Asynchronous Shared Data Sources

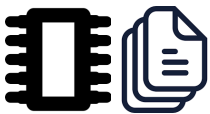
Mart Lubbers (✉) Haye Böhm Pieter Koopman Rinus Plasmeijer
{mart,pieter,rinus}@cs.ru.nl
haye.bohm@gmail.com

LambdaDays 2021, 16–19 February 2021
TFP 2021, 17–19 February 2021

Radboud University

Data Sources

Data Sources



- Read
- Write
- Update
- Lens
- Fuse
- . . .
- Notify
- Share

Shared Data Sources (SDSs)

Task Oriented Programming

Task Oriented Programming

Task Oriented Programming (TOP)

- Coordinate collaboration between people and machines to reach common goal.
- Declarative paradigm
- iTasks, mTasks, $\widehat{TOP}$
- Tasks are the basic building block
- Communication via Task Values and SDSs
- iTasks: event driven state transformers

SDSs in the iTask system

SDSs in iTasks

Programmer's interaction with shares

- get/watch
- set
- upd

...

User interaction with shares

- viewSharedInformation
- updateSharedInformation

...

System's reliance on shares

- events

$$
\begin{aligned}
&:: \mathit{Person} = \{ \quad \mathit{name} \quad :: \mathit{String} \\
&\qquad\quad , \quad \mathit{age} \quad :: \mathit{Int} \\
&\qquad\quad , \quad \mathit{gender} \quad :: \mathit{Gender} \\
&\qquad\quad \} \\
&:: \mathit{Gender} = \\
&\qquad \mathit{Male} \mid \mathit{Female} \mid \mathit{Other}\ \mathit{String}
\end{aligned}
$$

History

- Uniform Data Sources (Submitted for TFP 2012)
- Shared Data Sources (iTask system)
- Parametric Lenses (IFL 2014)
- Parametric Shared Data Sources (iTask system)

Limitations:

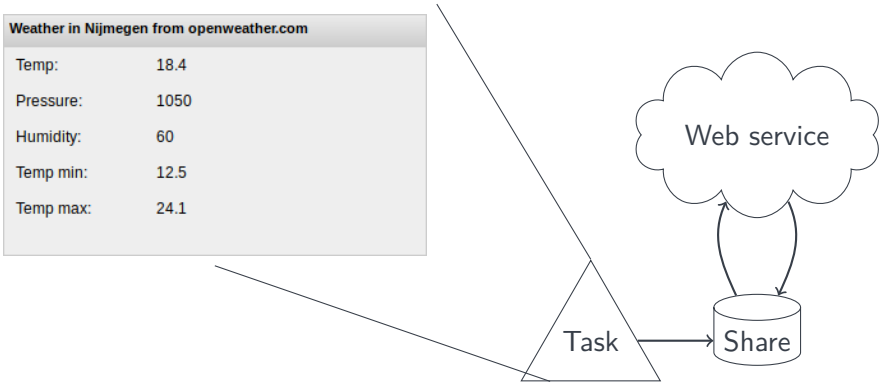
- Single thread
- Blocking
- Strict constraints
- Unsuitable for the real world

- Class based Shared Data Sources (MSc. Haye Böhm, this paper)
- Asynchronous Shared Data Sources (MSc. Haye Böhm, this paper)

Practical SDS use

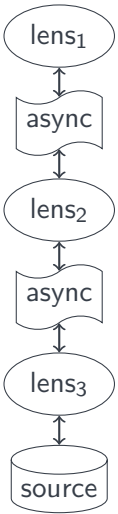
Practical use: Slow network data

- MySQL asynchronous interface
- Web services
- OS specific waits (select, poll)
- ...



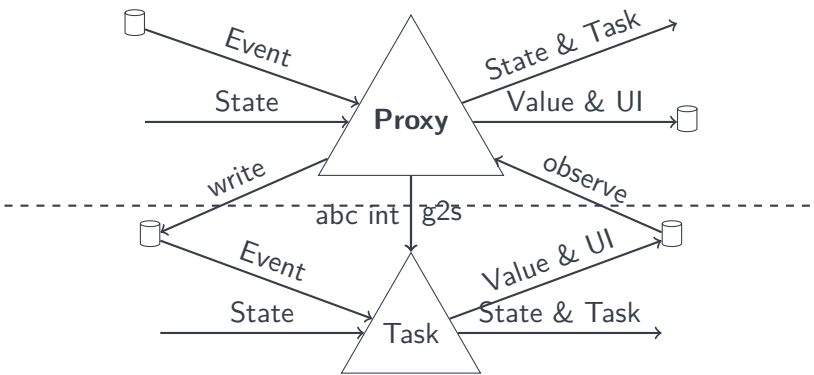
Practical use: Remote shares

- Shares on a different machine
- Combinations of shares



Practical use: Asynchronous tasks

- Blocking tasks
- Tasks on different machines
- Tasks on different processes
- Communication via shares
- Tap directly into system shares



SDSs in general

Class based asynchronous SDS Operations

In general

$:: PViewT\ m\ a = \ldots$ $:: Source\ m\ p\ r\ w = \ldots$

$:: Lens\ sds\ m\ p\ r\ w = \ldots$ $:: Pair\ sdsl\ sdsr\ m\ p\ r\ w = \ldots$

Lenses

$get \quad \in X \to Y$

$put \quad \in Y \times X \to X$

Parametrised Lenses

$get \quad \in \Phi \times X \to Y$

$put \quad \in \Phi \times Y \times X \to X \times (\Phi \to Bool)$

read

class $read\ v :: (v\ m\ p\ r\ w)\ p \to PViewT\ m\ r(ReadResult\ m\ p\ r\ w)$

$:: ReadResult\ m\ p\ r\ w = \quad Read\ r$

$\quad \mid \exists sds{:}\ Reading\ (sds\ m\ p\ r\ w)\ \&\ read\ sds$

write

class $write\ v :: (v\ m\ p\ r\ w)\ p\ w \to PViewT\ m\ ()(WriteResult\ m\ p\ r\ w)$

$:: WriteResult\ m\ p\ r\ w = \quad Written\ ()$

$\quad \mid \exists sds{:}\ Writing\ (sds\ m\ p\ r\ w)\ \&\ write\ sds$

Conclusion

Conclusion & Discussion

Conclusion

- Asynchronous Reads
- Asynchronous Writes
- Atomic updates*
- Used as we speak in the iTask system†

Discussion

- Complicated types‡ for the compiler or we box them
- :: $SDS\ m\ p\ r\ w = \exists sds\text{: }SDS\ (sds\ m\ p\ r\ w)\ \&\ read\ sds\ \&\ write\ sds$

Questions?