Refactoring Kieker's I/O Infrastructure to Improve Scalability and Extensibility

Holger Knoche

University of Kiel

November 10, 2017 @ SSP

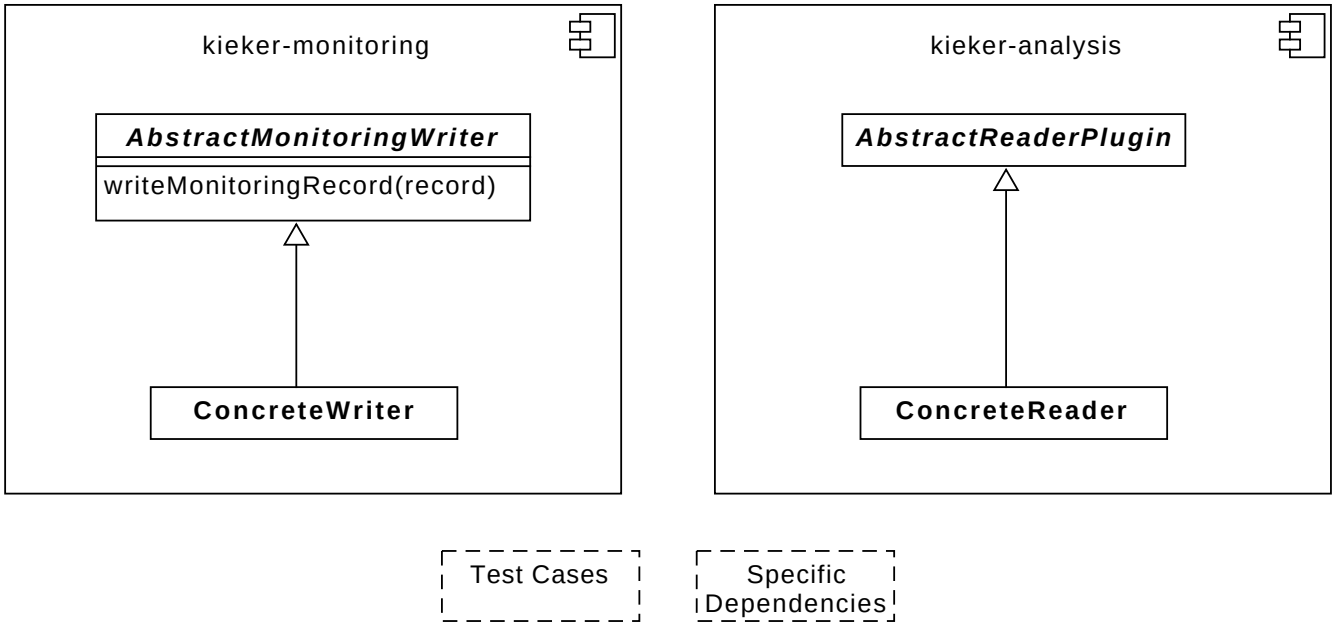
Agenda

1. Motivation
2. Kieker's I/O Infrastructure Today
3. Kieker's I/O Infrastructure Tomorrow
4. Performance Evaluation
5. Conclusions and Future Directions

Need for Scalability

- Large, scalable software systems produce large amounts of monitoring data
- Modern messaging solutions provide the basis for scalabe, distributed data processing
- **But:** Kieker's I/O infrastructure currently does not leverage the scalability of such platforms

Anatomy of a Reader-Writer Pair



- Reader and writer reside in separate components
- No place to put specific dependencies and test cases
- Data format hard-coded into the readers and writers

Default Binary Protocol

Kieker's default binary protocol uses string tables to avoid redundantly transferring string values.

1. All strings from a monitoring record are replaced by numeric IDs from the table
2. If a new string is encountered, a new ID is assigned and a special record is sent to update the consumer's table
3. The producer sends the encoded record
4. The consumer decodes the record using it's own copy of the string table

Problem: The string table is stateful.

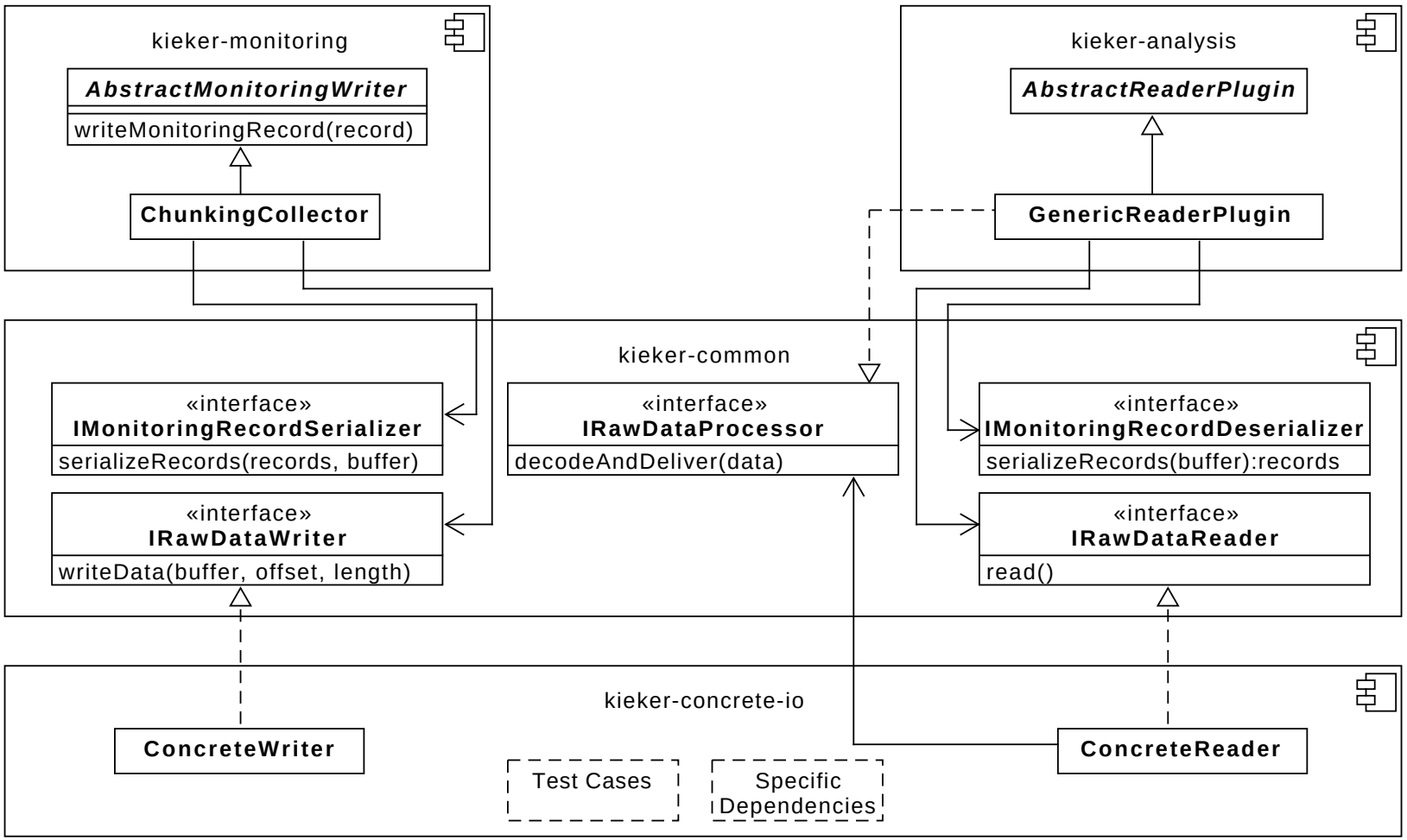
Identified Flaws / Mission Briefing

1. Stateful transfer protocol
   - Does not work with multiple record consumers
   - Prevents scalability
2. Data format is hard-coded into the writers
   - Little flexibility in terms of data format
3. Only one record is processed at a time
   - Increased overhead in messaging systems
4. Cumbersome separation of readers and writers
   - Leads to global dependencies
   - No place to put reader-writer tests

New Concepts

- **Collectors** collect multiple monitoring records and orchestrate the process of encoding and transferring the data
- **Serializers** and **Deserializers** encode monitoring records in a particular format and vice versa
- **Raw Data Readers** and **Writers** are responsible for tranferring the data using a particular medium

Reader-Writer Anatomy Revisited



1. Stateful transfer protocol
   - ✓ Fixed by a new, stateless container format
2. Data format is hard-coded into the writers
   - ✓ (De-)Serializers can be selected independent of the reader / writer
3. Only one record is processed at a time
   - ✓ Collectors allow to collect and batch-process multiple records
4. Cumbersome separation of readers and writers
   - ✓ Raw readers and writers (and their dependencies) can now be put into the same component

```
* Mission Accomplished *
```

But...

Major changes were made to Kieker's I/O Infrastructure, which may affect performance.

Performance Evaluation Questions:

EQ1 *Does the refactoring have a (negative) performance impact on the monitored application?*

EQ2 *To what extent does the chunking affect the overall resource consumption when using a messaging technology?*

EQ1: Application Performance

We...

- ...used MooBench
- ...used “null” writers that only did serialization
- ...ran everything on a Raspberry Pi 3
- ...evaluated four configurations

Configuration	**95% CI** (in μs)	σ
Baseline	[106.0;106.2]	21.0
Current infrastructure	[152.8;152.9]	28.3
Collector (no bypass)	[163.3;163.6]	64.9
Collector (bypass)	[141.5;141.6]	40.3

EQ2: System Performance

We...

- ...used a test harness issuing records at a constant rate
- ...measured CPU utilization by the harness process and overall network utilization
- ...ran the benchmarks on an otherwise idle system
- ...evaluated different chunk sizes

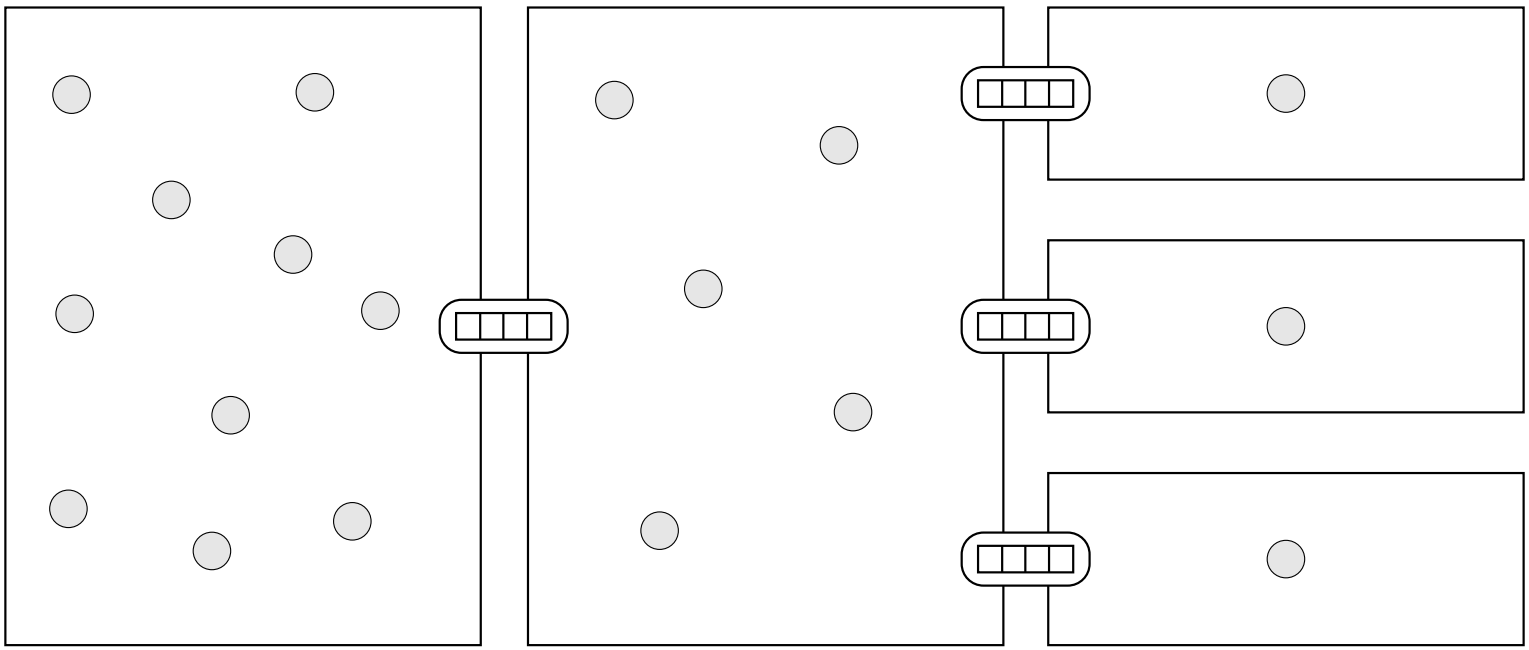
Chunk size	**95% CI CPU** in CPU sec. / sec.	**95% CI net** in KiB
Old writer	[0.612;0.620]	[1,913.7;1,914.2]
1	[0.768;0.780]	[2,706.4;2,710.3]
16	[0.460;0.477]	[684.2;684.3]
32	[0.273;0.275]	[639.9;640.1]
128	[0.340;0.356]	[593.8;594.2]
1024	[0.338;0.352]	[577.3;581.1]

```
* Mission Really
  Accomplished *
```

Conclusions

We...

- ...prepared Kieker to leverage the scalability of modern messaging infrastructures
- ...improved extensibility and flexibility along the way
- ...did not break anything in terms of performance
- ...even achieved performance improvements for message writers


Monitored Nodes
Trace Preprocessing / Filtering Nodes
Trace Processing Nodes