Trace Partitioning and Local Monitoring for Asynchronous Components

Duncan Attard[1] and Adrian Francalanza[1] · 8th September 2017

[1]CS, ICT, University of Malta, Malta

Outline

Introduction

Static pre-deployment techniques have limited applicability.

RV allows us to verify systems post-deployment

- it uses dedicated programs called **monitors**
- to analyse the **current execution** of system
- in a dynamic and **incremental** manner...

...**but** depends on the execution path taken by the system.

The traditional RV setup

Monitor generation

The traditional RV setup

Monitor generation
Property

The traditional RV setup

Monitor generation

Property $\rightarrow$ *compilation* $\rightarrow$

The traditional RV setup

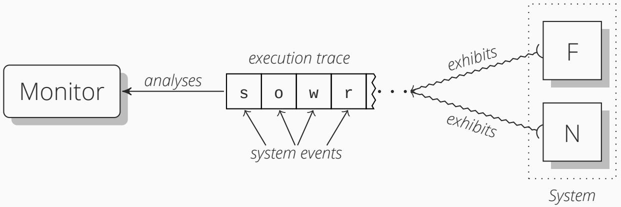
Monitor generation

Property $\rightarrow$ *compilation* $\rightarrow$ Monitor

The traditional RV setup

Monitor generation

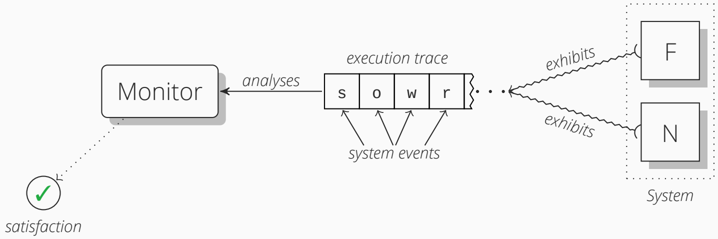
Property → *compilation* → Monitor



The traditional RV setup

Monitor generation

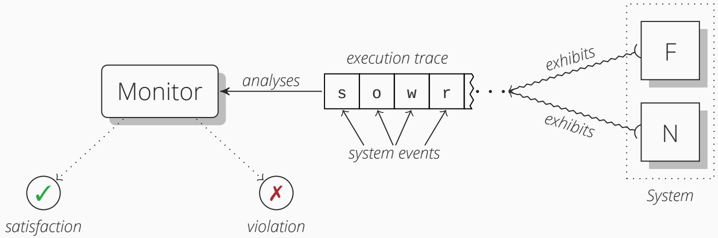
Property → *compilation* → Monitor



The traditional RV setup

Monitor generation

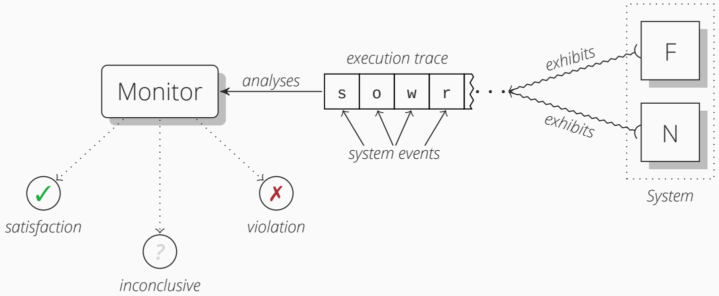
Property → *compilation* → Monitor



The traditional RV setup

Monitor generation

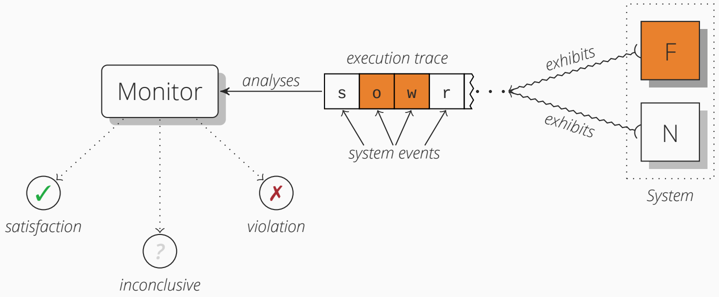
Property → *compilation* → Monitor



The traditional RV setup

Monitor generation

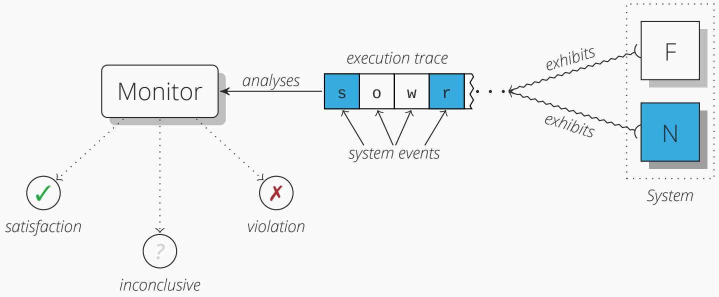
Property → *compilation* → Monitor



The traditional RV setup

Monitor generation

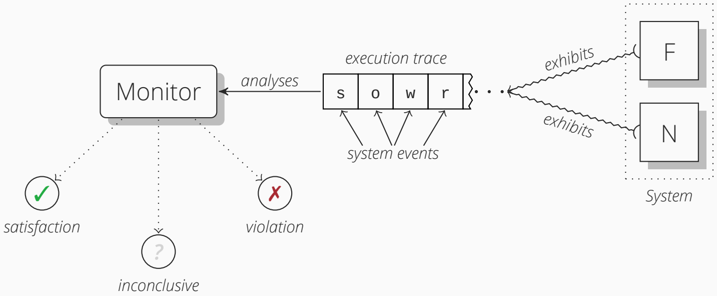
Property → *compilation* → Monitor



The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

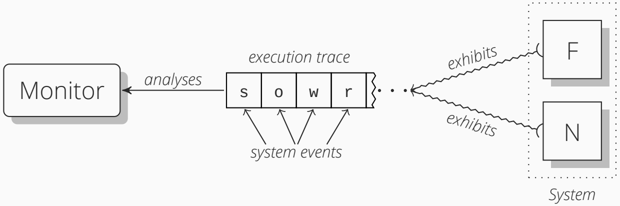


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

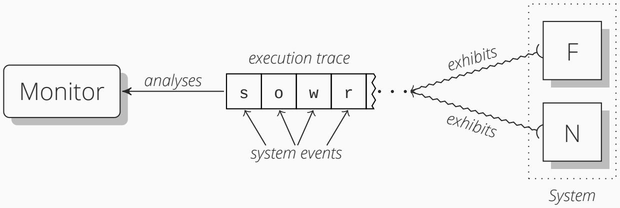


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

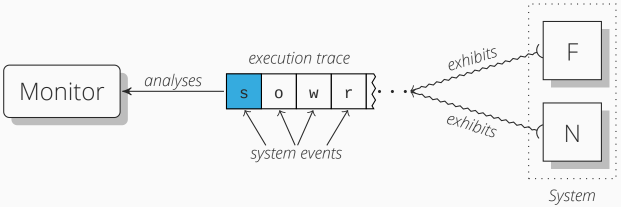


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

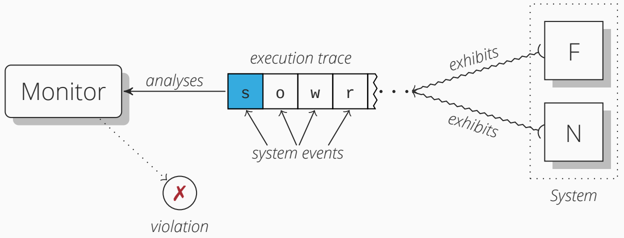


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

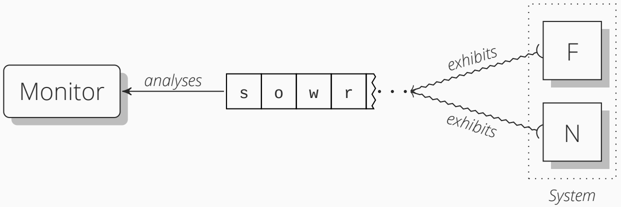


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

"The system cannot ***s*** *end messages at startup."*

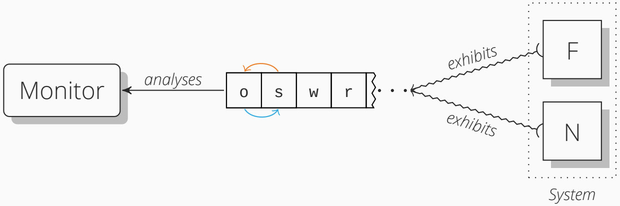


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

"The system cannot ***s***<u>*</u>*end messages at startup."*

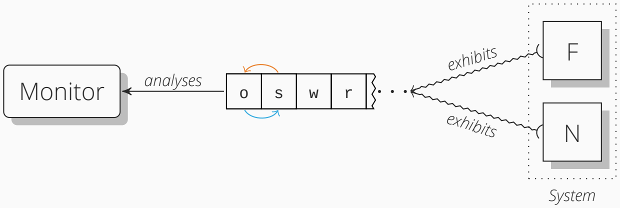


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

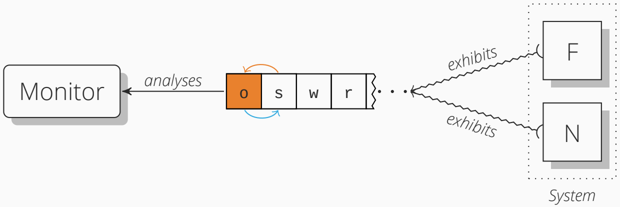


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

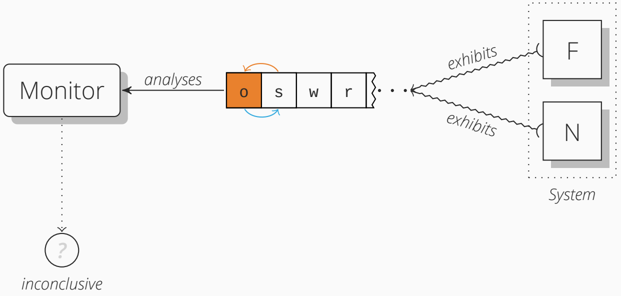


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

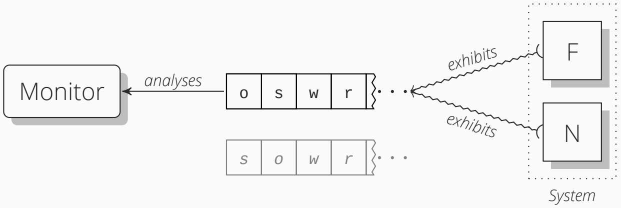


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

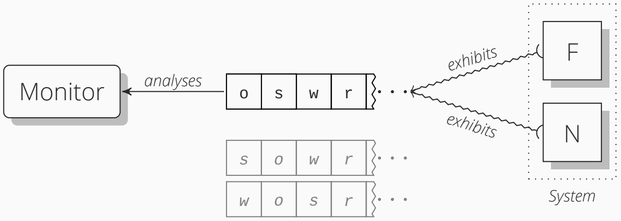


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

"The system cannot ***s****end messages at startup."*

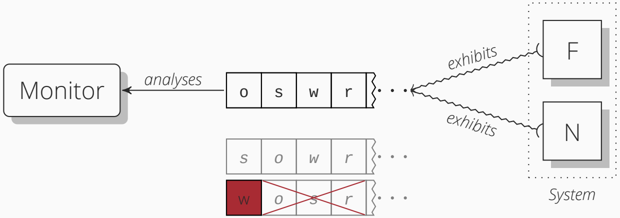


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

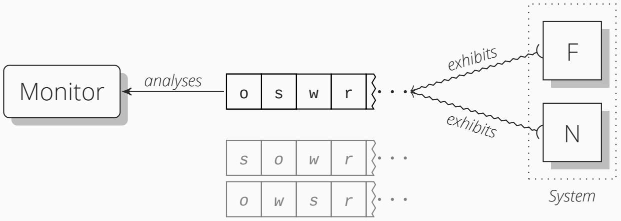


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

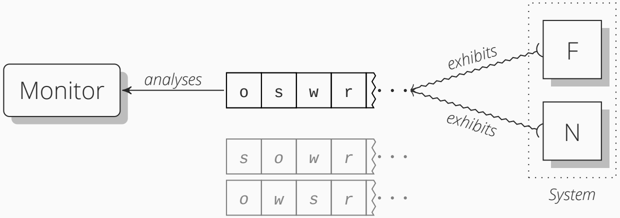


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

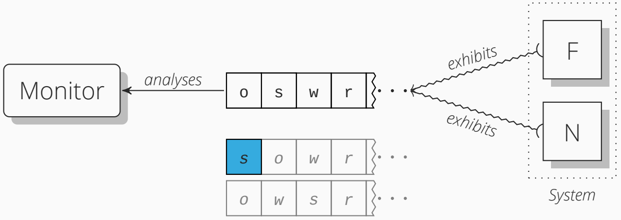


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

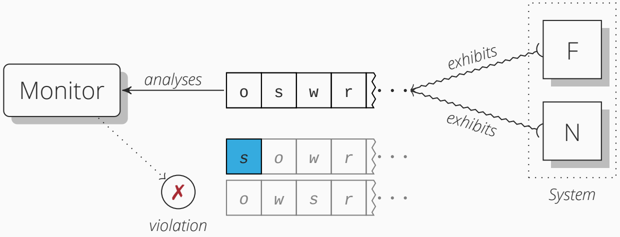


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"The system cannot **s**end messages at startup."*

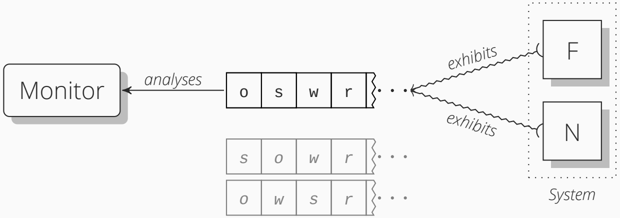


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

“At startup, the system cannot ***o**pen files* ***and*** ***s**end messages .”*

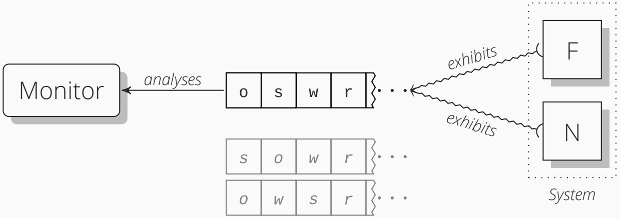


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*“At startup, the system cannot o̲pen files **and s̲**end messages .”*

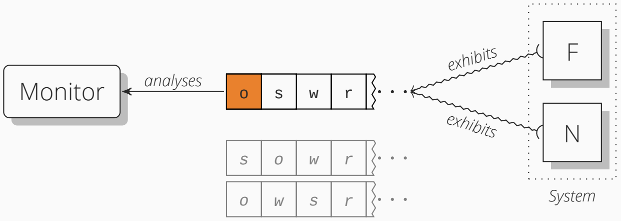


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

"At startup, the system cannot ***o****pen files* ***and*** ***s****end messages ."*

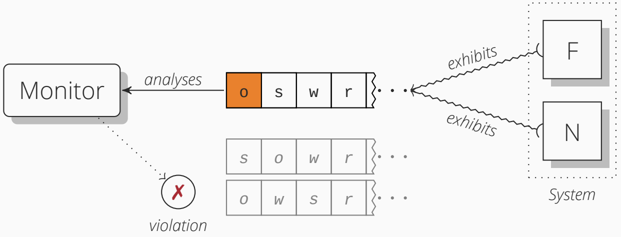


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"At startup, the system cannot **o**pen files* ***and*** *__s__end messages ."*

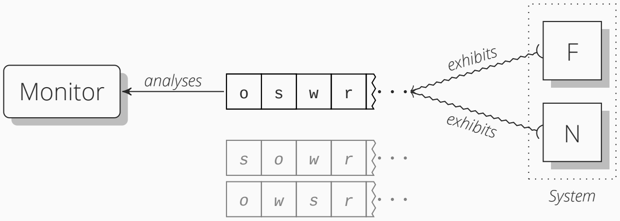


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"At startup, the system cannot **o**pen files **and** **s**end messages ."*

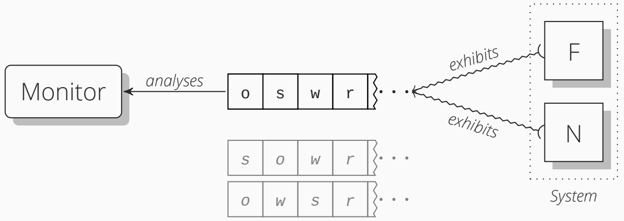


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

"At startup, the system cannot ***o****pen files* ***and*** *send messages .*"

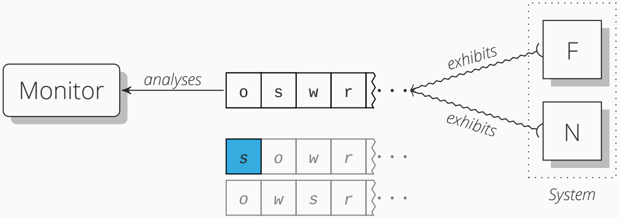


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"At startup, the system cannot **o**pen files **and** **s**end messages ."*

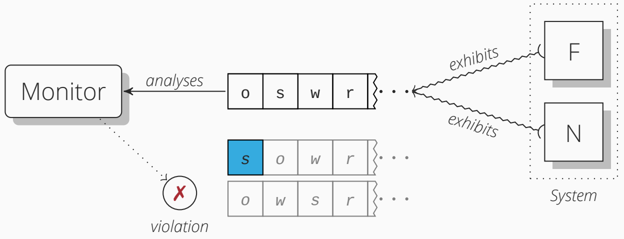


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"At startup, the system cannot **o**pen files* ***and*** *<u>s</u>end messages ."*

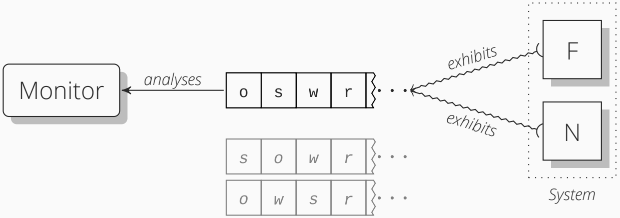


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"At startup, the system can neither **o**pen files **nor** **s**end messages."*

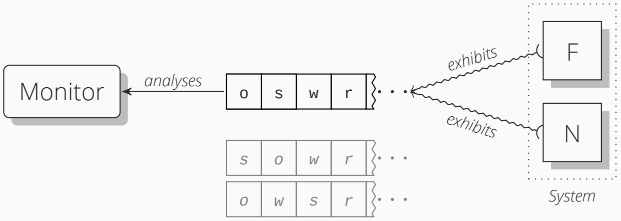


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

“At startup, the system can neither o<u>pen files</u> ***nor*** *send messages.”*

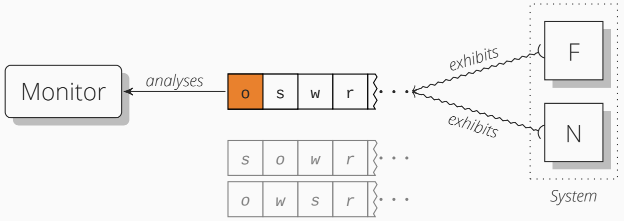


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

“At startup, the system can neither o̲pen files ***nor*** *s̲end messages.”*

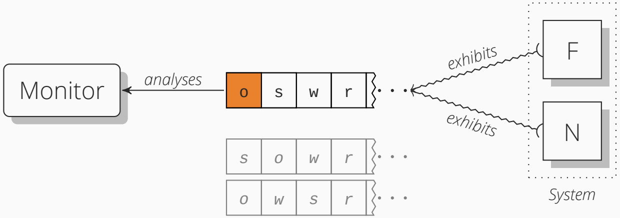


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"At startup, the system can neither <u>**o**</u>pen files **nor** <u>**s**</u>end messages."*

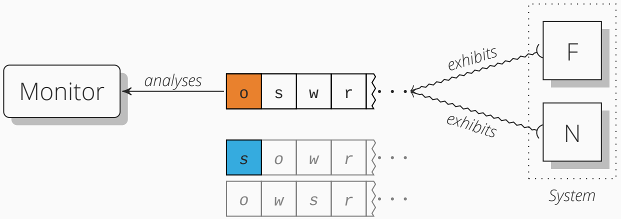


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"At startup, the system can neither **o**pen files **nor** **s**end messages."*

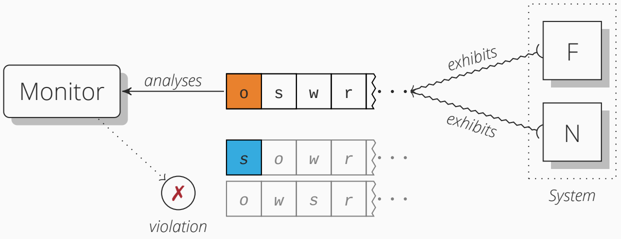


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

*"At startup, the system can neither open files **nor** send messages."*

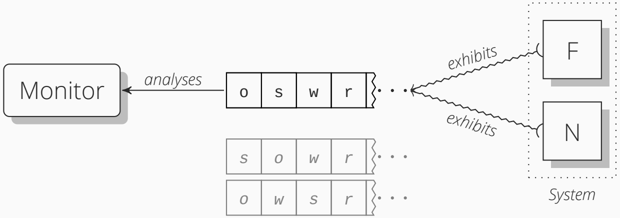


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

“Message sending cannot issue any ***e****rrors.”*

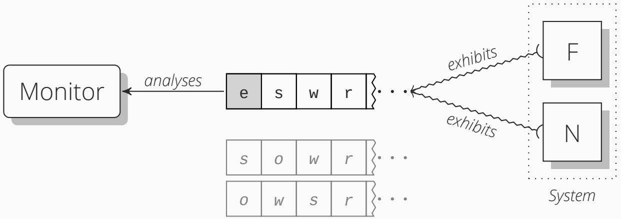


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

"Message sending cannot issue any ***e**rrors."*

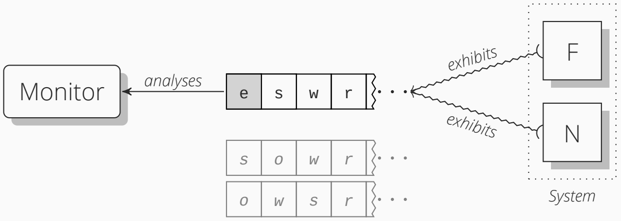


The traditional RV setup

Monitor generation

Property → *compilation* → Monitor

“Message sending cannot issue any ***e****rrors.”…?*

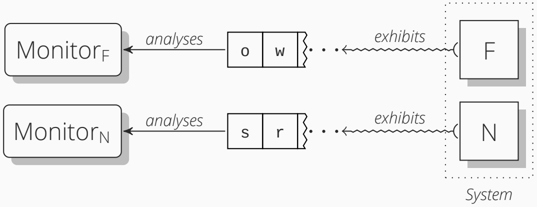


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

Stop treating the system as one monolithic block...

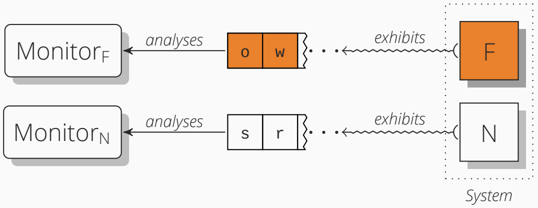


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

Stop treating the system as one monolithic block...

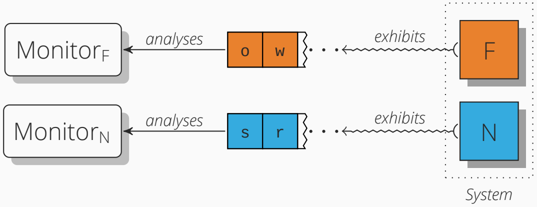


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

Stop treating the system as one monolithic block...

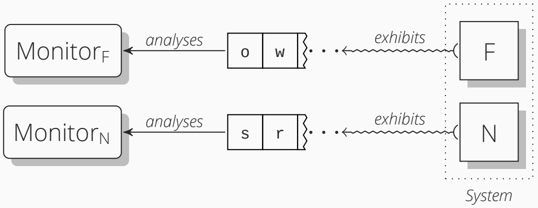


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

Stop treating the system as one monolithic block...

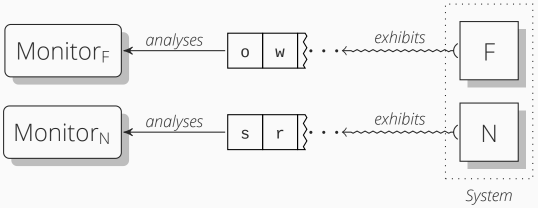


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"The system cannot **s**end messages at startup."*

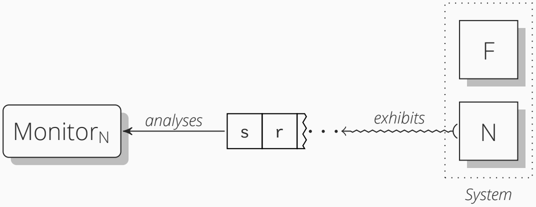


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"The system cannot **s**end messages at startup."*

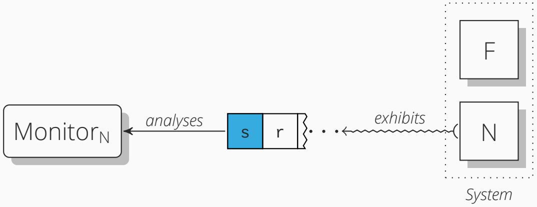


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"The system cannot **s**end messages at startup."*

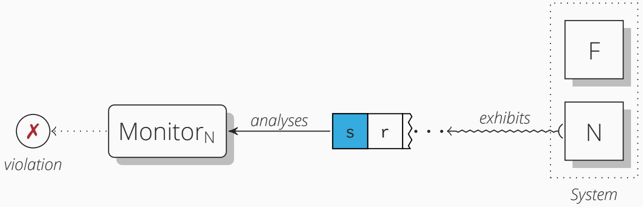


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"The system cannot **s**end messages at startup."*

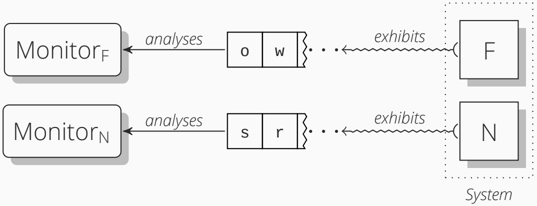


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

"At startup, the system cannot ***o****pen files* ***and s****end messages."*

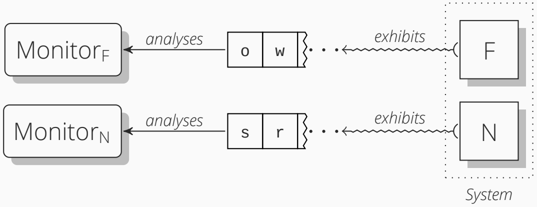


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system cannot o̲pen files **and** s̲end messages."*

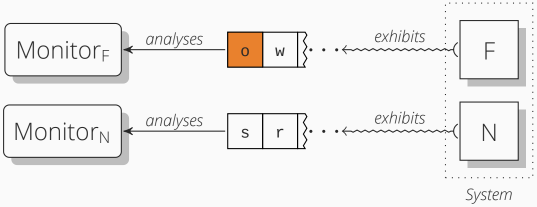


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system cannot open files **and** send messages."*

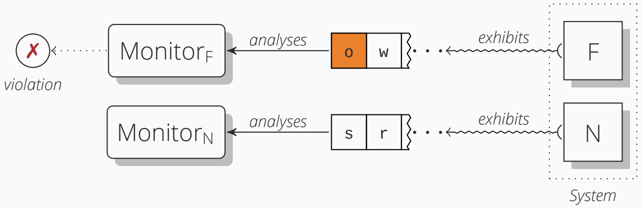


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system cannot <u>**o**</u>pen files **and** <u>**s**</u>end messages."*

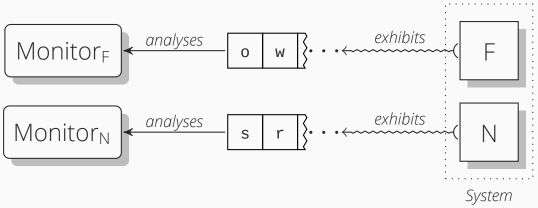


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

"At startup, the system cannot ***o****pen files* ***and s****end messages."*

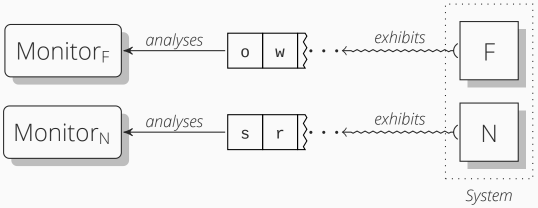


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

"At startup, the system cannot ***o****pen files* ***and*** ***s****end messages."*

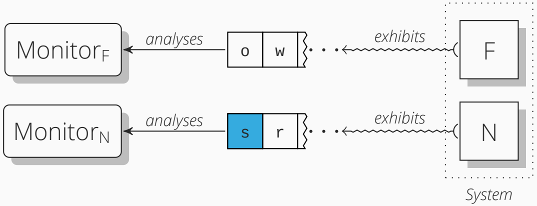


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system cannot **o**pen files **and** **s**end messages."*

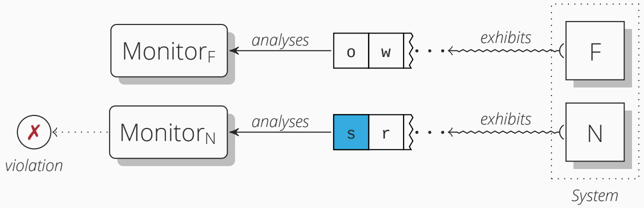


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system cannot **o**pen files **and** **s**end messages."*

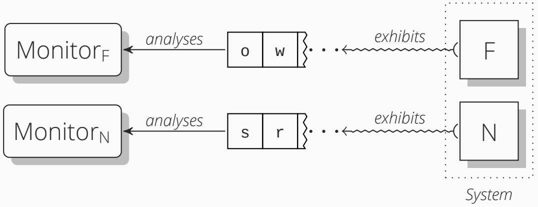


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

"At startup, the system can neither **o**_pen files_ ***nor*** **s**_end messages."_

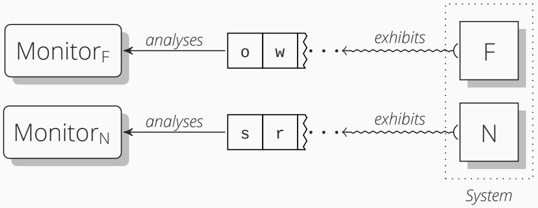


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system can neither <u>o</u>pen files **nor** <u>s</u>end messages."*

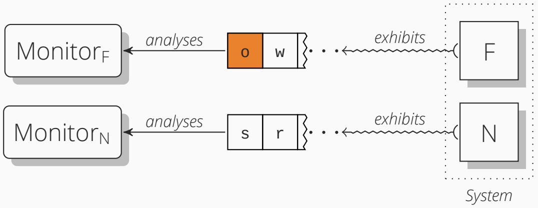


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system can neither open files **nor** send messages."*

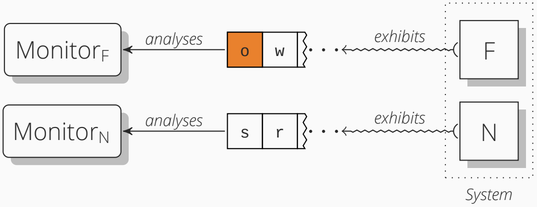


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system can neither <u>o</u>pen files **nor** <u>s</u>end messages."*

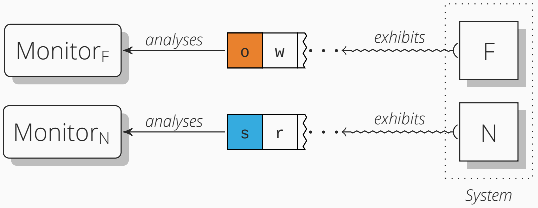


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system can neither **o**pen files **nor** **s**end messages."*

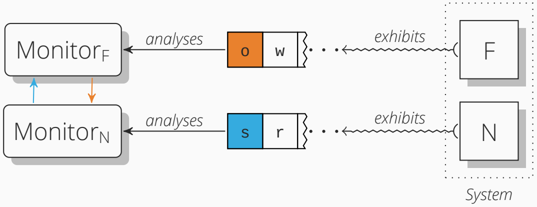


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system can neither **o**pen files **nor** **s**end messages."*

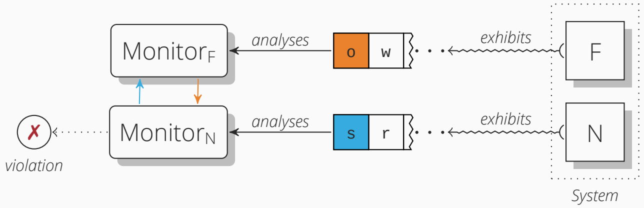


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system can neither open files **nor** send messages."*

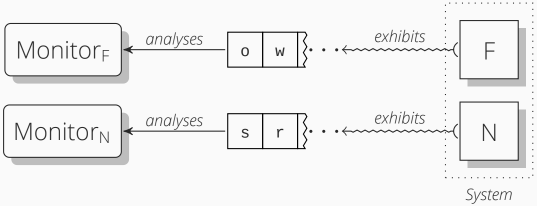


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

"Message sending cannot issue any ***e****rrors."*

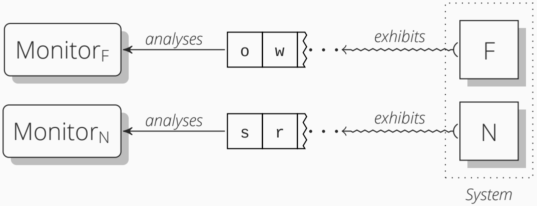


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

"Message sending cannot issue any errors."

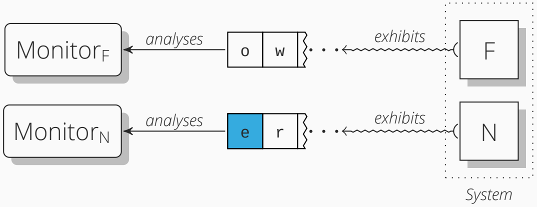


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

“Message sending cannot issue any errors.”

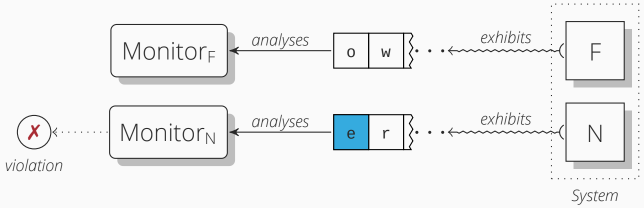


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"Message sending cannot issue any **e**rrors."*

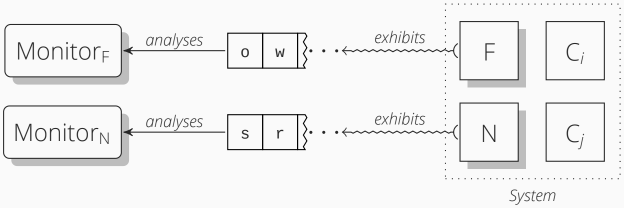


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

"Message sending cannot issue any ***e****rrors."*

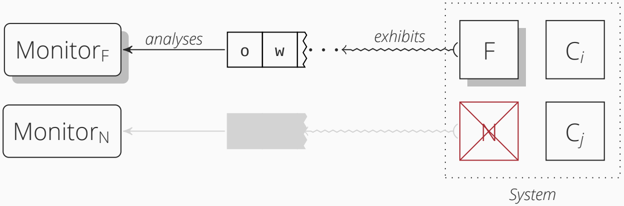


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"Message sending cannot issue any **e**rrors."*

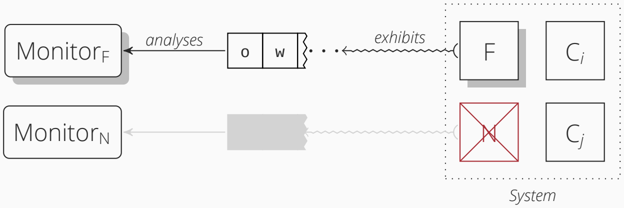


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system cannot <u>**o**</u>pen files **and** <u>**s**</u>end messages."*

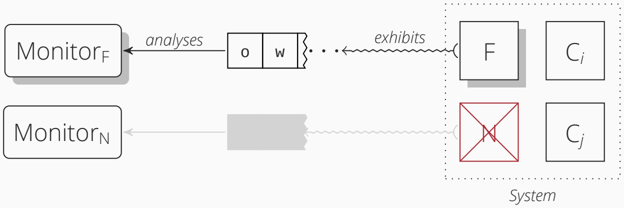


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

"At startup, the system cannot ***o****pen files* ***and s****end messages."*

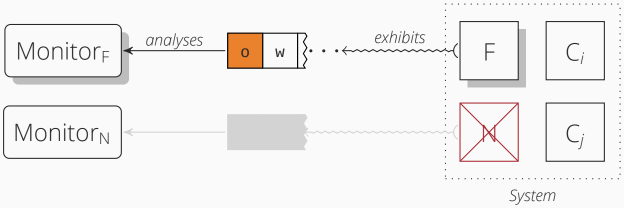


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system cannot **o**pen files **and** **s**end messages."*

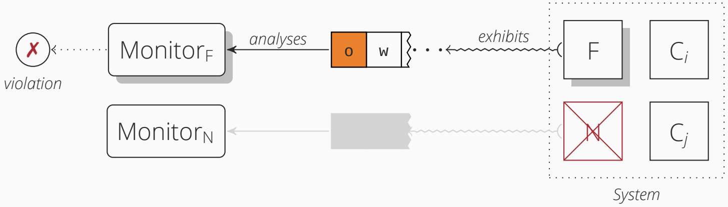


Partitioned traces

Approach valid but expensive

- We need an unbounded buffer to store inferred traces.
- Inferring new traces requires computational resources...

*"At startup, the system cannot <u>o</u>pen files **and** <u>s</u>end messages."*



Tracing vs. monitoring

Tracing differs from monitoring

- Trace partitioning operates at the **instrumentation** level…
  …and deals with **what** events are extracted from **where**.
- Monitors deal with **how** these events are analysed.

Tracing vs. monitoring

Tracing differs from monitoring

- Trace partitioning operates at the **instrumentation** level… …and deals with **what** events are extracted from **where**.
- Monitors deal with **how** these events are analysed.

Component monitors are but one method of trace analysis.

Tracing vs. monitoring

Tracing differs from monitoring

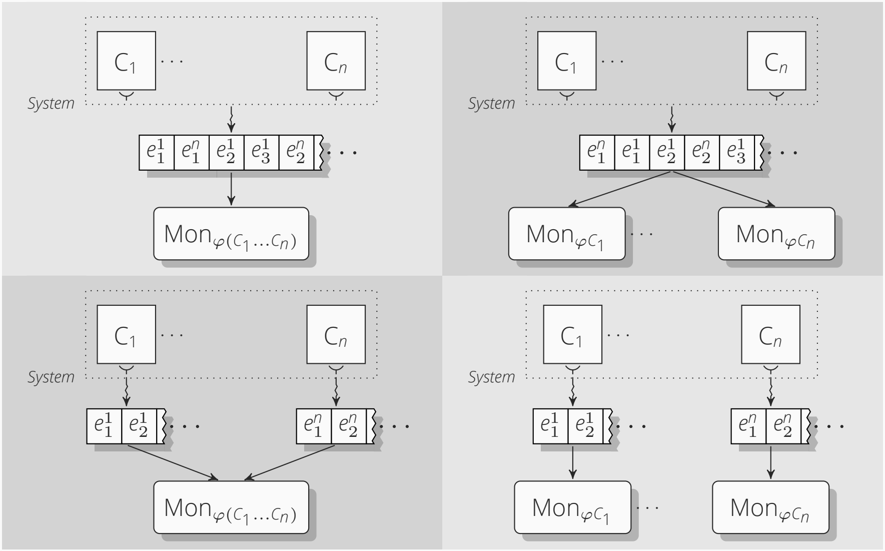
- Trace partitioning operates at the **instrumentation** level… …and deals with **what** events are extracted from **where**.
- Monitors deal with **how** these events are analysed.

Component monitors are but one method of trace analysis.

This separation of concerns is nice:

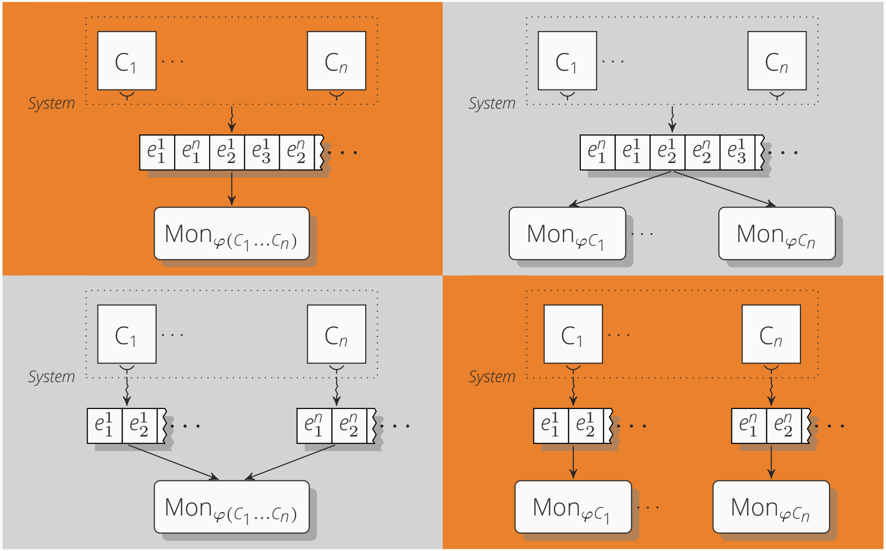
- it allows us to reason separately on both;
- it makes this reasoning manageable.

What and where, how...



System
C_1
C_n
e^1_1 e^n_1 e^1_2 e^1_3 e^n_2
$\text{Mon}_{\varphi(C_1 \ldots C_n)}$
System
C_1
C_n
e^n_1 e^1_1 e^1_2 e^n_2 e^1_3
$\text{Mon}_{\varphi C_1}$
$\text{Mon}_{\varphi C_n}$
System
C_1
C_n
e^1_1 e^1_2
e^n_1 e^n_2
$\text{Mon}_{\varphi(C_1 \ldots C_n)}$
System
C_1
C_n
e^1_1 e^1_2
e^n_1 e^n_2
$\text{Mon}_{\varphi C_1}$
$\text{Mon}_{\varphi C_n}$

What and where, how...



C_1 ⋯ C_n
System
e^1_1 e^n_1 e^1_2 e^1_3 e^n_2 ⋯
$\mathrm{Mon}_{\varphi(C_1\ldots C_n)}$
C_1 ⋯ C_n
System
e^n_1 e^1_1 e^1_2 e^n_2 e^1_3 ⋯
$\mathrm{Mon}_{\varphi C_1}$ ⋯ $\mathrm{Mon}_{\varphi C_n}$
C_1 ⋯ C_n
System
e^1_1 e^1_2 ⋯
e^n_1 e^n_2 ⋯
$\mathrm{Mon}_{\varphi(C_1\ldots C_n)}$
C_1 ⋯ C_n
System
e^1_1 e^1_2 ⋯
e^n_1 e^n_2 ⋯
$\mathrm{Mon}_{\varphi C_1}$ ⋯ $\mathrm{Mon}_{\varphi C_n}$

The advantages of trace partitioning

1. **compactly encodes** interleaving behaviour,
2. inherently **filters** extraneous events from a partition,
3. can be **selectively applied** to components of interest,
4. is easily applicable to non-replicated components,
5. **preserves the origin** of indistinguishable events,
6. may be naturally extended to distributed settings.

The advantages of trace partitioning

1. **compactly encodes** interleaving behaviour,
2. inherently **filters** extraneous events from a partition,
3. can be **selectively applied** to components of interest,
4. is easily applicable to non-replicated components,
5. **preserves the origin** of indistinguishable events,
6. may be naturally extended to distributed settings.

Outline

1. Monitoring Asynchronous Component Systems
2. Trace Partitioning and Local Monitoring in Practice

An empirical evaluation of local monitoring

The study was conducted on a third-party app called Ranch.

An empirical evaluation of local monitoring

The study was conducted on a third-party app called Ranch.

Aim: to test local monitoring in a **real-world** setting and

1. explore its **applicability** in industry-level apps,
2. investigate its **feasibility** in terms of performance.

An empirical evaluation of local monitoring

The study was conducted on a third-party app called Ranch.

Aim: to test local monitoring in a **real-world** setting and

1. explore its **applicability** in industry-level apps,
2. investigate its **feasibility** in terms of performance.

Focus: global properties that can be written as local ones.

An empirical evaluation of local monitoring

The study was conducted on a third-party app called Ranch.

Aim: to test local monitoring in a **real-world** setting and

1. explore its **applicability** in industry-level apps,
2. investigate its **feasibility** in terms of performance.

Focus: global properties that can be written as local ones.

Method: identified a set of components within Ranch and:

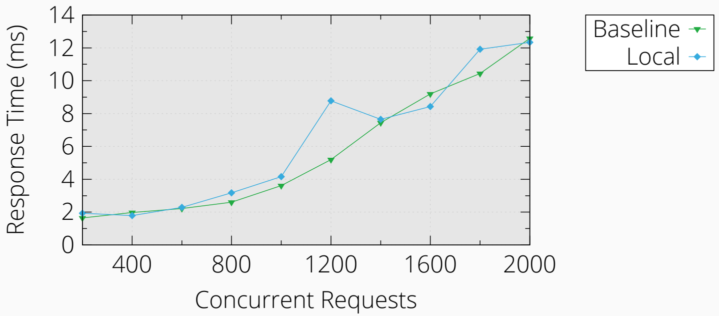
1. formulated a number of global properties,
2. monitored the components using global monitors,
3. reformulated the global properties as local ones,
4. monitored the same components using local monitors.

Experiments and results

100 components (recommended number for production use).

Global monitoring **consistently** led to resource exhaustion.

Local monitoring scaled well in relation to the baseline.

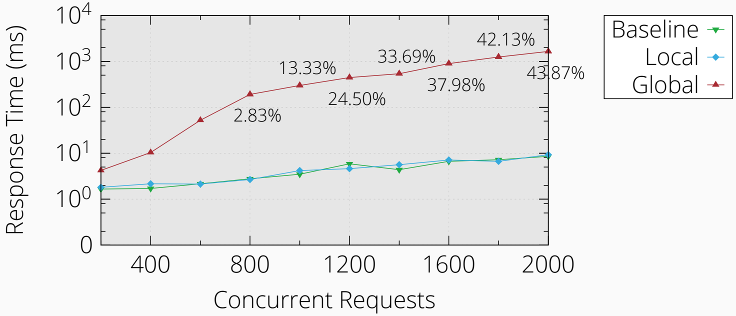


Experiments and results

< 5 components (used to be able to test global monitoring).

Global monitoring scaled poorly.

Local monitoring scaled well in relation to the baseline.



Conclusion

Trace partitioning $\Rightarrow$ maximising information

- Extra **information** is extracted from the system execution.
- This helps **improve** the **monitor analysis**.

Conclusion

Trace partitioning $\Rightarrow$ maximising information

- Extra **information** is extracted from the system execution.
- This helps **improve** the **monitor analysis**.

Local monitoring

- The instrumentation technique **fits local monitors** naturally.
- Local monitoring + partitioned traces = **lower overheads**.

Thank you