



Crawling the Web with Akka HTTP

Christoph Knabe
Dpt. Informatics and Media
07.11.2017



Content

- Bio
- Motivation
- Technologies used
- Communication between actors
- Source Code, Demonstration
- Lessons learned
- Measure Results
- Questions





Biography

- 1981-1990 Software Developer @ PSI GmbH:
Factory Automation, Software Engineering Tools
- 1990-... Prof. @ Beuth University of Applied Sciences
Berlin:
Teaches Software Engineering, Programming
Main interests: Scala, Backend Development





Motivation

- Learned in Coursera course „Reactive Programming“: Massive parallelization only possible by asynchronous, event-oriented processing, not by threaded processing.
- Wanted to demonstrate it for my students, who do not own big servers in the web.
- \Rightarrow Study massive parallelization on a web client.
- App: **Web Crawler**, which gets hold of as many working web page addresses as possible.





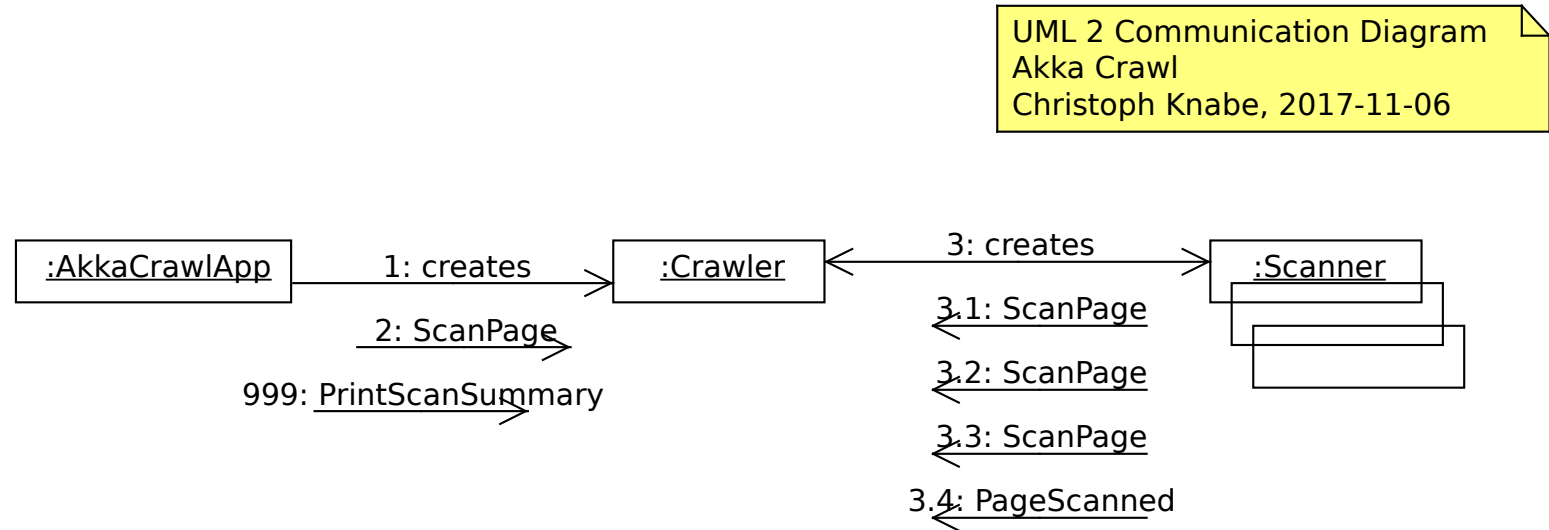
Technologies used

- **Akka Actors** for managing the web crawling, and the web page scanning.
- **Akka HTTP** for sending a page request, and receiving a Stream with the HTML text.
- **Akka Streams** for processing each web page.
- **Scala Futures** for
 - piping an `HttpResponse` into the `Crawler Actor`,
 - awaiting the actor system termination.





Communication



- See this in source code.
- Run it.



Lessons learned

- **Indeterministic:**
 - Run it often
 - Run it in different environmentsin order to understand the behavior.
- **Actors, Futures, Streams:** Fully interoperable!!!
No problems with this.
- **Always set Timeout** in order not to collect uncompleted actors:
`context.setReceiveTimeout(responseTimeout)`
- Use property `akka.loglevel` in the beginning,
in the end switch to `off`.



Measure Results

- **At home (Laptop@DSL):**
 - Scanned 318 pages in 1 minute.
 - About 10 unprocessed `ScanPage` commands.
 - Page scans get slower and slower.
- **In University WLAN (Laptop@WLAN):**
 - Scanned 2.917 pages in 1 minute
 - About 390 unprocessed `ScanPage` commands.
- **On University Server:**
 - Scanned 6.480 pages in 1 minute
 - About 390 unprocessed `ScanPage` commands.



Questions

- What is the limiting factor?
 - CPU
 - Network bandwidth
 - Number of open ports
 - JVM-RAM, ...
- Would a streamed solution (with backpressure) work better?
- Can I start an Akka HTTP request from a stream?
- Can I feed multiple response streams into one crawler stream?





Thank You