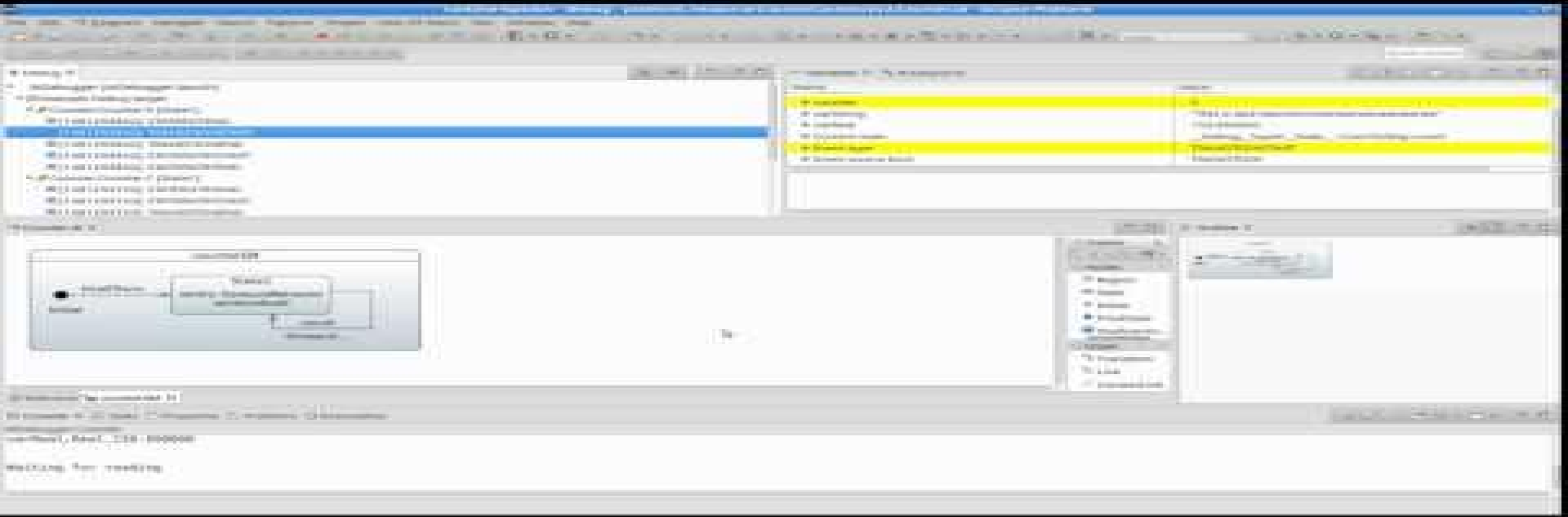



Figure 7: Implementation Overview

# MDebugger - Command Line

```
mdebugger#help
Available Options
"help|h"                                          (Show the commands and their options)
"breakpoint|b"  -c capsuleName -t name -b  -i traceNo(Set breakpoint at start of a transition)
"breakpoint|b"  -c capsuleName -t name -e  -i traceNo(Set breakpoint at end of a transition)
"breakpoint|b"  -c capsuleName -t name -s  -r  -i traceNo(Remove breakpoint at end of a transition)
"breakpoint|b"  -c capsuleName -t name -e  -r  -i traceNo(Remove breakpoint at end of a transition)
"next|n"        -c capsuleName -i traceNo       (Execute until next step)
"continue|c"    -c capsuleName -i traceNo       (Continue execution until next breakpoint)
"run|r"         -c capsuleName -i traceNo       (Run capsule without interrupt)
"modify|m"      -c capsuleName -n name -v value -i traceNo(Modify a attribute of capsule)
"view|v"        -c capsuleName -v  -i traceNo   (View the capsule's attributes)
"view|v"        -c capsuleName -n count -e  -i traceNo(View n last action of capsule's action chain)
"list|l"        -i traceNo                      (List running capsules and their current state)
"list|l"        -c capsuleName -i traceNo       (List capsule's configuration including breakpoints and etc)
"list|l"        -c capsuleName -b  -i traceNo   (List exiting breakpoint)
"save|s"        -c capsuleName -i traceNo       (Save existing events)
"connect|con"   -h host -p port -i traceNo      (Connect to eclipse debugger)
```

# MDebugger Integration with PapyrusRT



# Conclusion

- We presented a new way of providing debugging at model-level.
- Our solution is implemented at model-level using modeling concept and is not dependent on program debugger or generated code.
- The size overhead of our approach is comparable with other methods.
- The performance overhead of the approach is small and acceptable for debugging RTE systems.

# Thanks!

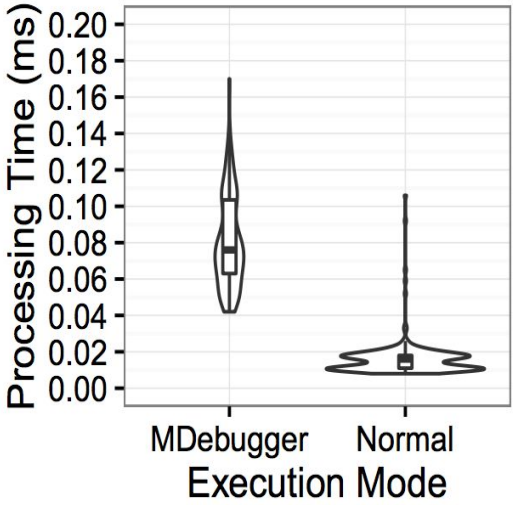
# Evaluation

**Performance:** We hypothesize that our approach relying on model instrumentation causes reasonable performance overhead, negligible enough so it can be applicable to RTE systems.

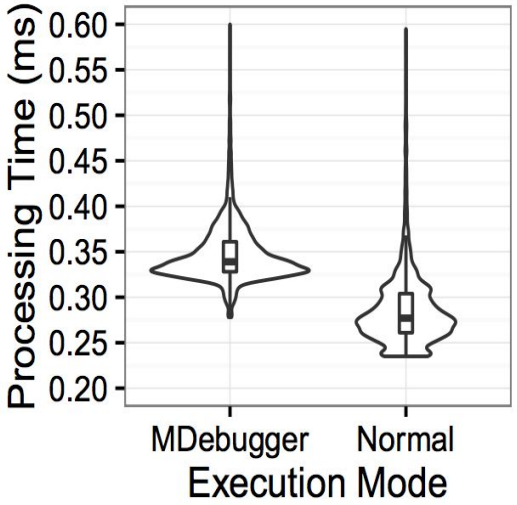
**Size Overhead:** While our approach based on model instrumentation undoubtedly increases the size of the generated code, we hypothesize that the size of the generated binary file is within the size range of binary files containing debugging symbols used by general-purpose debuggers.

# Performance


(a) RequestReply

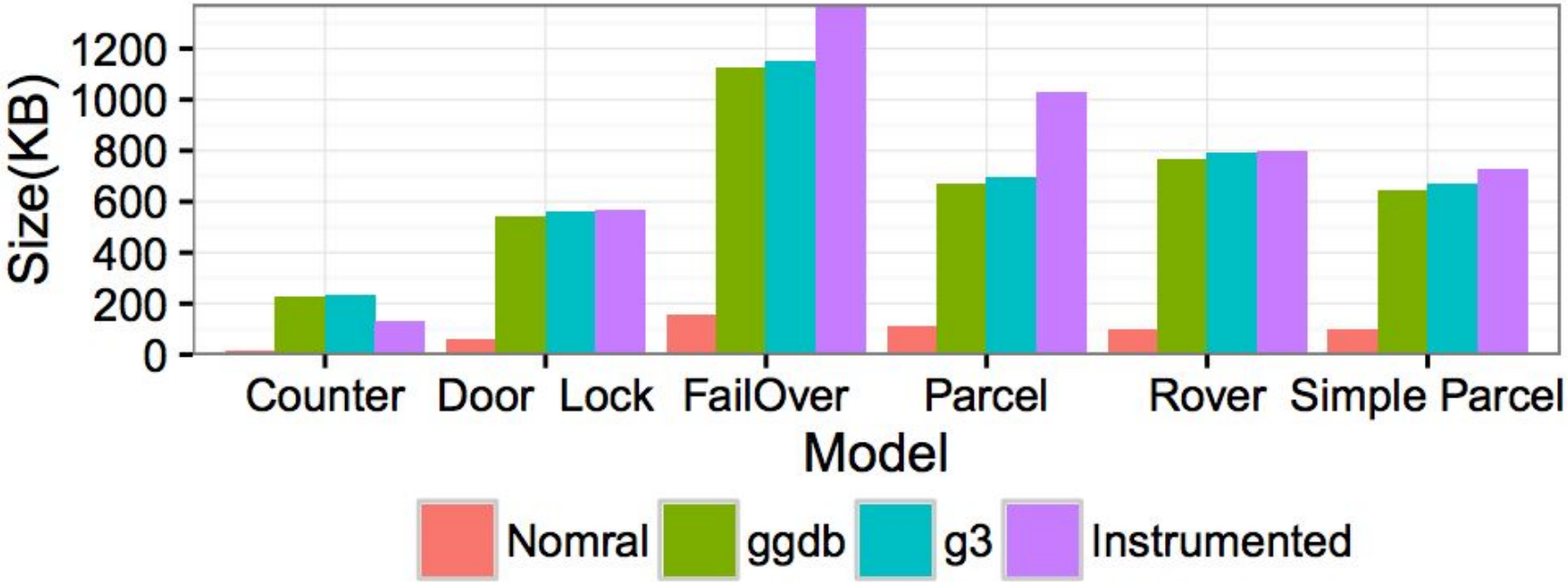



(b) SendKeepAlive




(c) ProcessResponse

# Size Overhead



# Stop and Resume