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, \( \text{TOP} \)
- 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
- process lists

:: Person = { name :: String, age :: Int, gender :: Gender }
:: Gender = Male | Female | Other String

:: Person = { name :: String, age :: Int, gender :: Gender }
:: Gender = Male | Female | Other String

:: Person = { name :: String, age :: Int, gender :: Gender }
:: Gender = Male | Female | Other String
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)
- ...

Weather in Nijmegen from openweather.com

- Temp: 18.4
- Pressure: 1050
- Humidity: 60
- Temp min: 12.5
- Temp max: 24.1

Task ➔ Web service ➔ Share
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:

\[ \text{PViewT} \quad m \quad a = \ldots \]
\[ \text{Source} \quad m \quad p \quad r \quad w = \ldots \]
\[ \text{Lens} \quad sds \quad m \quad p \quad r \quad w = \ldots \]
\[ \text{Pair} \quad sdsl \quad sdsr \quad m \quad p \quad r \quad w = \ldots \]

Lenses

\[ \text{get} \quad \in \quad X \rightarrow Y \]
\[ \text{put} \quad \in \quad Y \times X \rightarrow X \]

Parametrised Lenses

\[ \text{get} \quad \in \quad \Phi \times X \rightarrow Y \]
\[ \text{put} \quad \in \quad \Phi \times Y \times X \rightarrow X \times (\Phi \rightarrow \text{ Bool}) \]

read

**class** read \( v :: (v \ m \ p \ r \ w) \ p \rightarrow \text{PViewT} \ m \ r(\text{ReadResult} \ m \ p \ r \ w) \)
\[ \text{ReadResult} \ m \ p \ r \ w = \quad \text{Read} \ r \]
\[ \quad \mid \exists sds: \text{Reading} \ (sds \ m \ p \ r \ w) \ & \ \text{read} \ sds \]

write

**class** write \( v :: (v \ m \ p \ r \ w) \ p \ w \rightarrow \text{PViewT} \ m \ ()(\text{WriteResult} \ m \ p \ r \ w) \)
\[ \text{WriteResult} \ m \ p \ r \ w = \quad \text{Written} \ () \]
\[ \quad \mid \exists sds: \text{Writing} \ (sds \ m \ p \ r \ w) \ & \ \text{write} \ sds \]
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: SDS (sds m p r w) & read sds & write sds

Questions?