

Зачем было тащить

Undertow

Григорий
Кошелев
Контур

План

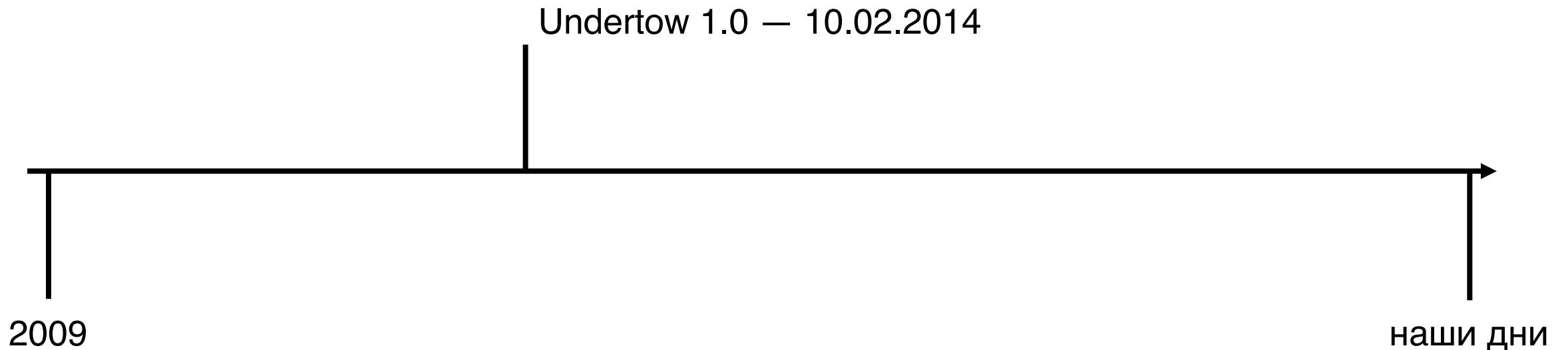
- Историческая справка
- Архитектура Undertow
- IO-intensive приложения
- Undertow + XNIO
- Производительность
и оптимизации

Историческая справка



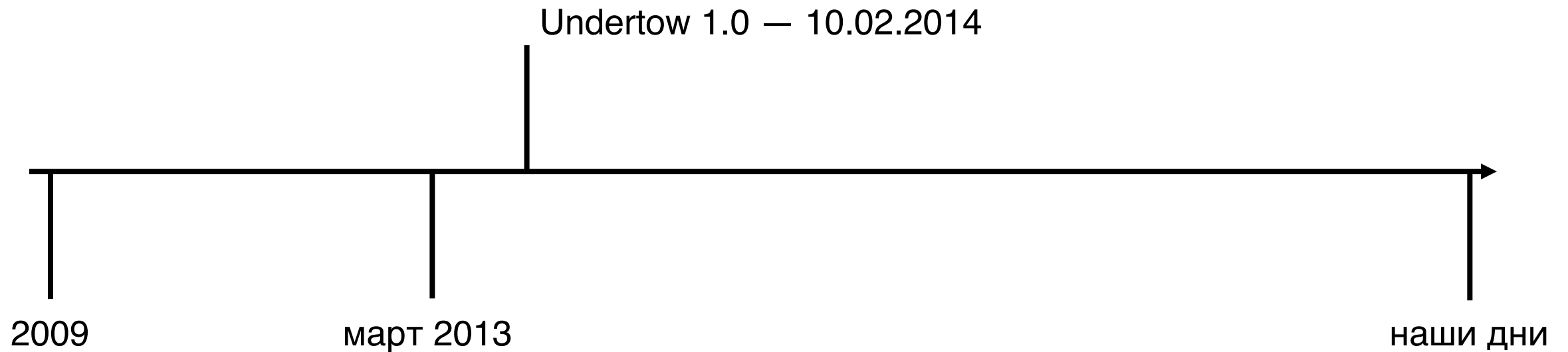
Историческая справка

Undertow 1.0 — 10 февраля 2014



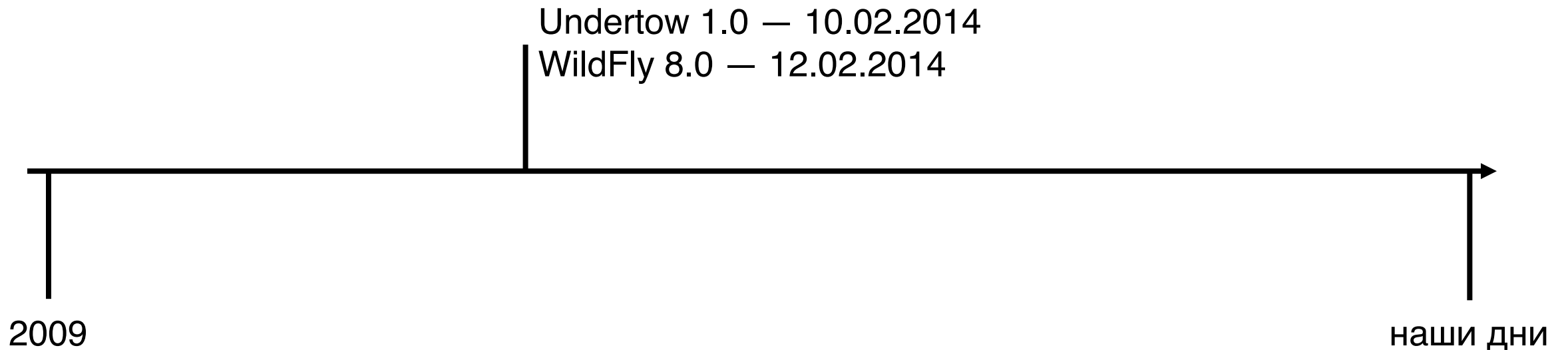
Историческая справка

Undertow 1.0 — 10 февраля 2014



Историческая справка

WildFly 8.0 — 12 февраля 2014



WildFly 8.0

Undertow, the new cutting-edge web server in WildFly 8, is designed for maximum throughput and scalability, including environments with over a million connections. It supports non-blocking and blocking handlers, traditional and asynchronous servlets, and JSR-356 web socket handlers. It is highly customizable, with the ability for applications to implement nearly anything from dynamic request routing to custom protocols. It can also function as a very efficient, pure non-blocking reverse proxy, allowing WildFly to delegate to other web servers with minimal impact to running applications. This release adds numerous improvements including greater extensibility and enhanced security capabilities.

WildFly 8.0

Undertow, the new cutting-edge web server in WildFly 8, is **designed for maximum throughput and scalability, including environments with over a million connections**. It supports non-blocking and blocking handlers, traditional and asynchronous servlets, and JSR-356 web socket handlers. It is highly customizable, with the ability for applications to implement nearly anything from dynamic request routing to custom protocols. It can also function as a very efficient, pure non-blocking reverse proxy, allowing WildFly to delegate to other web servers with minimal impact to running applications. This release adds numerous improvements including greater extensibility and enhanced security capabilities.

WildFly 8.0

Undertow, the new cutting-edge web server in WildFly 8, is designed for maximum throughput and scalability, including environments with over a million connections. It **supports non-blocking and blocking handlers, traditional and asynchronous servlets, and JSR-356 web socket handlers**. It is highly customizable, with the ability for applications to implement nearly anything from dynamic request routing to custom protocols. It can also function as a very efficient, pure non-blocking reverse proxy, allowing WildFly to delegate to other web servers with minimal impact to running applications. This release adds numerous improvements including greater extensibility and enhanced security capabilities.

WildFly 8.0

Undertow, the new cutting-edge web server in WildFly 8, is designed for maximum throughput and scalability, including environments with over a million connections. It supports non-blocking and blocking handlers, traditional and asynchronous servlets, and JSR-356 web socket handlers. It is **highly customizable**, with the ability for applications to implement nearly anything from dynamic request routing to custom protocols. It can also function as a very efficient, pure non-blocking reverse proxy, allowing WildFly to delegate to other web servers with minimal impact to running applications. This release adds numerous improvements including greater extensibility and enhanced security capabilities.

WildFly 8.0

Undertow, the new cutting-edge web server in WildFly 8, is designed for maximum throughput and scalability, including environments with over a million connections. It supports non-blocking and blocking handlers, traditional and asynchronous servlets, and JSR-356 web socket handlers. It is highly customizable, with the **ability for applications to implement nearly anything from dynamic request routing to custom protocols**. It can also function as a very efficient, pure non-blocking reverse proxy, allowing WildFly to delegate to other web servers with minimal impact to running applications. This release adds numerous improvements including greater extensibility and enhanced security capabilities.

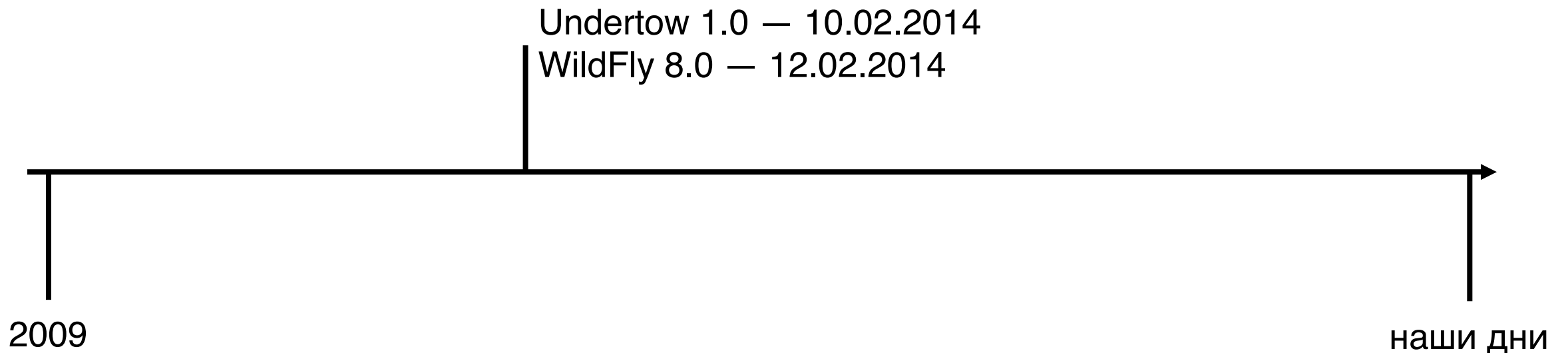


SHUT UP

AND TAKE MY SERVLETS

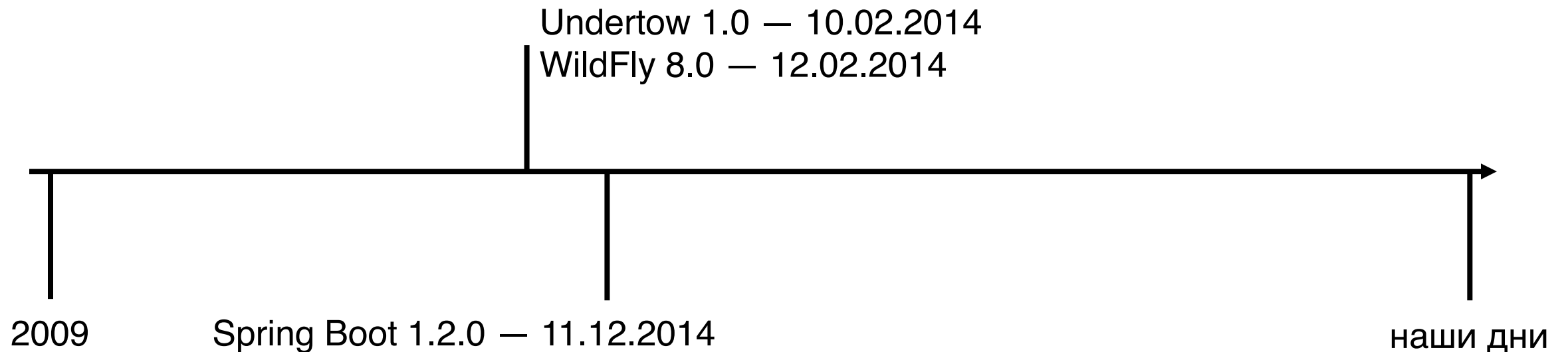
Историческая справка

WildFly 8.0 — 12 февраля 2014



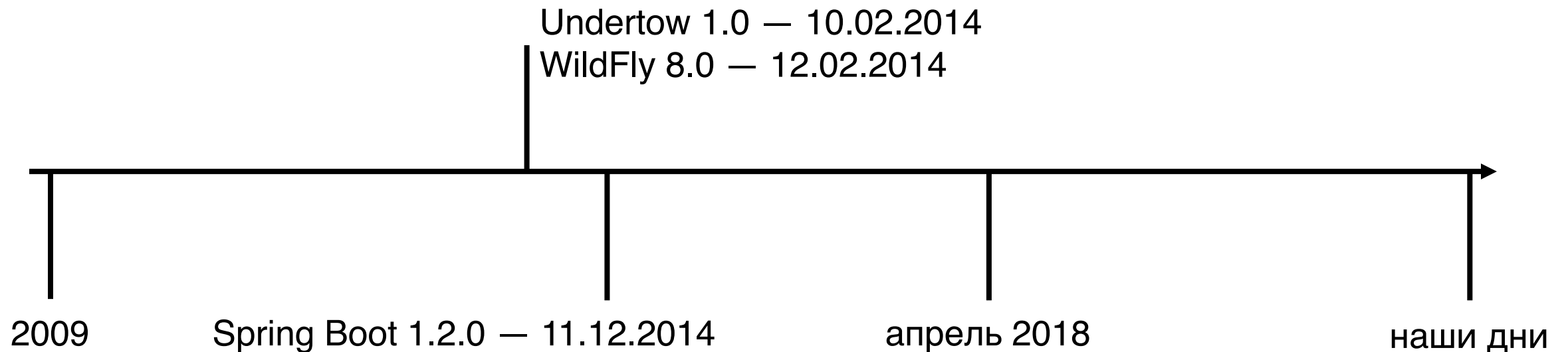
Историческая справка

Spring Boot 1.2.0 — 11 декабря 2014



Историческая справка

Vostok Hercules — апрель 2018



Web Framework Benchmarks

2018-02-14

Round 15

A valentine for performance

Roses are red






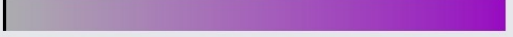
















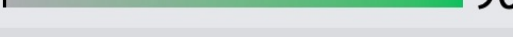

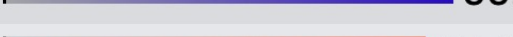


Violets are blue

A fast server






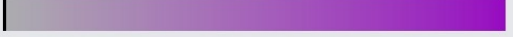
















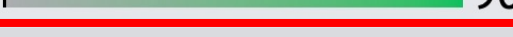

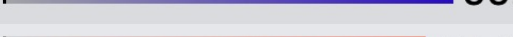


Makes users love you

View [additional commentary about Round 15 at our blog.](#)

Web Framework Benchmarks

Rnk	Framework	Best performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	IA
1	 fasthttp	2,665,278  100.0%	0	Plt	Go	Non	Non	Lin	Rea
2	 rapidoid-http-fast	2,654,663  99.6%	0	Plt	Jav	Rap	Non	Lin	Rea
3	 hyper	2,633,824  98.8%	0	Mcr	rs	rs	hyp	Lin	Rea
4	  actix	2,633,461  98.8%	0	Mcr	rs	act	act	Lin	Rea
5	 rapidoid	2,608,986  97.9%	0	Plt	Jav	Rap	Non	Lin	Rea
6	 undertow	2,544,955  95.5%	0	Plt	Jav	Utw	Non	Lin	Rea
7	 vertx	2,503,741  93.9%	0	Plt	Jav	ver	Non	Lin	Rea
8	 netty	2,447,315  91.8%	1	Plt	Jav	Nty	Non	Lin	Rea
9	 proteus	2,441,688  91.6%	0	Mcr	Jav	Utw	Non	Lin	Rea
10	 light-4j	2,435,150  91.4%	0	Plt	Jav	lig	Non	Lin	Rea
11	 vertx-web	2,411,118  90.5%	0	Mcr	Jav	ver	Non	Lin	Rea
12	 japronto	2,356,952  88.4%	178	Mcr	Py	Non	Non	Lin	Rea
13	 aspcore-mw	2,216,711  83.2%	110	Mcr	C#	.NE	kes	Lin	Rea






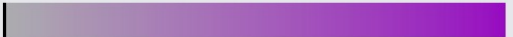










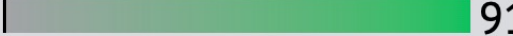



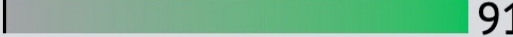



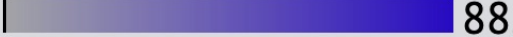


Web Framework Benchmarks

Rnk	Framework	Best performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	IA
1	 fasthttp	2,665,278  100.0%	0	Plt	Go	Non	Non	Lin	Rea
2	 rapidoid-http-fast	2,654,663  99.6%	0	Plt	Jav	Rap	Non	Lin	Rea
3	 hyper	2,633,824  98.8%	0	Mcr	rs	rs	hyp	Lin	Rea
4	  actix	2,633,461  98.8%	0	Mcr	rs	act	act	Lin	Rea
5	 rapidoid	2,608,986  97.9%	0	Plt	Jav	Rap	Non	Lin	Rea
6	 undertow	2,544,955  95.5%	0	Plt	Jav	Utw	Non	Lin	Rea
7	 vertx	2,503,741  93.9%	0	Plt	Jav	ver	Non	Lin	Rea
8	 netty	2,447,315  91.8%	1	Plt	Jav	Nty	Non	Lin	Rea
9	 proteus	2,441,688  91.6%	0	Mcr	Jav	Utw	Non	Lin	Rea
10	 light-4j	2,435,150  91.4%	0	Plt	Jav	lig	Non	Lin	Rea
11	 vertx-web	2,411,118  90.5%	0	Mcr	Jav	ver	Non	Lin	Rea
12	 japronto	2,356,952  88.4%	178	Mcr	Py	Non	Non	Lin	Rea
13	 aspcore-mw	2,216,711  83.2%	110	Mcr	C#	.NE	kes	Lin	Rea




























Web Framework Benchmarks

Rnk	Framework	Best performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	IA
1	fasthttp	2,665,278 100.0%	0	Plt	Go	Non	Non	Lin	Rea
2	rapidoid-http-fast	2,654,663 99.6%	0	Plt	Jav	Rap	Non	Lin	Rea
3	hyper	2,633,824 98.8%	0	Mcr	rs	rs	hyp	Lin	Rea
4	actix	2,633,461 98.8%	0	Mcr	rs	act	act	Lin	Rea
5	rapidoid	2,608,986 97.9%	0	Plt	Jav	Rap	Non	Lin	Rea
6	undertow	2,544,955 95.5%	0	Plt	Jav	Utw	Non	Lin	Rea
7	vertx	2,503,741 93.9%	0	Plt	Jav	ver	Non	Lin	Rea
8	netty	2,447,315 91.8%	1	Plt	Jav	Nty	Non	Lin	Rea
9	proteus	2,441,688 91.6%	0	Mcr	Jav	Utw	Non	Lin	Rea
10	light-4j	2,435,150 91.4%	0	Plt	Jav	lig	Non	Lin	Rea
11	vertx-web	2,411,118 90.5%	0	Mcr	Jav	ver	Non	Lin	Rea
12	japronto	2,356,952 88.4%	178	Mcr	Py	Non	Non	Lin	Rea
13	aspcore-mw	2,216,711 83.2%	110	Mcr	C#	.NE	kes	Lin	Rea

Web Framework Benchmarks

Rnk	Framework	Best performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	IA
1	 fasthttp	2,665,278  100.0%	0	Plt	Go	Non	Non	Lin	Rea
2	 rapidoid-http-fast	2,654,663  99.6%	0	Plt	Jav	Rap	Non	Lin	Rea
3	 hyper	2,633,824  98.8%	0	Mcr	rs	rs	hyp	Lin	Rea
4	  actix	2,633,461  98.8%	0	Mcr	rs	act	act	Lin	Rea
5	 rapidoid	2,608,986  97.9%	0	Plt	Jav	Rap	Non	Lin	Rea
6	 undertow	2,544,955  95.5%	0	Plt	Jav	Utw	Non	Lin	Rea
7	 vertx	2,503,741  93.9%	0	Plt	Jav	ver	Non	Lin	Rea
8	 netty	2,447,315  91.8%	1	Plt	Jav	Nty	Non	Lin	Rea
9	 proteus	2,441,688  91.6%	0	Mcr	Jav	Utw	Non	Lin	Rea
10	 light-4j	2,435,150  91.4%	0	Plt	Jav	lig	Non	Lin	Rea
11	 vertx-web	2,411,118  90.5%	0	Mcr	Jav	ver	Non	Lin	Rea
12	 japronto	2,356,952  88.4%	178	Mcr	Py	Non	Non	Lin	Rea
13	 aspcore-mw	2,216,711  83.2%	110	Mcr	C#	.NE	kes	Lin	Rea

Web Framework Benchmarks






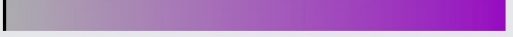










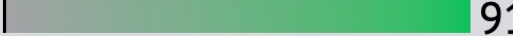



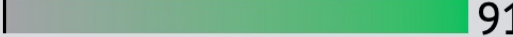



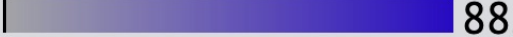


Rnk	Framework	Best performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	IA
1	 fasthttp	2,665,278  100.0%	0	Plt	Go	Non	Non	Lin	Rea
2	 rapidoid-http-fast	2,654,663  99.6%	0	Plt	Jav	Rap	Non	Lin	Rea
3	 hyper	2,633,824  98.8%	0	Mcr	rs	rs	hyp	Lin	Rea
4	  actix	2,633,461  98.8%	0	Mcr	rs	act	act	Lin	Rea
5	 rapidoid	2,608,986  97.9%	0	Plt	Jav	Rap	Non	Lin	Rea
6	 undertow	2,544,955  95.5%	0	Plt	Jav	Utw	Non	Lin	Rea
7	 vertx	2,503,741  93.9%	0	Plt	Jav	ver	Non	Lin	Rea
8	 netty	2,447,315  91.8%	1	Plt	Jav	Nty	Non	Lin	Rea
9	 proteus	2,441,688  91.6%	0	Mcr	Jav	Utw	Non	Lin	Rea
10	 light-4j	2,435,150  91.4%	0	Plt	Jav	lig	Non	Lin	Rea
11	 vertx-web	2,411,118  90.5%	0	Mcr	Jav	ver	Non	Lin	Rea
12	 japronto	2,356,952  88.4%	178	Mcr	Py	Non	Non	Lin	Rea
13	 aspcore-mw	2,216,711  83.2%	110	Mcr	C#	.NE	kes	Lin	Rea

Rapidoid

«Высокопроизводительное Java-приложение
в сердце стриминговой архитектуры»

Алексей Кирпичников, Контур
java.ural.Meetup@1






















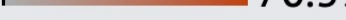

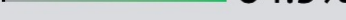

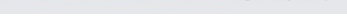
Web Framework Benchmarks

Rnk	Framework	Best performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	IA
1	 fasthttp	2,665,278  100.0%	0	Plt	Go	Non	Non	Lin	Rea
2	 rapidoid-http-fast	2,654,663  99.6%	0	Plt	Jav	Rap	Non	Lin	Rea
3	 hyper	2,633,824  98.8%	0	Mcr	rs	rs	hyp	Lin	Rea
4	  actix	2,633,461  98.8%	0	Mcr	rs	act	act	Lin	Rea
5	 rapidoid	2,608,986  97.9%	0	Plt	Jav	Rap	Non	Lin	Rea
6	 undertow	2,544,955  95.5%	0	Plt	Jav	Utw	Non	Lin	Rea
7	 vertx	2,503,741  93.9%	0	Plt	Jav	ver	Non	Lin	Rea
8	 netty	2,447,315  91.8%	1	Plt	Jav	Nty	Non	Lin	Rea
9	 proteus	2,441,688  91.6%	0	Mcr	Jav	Utw	Non	Lin	Rea
10	 light-4j	2,435,150  91.4%	0	Plt	Jav	lig	Non	Lin	Rea
11	 vertx-web	2,411,118  90.5%	0	Mcr	Jav	ver	Non	Lin	Rea
12	 japronto	2,356,952  88.4%	178	Mcr	Py	Non	Non	Lin	Rea
13	 aspcore-mw	2,216,711  83.2%	110	Mcr	C#	.NE	kes	Lin	Rea


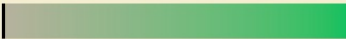

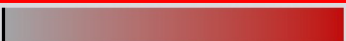






















Web Framework Benchmarks

Rnk	Framework	Best performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	IA
1	fasthttp	2,665,278 100.0%	0	Plt	Go	Non	Non	Lin	Rea
2	rapidoid-http-fast	2,654,663 99.6%	0	Plt	Jav	Rap	Non	Lin	Rea
3	hyper	2,633,824 98.8%	0	Mcr	rs	rs	hyp	Lin	Rea
4	actix	2,633,461 98.8%	0	Mcr	rs	act	act	Lin	Rea
5	rapidoid	2,608,986 97.9%	0	Plt	Jav	Rap	Non	Lin	Rea
6	undertow	2,544,955 95.5%	0	Plt	Jav	Utw	Non	Lin	Rea
7	vertx	2,503,741 93.9%	0	Plt	Jav	ver	Non	Lin	Rea
8	netty	2,447,315 91.8%	1	Plt	Jav	Nty	Non	Lin	Rea
9	proteus	2,441,688 91.6%	0	Mcr	Jav	Utw	Non	Lin	Rea
10	light-4j	2,435,150 91.4%	0	Plt	Jav	lig	Non	Lin	Rea
11	vertx-web	2,411,118 90.5%	0	Mcr	Jav	ver	Non	Lin	Rea
12	japronto	2,356,952 88.4%	178	Mcr	Py	Non	Non	Lin	Rea
13	aspcore-mw	2,216,711 83.2%	110	Mcr	C#	.NE	kes	Lin	Rea














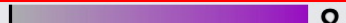











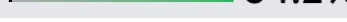
Web Framework Benchmarks

Rnk	Framework	Performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	DB	Dos	Orm	IA
1	 undertow-postgresql	9,514  100.0%	0	Plt	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
2	 h2o	9,313  97.9%	0	Plt	C	Non	h2o	Lin	Pg	Lin	Raw	Rea
3	 jooby	9,196  96.7%	0	Ful	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
4	 vertx-web-postgres	9,012  94.7%	0	Mcr	Jav	ver	Non	Lin	Pg	Lin	Raw	Rea
5	 vertx-postgres	8,697  91.4%	0	Plt	Jav	ver	Non	Lin	Pg	Lin	Raw	Rea
6	 jawn	8,332  87.6%	0	Ful	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
7	 actix	8,157  85.7%	0	Mcr	rs	act	act	Lin	Pg	Lin	Raw	Rea
8	 cutelyst-pf-pg	7,563  79.5%	0	Ful	C++	cut	Non	Lin	Pg	Lin	Raw	Rea
9	 cpoll_cppsp-postgres-raw	7,177  75.4%	0	Plt	C++	Non	Non	Lin	Pg	Lin	Raw	Rea
10	 urweb	6,780  71.3%	0	Ful	Ur	Ur/	Non	Lin	Pg	Lin	Mcr	Rea
11	 treefrog-mongodb	6,688  70.3%	0	Ful	C++	Non	Non	Lin	Mo	Lin	Mcr	Rea
12	 revenj-jvm	6,133  64.5%	0	Ful	Jav	Svt	Res	Lin	Pg	Lin	Ful	Rea
13	 gemini-postgres	6,109  64.2%	0	Ful	Jav	Svt	Res	Lin	Pg	Lin	Mcr	Rea

Web Framework Benchmarks

Rnk	Framework	Performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	DB	Dos	Orm	IA
1	 undertow-postgresql	9,514  100.0%	0	Plt	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
2	 h2o	9,313  97.9%	0	Plt	C	Non	h2o	Lin	Pg	Lin	Raw	Rea
3	 jooby	9,196  96.7%	0	Ful	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
4	 vertx-web-postgres	9,012  94.7%	0	Mcr	Jav	ver	Non	Lin	Pg	Lin	Raw	Rea
5	 vertx-postgres	8,697  91.4%	0	Plt	Jav	ver	Non	Lin	Pg	Lin	Raw	Rea
6	 jawn	8,332  87.6%	0	Ful	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
7	 actix	8,157  85.7%	0	Mcr	rs	act	act	Lin	Pg	Lin	Raw	Rea
8	 cutelyst-pf-pg	7,563  79.5%	0	Ful	C++	cut	Non	Lin	Pg	Lin	Raw	Rea
9	 cpoll_cppsp-postgres-raw	7,177  75.4%	0	Plt	C++	Non	Non	Lin	Pg	Lin	Raw	Rea
10	 urweb	6,780  71.3%	0	Ful	Ur	Ur/	Non	Lin	Pg	Lin	Mcr	Rea
11	 treefrog-mongodb	6,688  70.3%	0	Ful	C++	Non	Non	Lin	Mo	Lin	Mcr	Rea
12	 revenj-jvm	6,133  64.5%	0	Ful	Jav	Svt	Res	Lin	Pg	Lin	Ful	Rea
13	 gemini-postgres	6,109  64.2%	0	Ful	Jav	Svt	Res	Lin	Pg	Lin	Mcr	Rea

Web Framework Benchmarks

Rnk	Framework	Performance (higher is better)	Errors	Cls	Lng	Plt	FE	Aos	DB	Dos	Orm	IA
1	 undertow-postgresql	9,514  100.0%	0	Plt	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
2	 h2o	9,313  97.9%	0	Plt	C	Non	h2o	Lin	Pg	Lin	Raw	Rea
3	 jooby	9,196  96.7%	0	Ful	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
4	 vertx-web-postgres	9,012  94.7%	0	Mcr	Jav	ver	Non	Lin	Pg	Lin	Raw	Rea
5	 vertx-postgres	8,697  91.4%	0	Plt	Jav	ver	Non	Lin	Pg	Lin	Raw	Rea
6	 jawn	8,332  87.6%	0	Ful	Jav	Utw	Non	Lin	Pg	Lin	Raw	Rea
7	 actix	8,157  85.7%	0	Mcr	rs	act	act	Lin	Pg	Lin	Raw	Rea
8	 cutelyst-pf-pg	7,563  79.5%	0	Ful	C++	cut	Non	Lin	Pg	Lin	Raw	Rea
9	 cpoll_cppsp-postgres-raw	7,177  75.4%	0	Plt	C++	Non	Non	Lin	Pg	Lin	Raw	Rea
10	 urweb	6,780  71.3%	0	Ful	Ur	Ur/	Non	Lin	Pg	Lin	Mcr	Rea
11	 treefrog-mongodb	6,688  70.3%	0	Ful	C++	Non	Non	Lin	Mo	Lin	Mcr	Rea
12	 revenj-jvm	6,133  64.5%	0	Ful	Jav	Svt	Res	Lin	Pg	Lin	Ful	Rea
13	 gemini-postgres	6,109  64.2%	0	Ful	Jav	Svt	Res	Lin	Pg	Lin	Mcr	Rea

Независимое тестирование

1. Undertow
2. Jetty
3. Tomcat

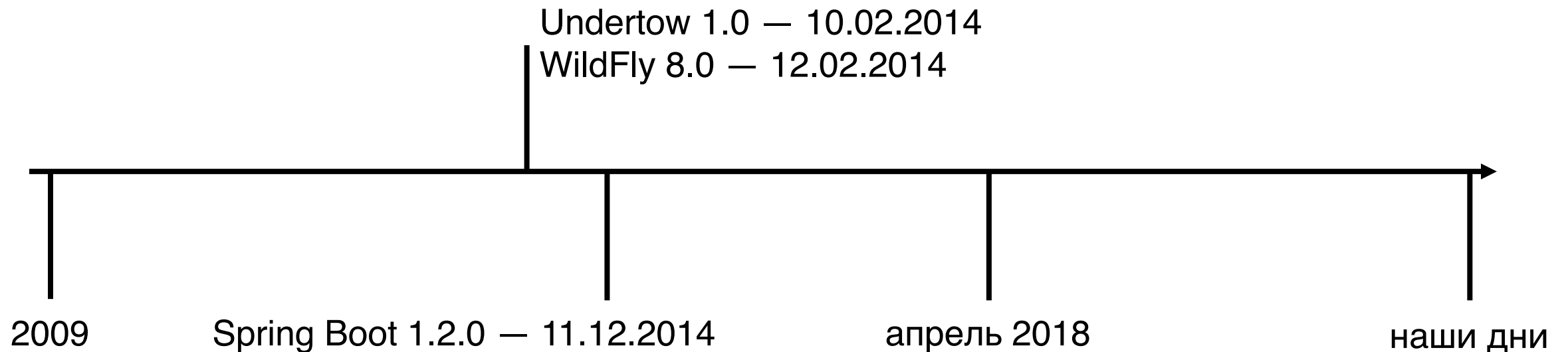
Независимое тестирование

1. Undertow
2. Jetty
3. Tomcat



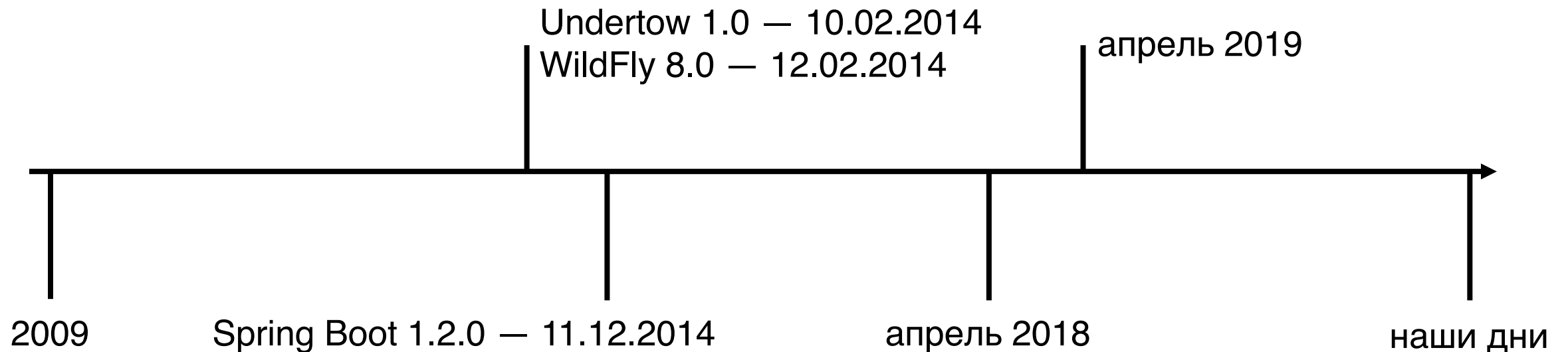
Историческая справка

Vostok Hercules — апрель 2018



Историческая справка

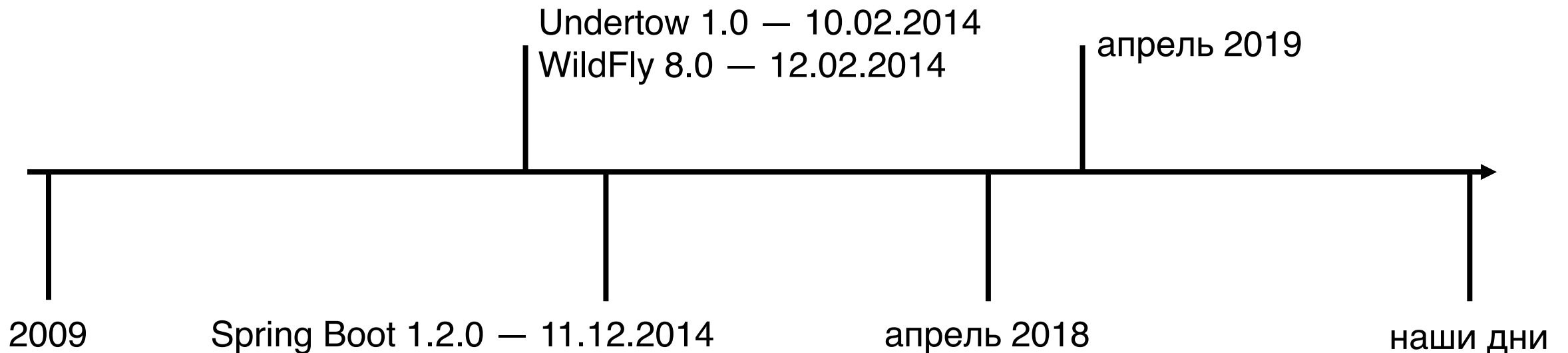
Анонс Undertow 3.0 — апрель 2019



Историческая справка

Анонс Undertow 3.0 — апрель 2019

* XNIO → Netty

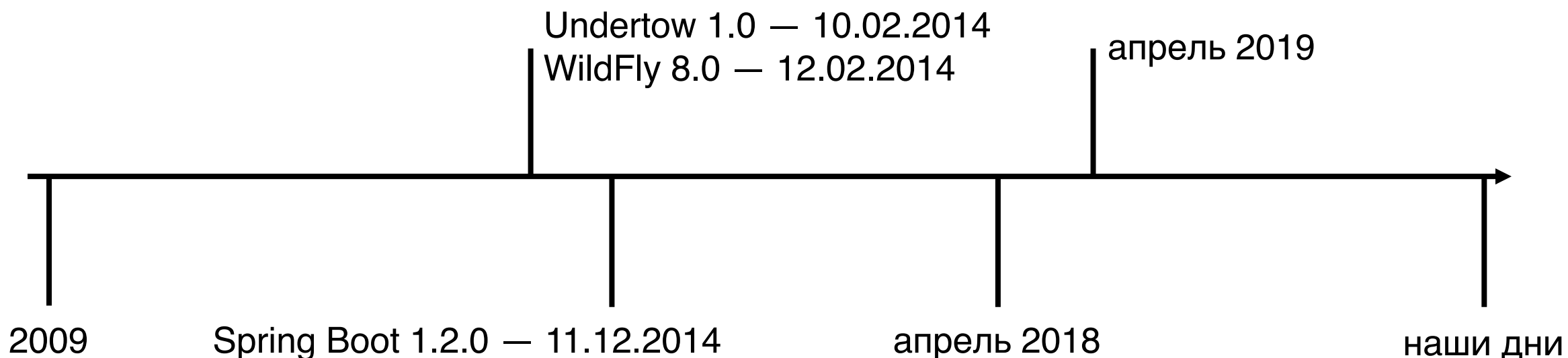


Историческая справка

Анонс Undertow 3.0 — апрель 2019

* XNIO → Netty

* Смена лидера проекта

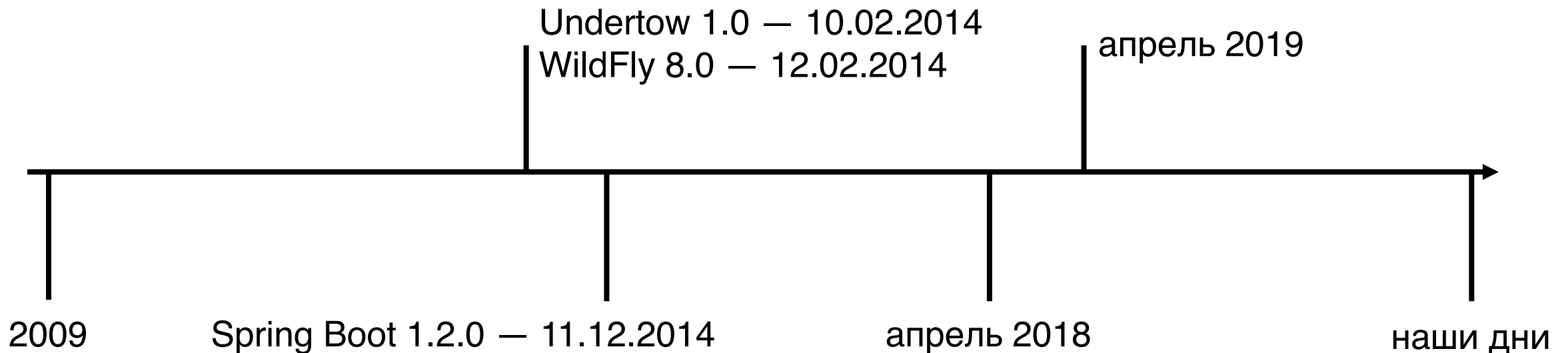


Историческая справка

Анонс Undertow 3.0 — апрель 2019

* XNIO → Netty

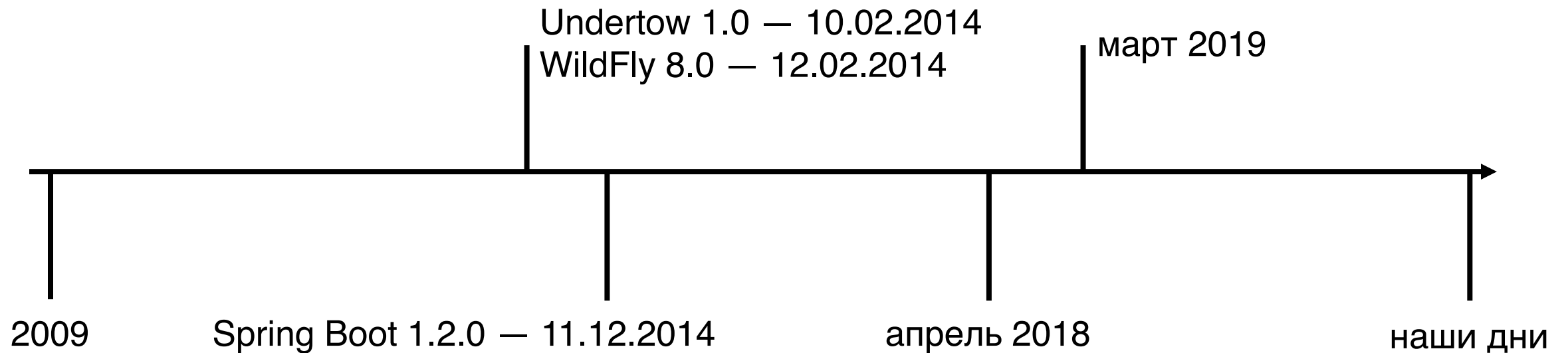
* Смена лидера проекта



Stuart Douglas -> Flavia Rainone

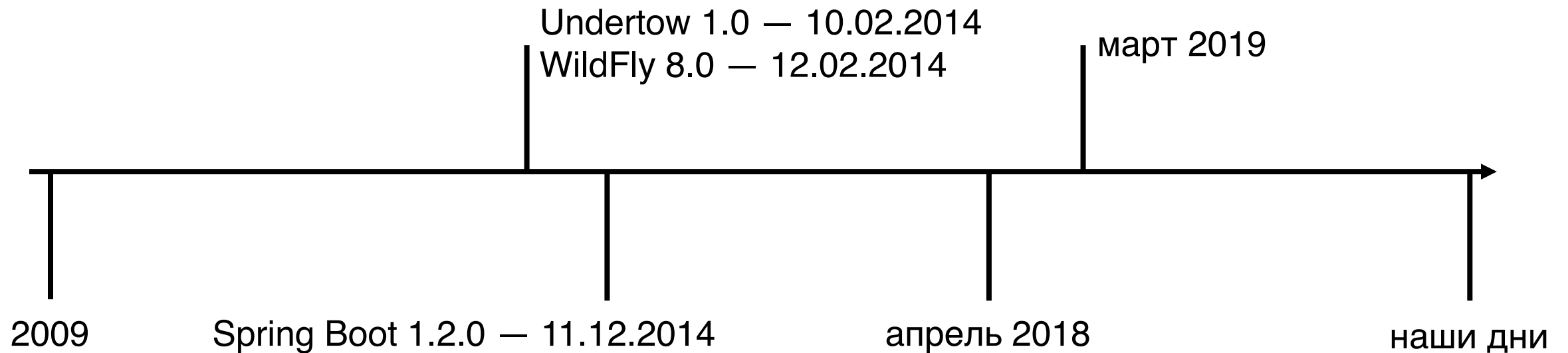
Историческая справка

???



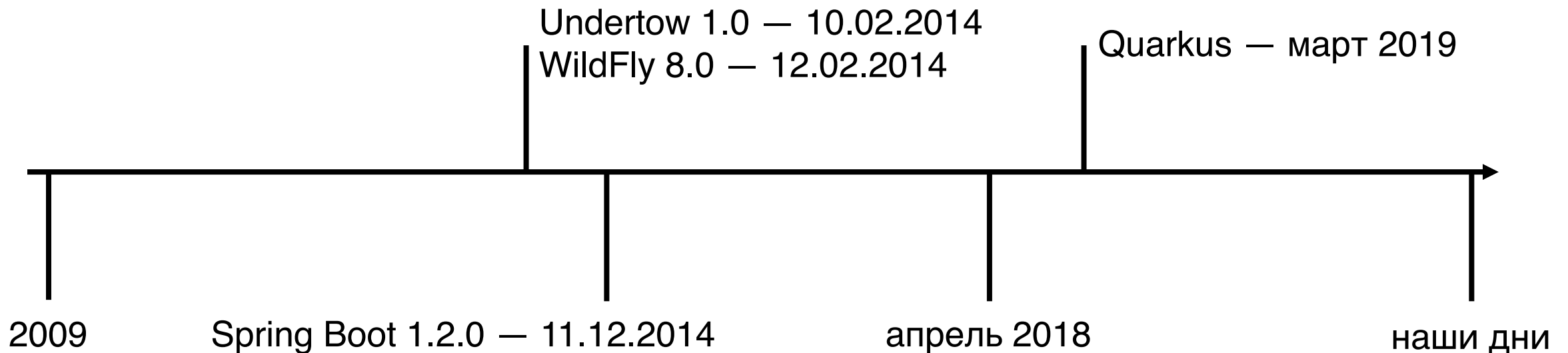
Историческая справка

Supersonic Subatomic Java



Историческая справка

Supersonic Subatomic Java — Quarkus



3.x

15 Branches 317 Tags

Go to file

Code

This branch is 107 commits ahead of, 1269 commits behind master .



fl4via Merge pull request #767 from fl4via/UNDERTOW-1546_3.x

9f45aab · 5 years ago 4,511 Commits

benchmarks	Add an undertow-benchmarks module with basic JMH ben...	5 years ago
core	Change buffer allocation so we control the size	5 years ago
coverage-report	More changes	6 years ago
dist	More changes	6 years ago
examples	Minor cleanup	5 years ago
karaf	IO improvements	6 years ago
servlet	Change buffer allocation so we control the size	5 years ago
websockets-jsr	Don't allow the connection to be closed directly	5 years ago

3.x

15 Branches 317 Tags

Go to file

Code

This branch is 107 commits ahead of, 1269 commits behind master .

fl4via Merge pull request #767 from fl4via/UNDERTOW-1546_3.x

9f45aab · 5 years ago 4,511 Commits

benchmarks	Add an undertow-benchmarks module with basic JMH ben...	5 years ago
core	Change buffer allocation so we control the size	5 years ago
coverage-report	More changes	6 years ago
dist	More changes	6 years ago
examples	Minor cleanup	5 years ago
karaf	IO improvements	6 years ago
servlet	Change buffer allocation so we control the size	5 years ago
websockets-jsr	Don't allow the connection to be closed directly	5 years ago

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Архитектура Undertow

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

IO-intensive приложения

IO-intensive приложения

Что это такое?

IO-intensive приложения

Что это такое?

- Много входящего трафика
- Много исходящего трафика

IO-intensive приложения

Что это такое?

- Много входящего трафика
- Много исходящего трафика

Количество HTTP-запросов

IO-intensive приложения

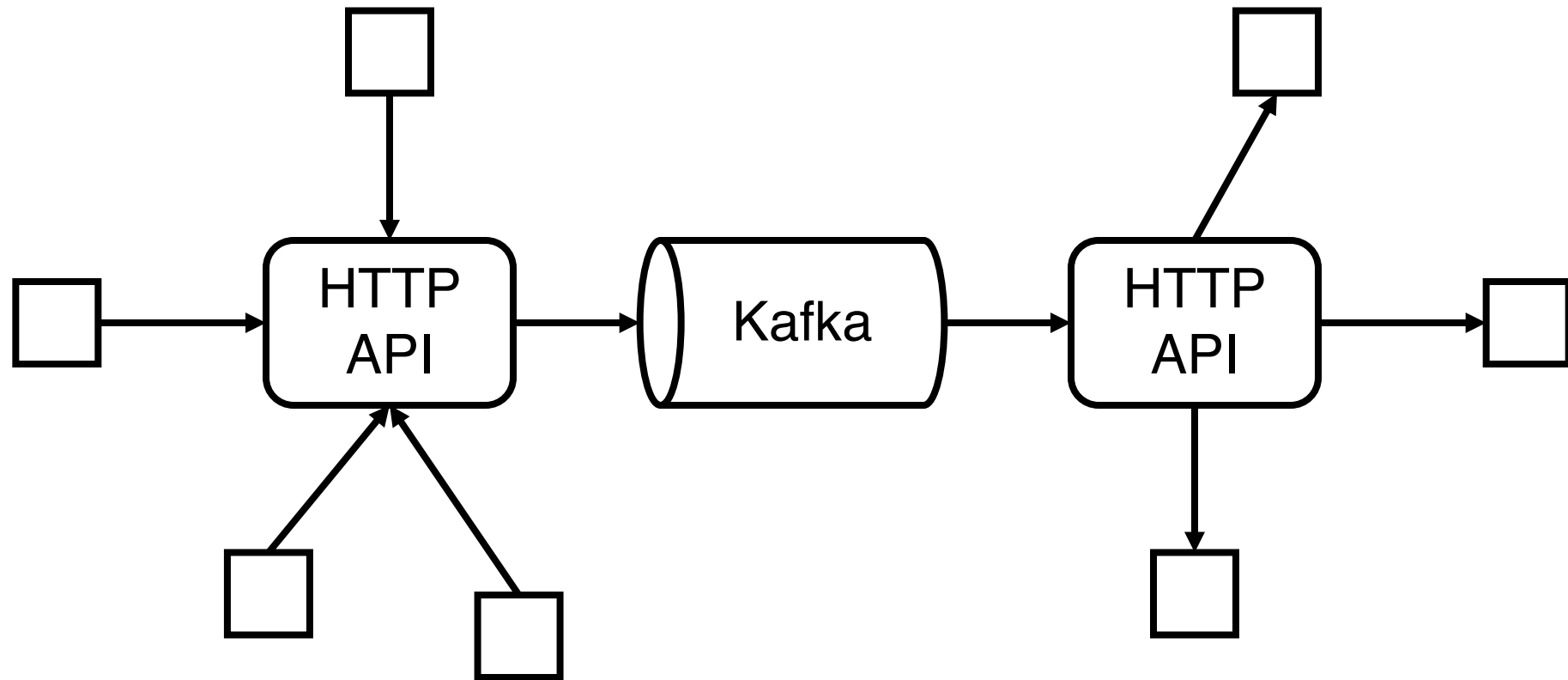
Что это такое?

- Много входящего трафика
- Много исходящего трафика

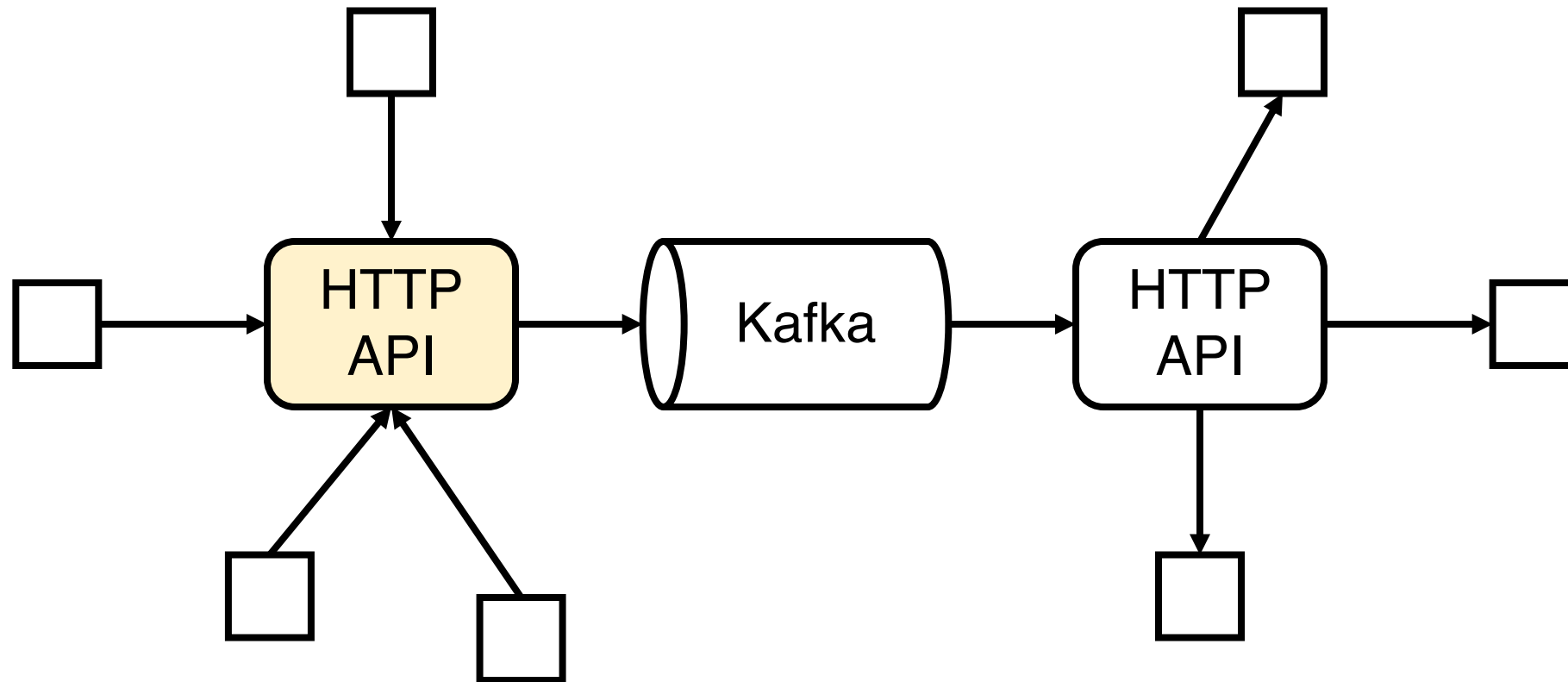
Количество HTTP-запросов

x размер запросов

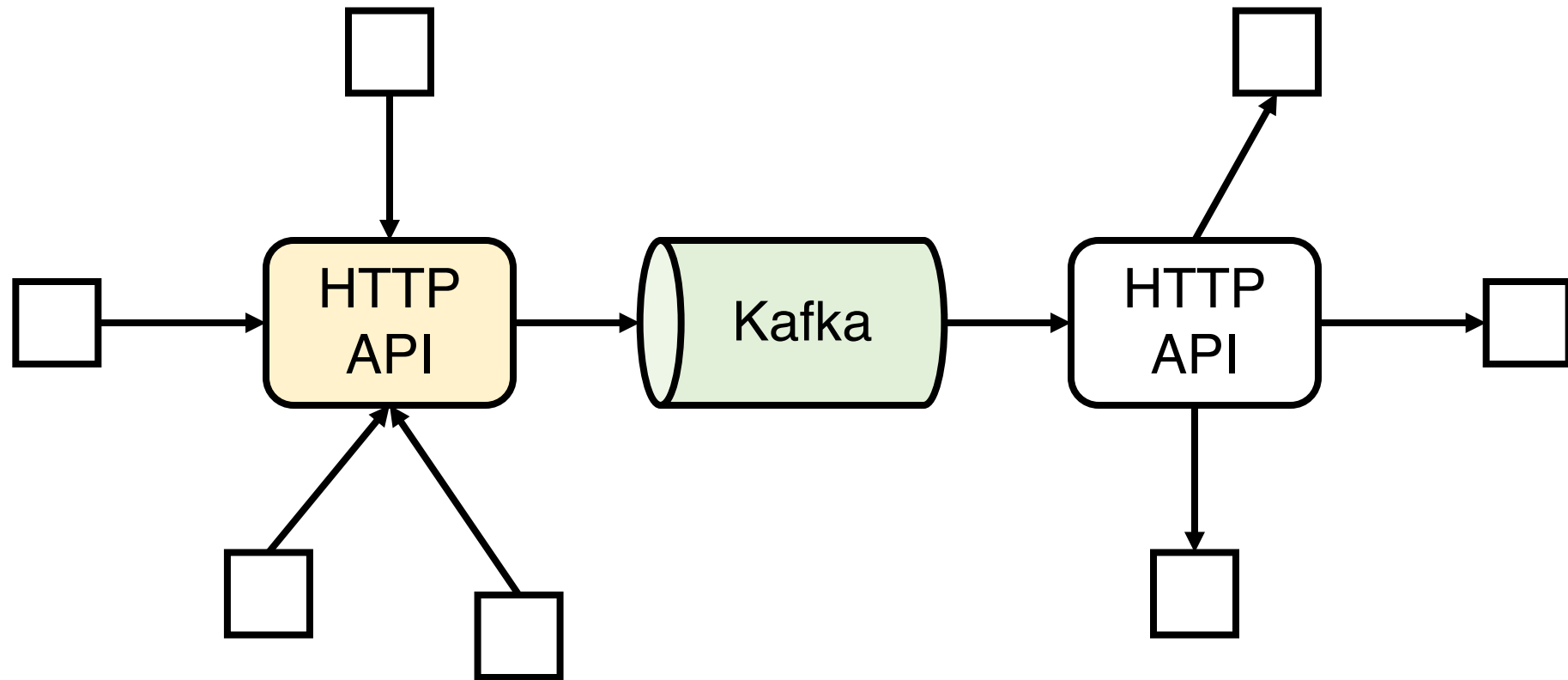
IO-intensive приложения



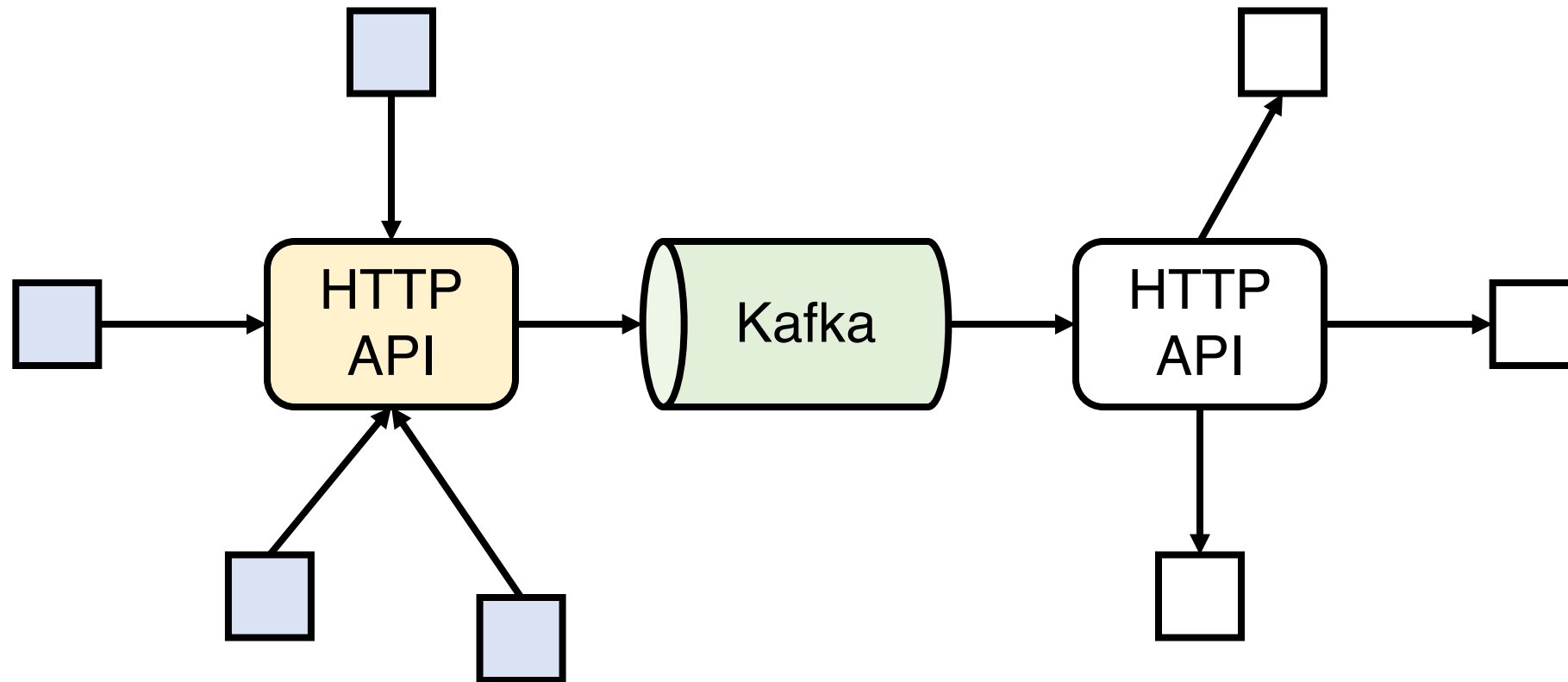
IO-intensive приложения



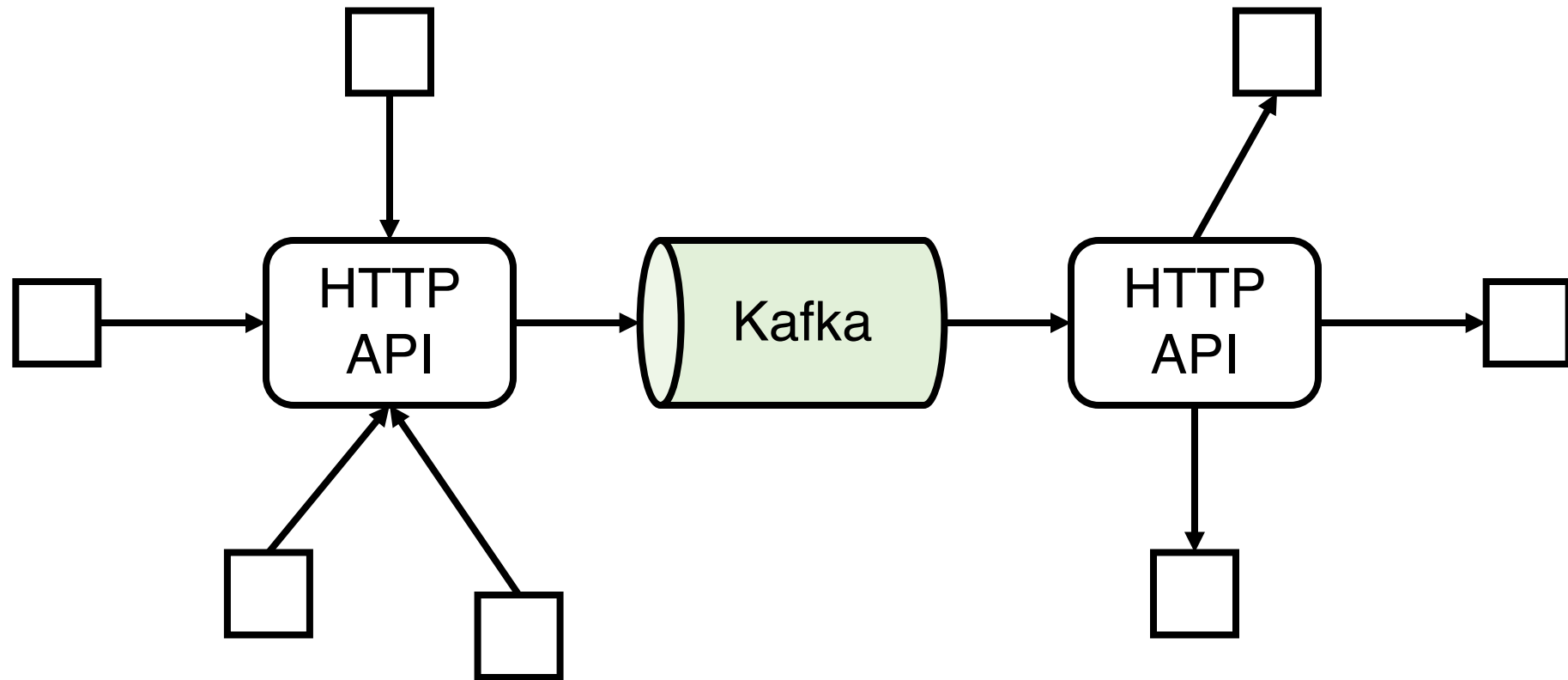
IO-intensive приложения



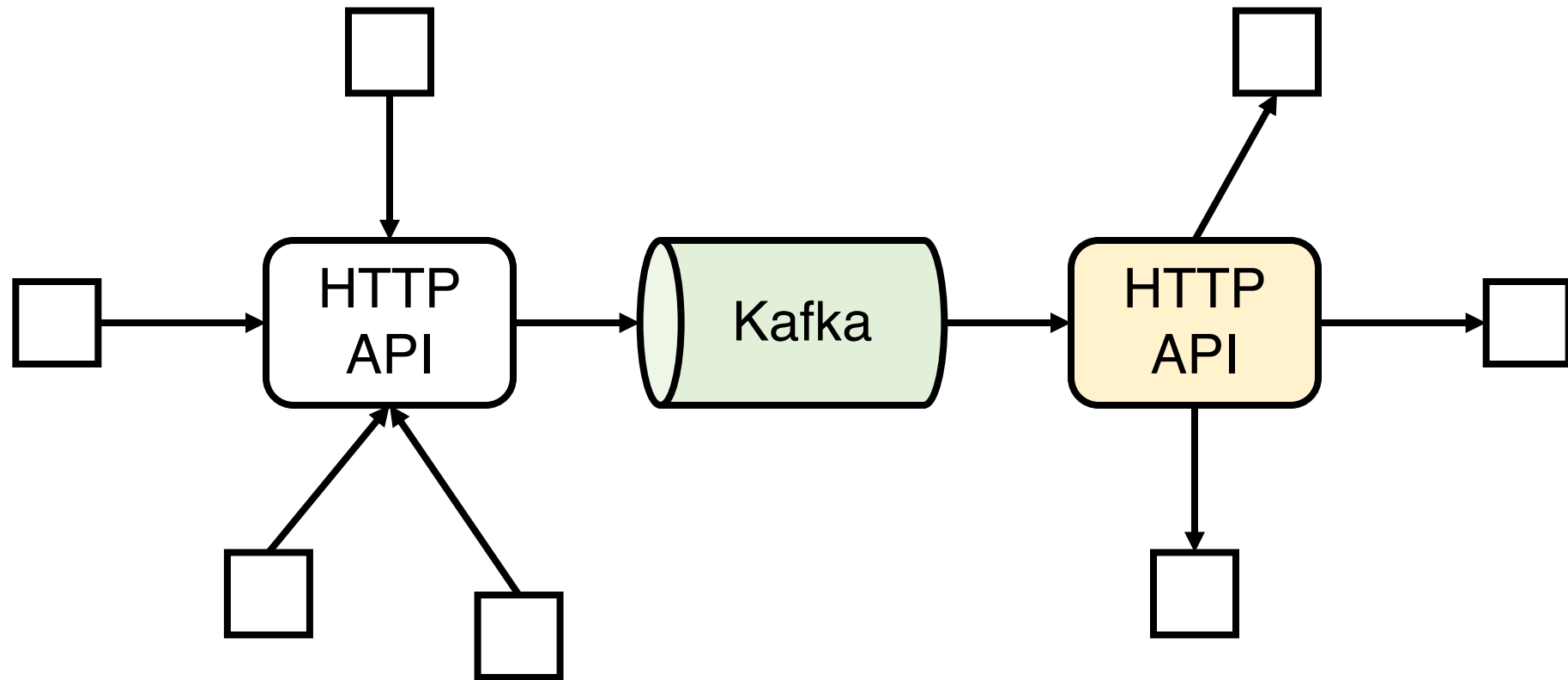
IO-intensive приложения



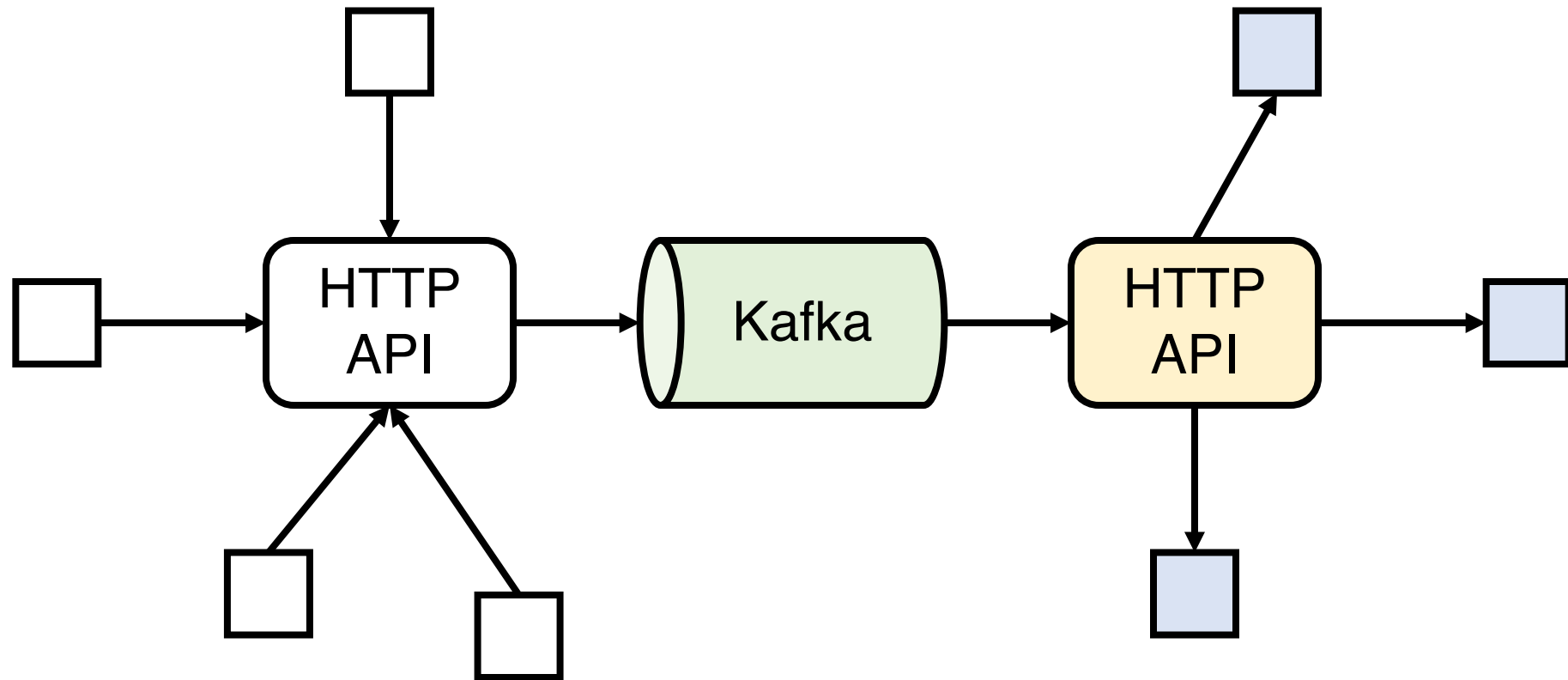
IO-intensive приложения



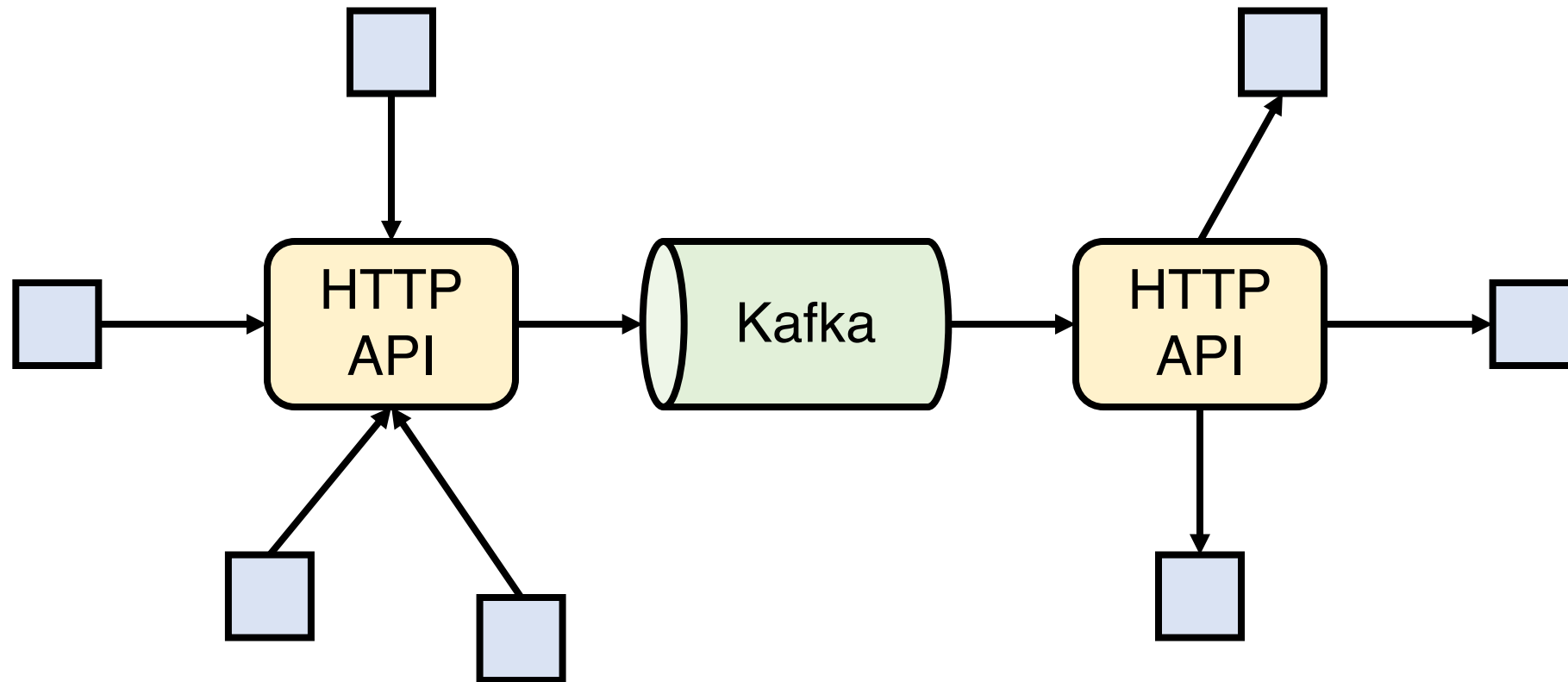
IO-intensive приложения



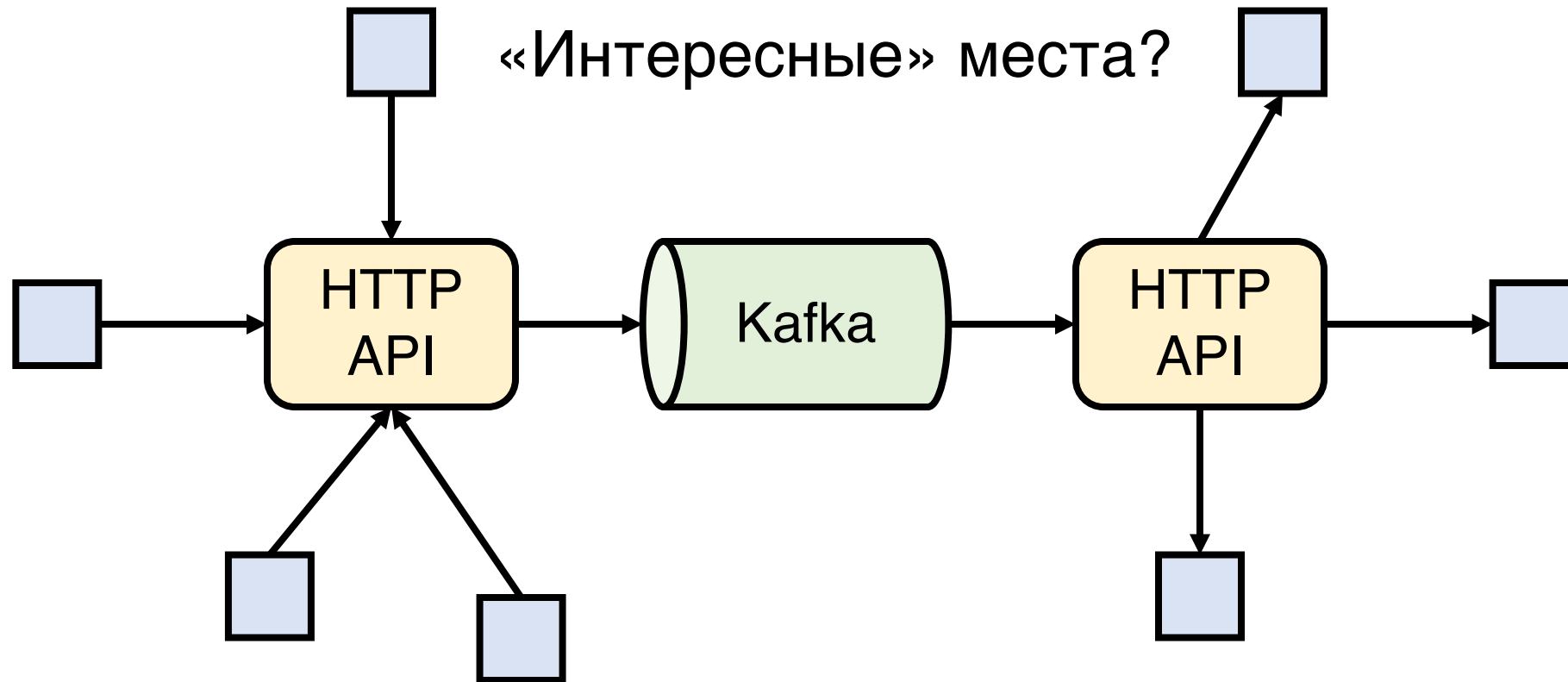
IO-intensive приложения



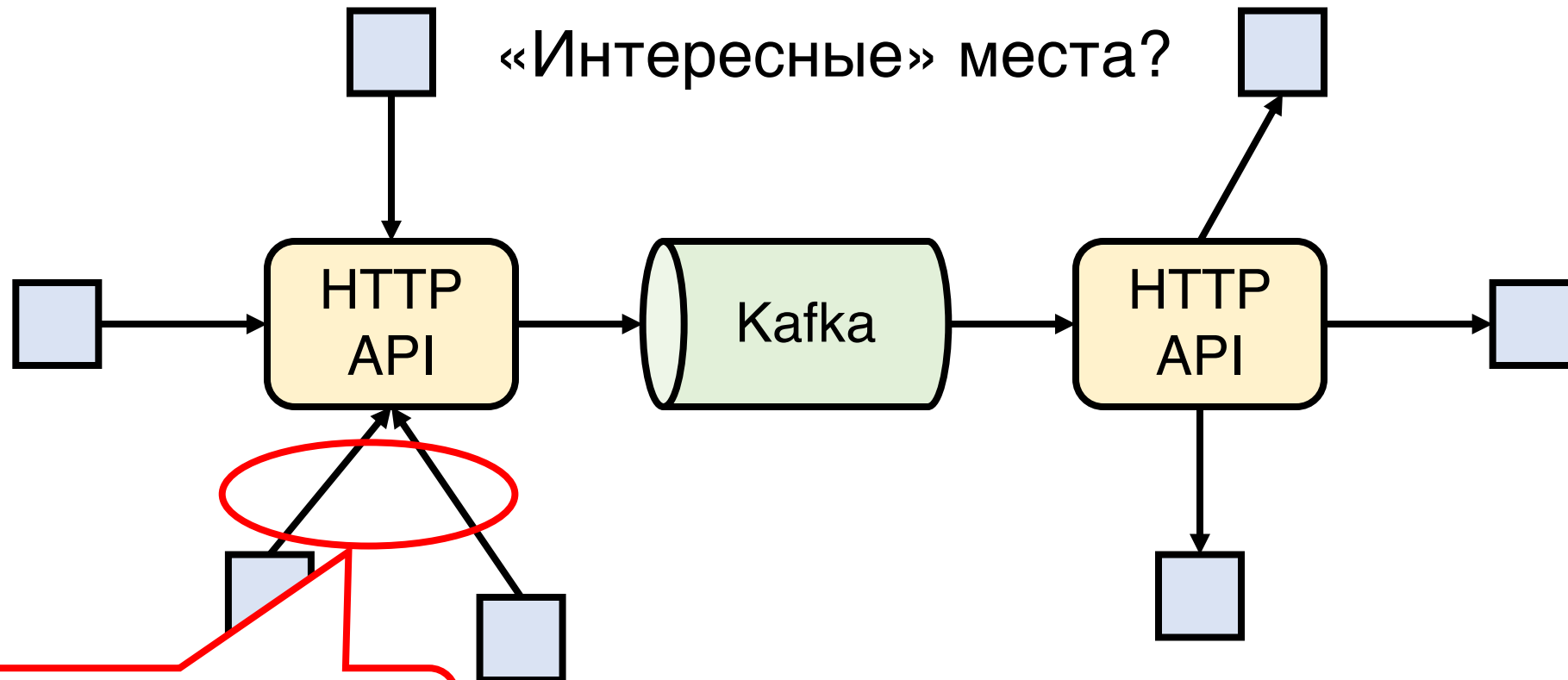
IO-intensive приложения



IO-intensive приложения

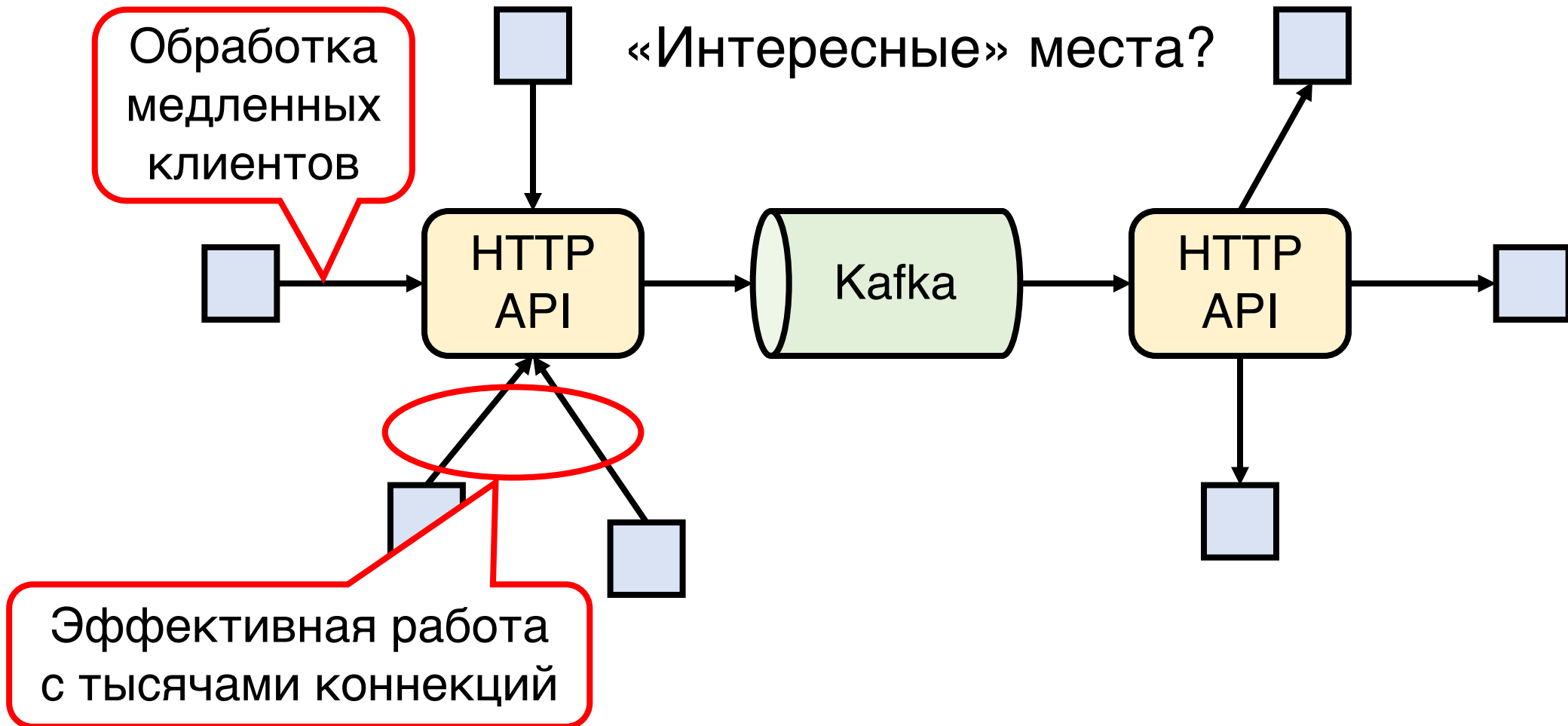


IO-intensive приложения

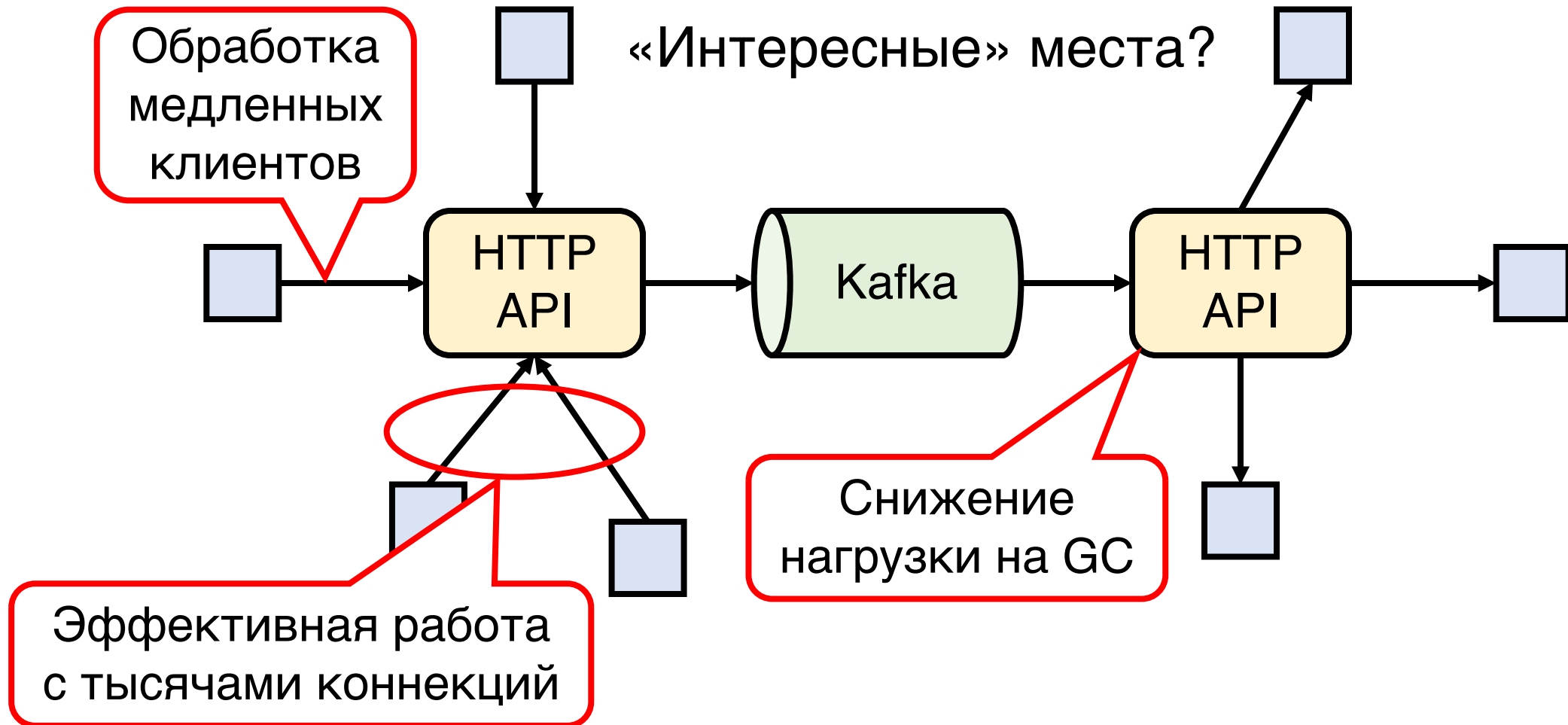


Эффективная работа
с тысячами соединений

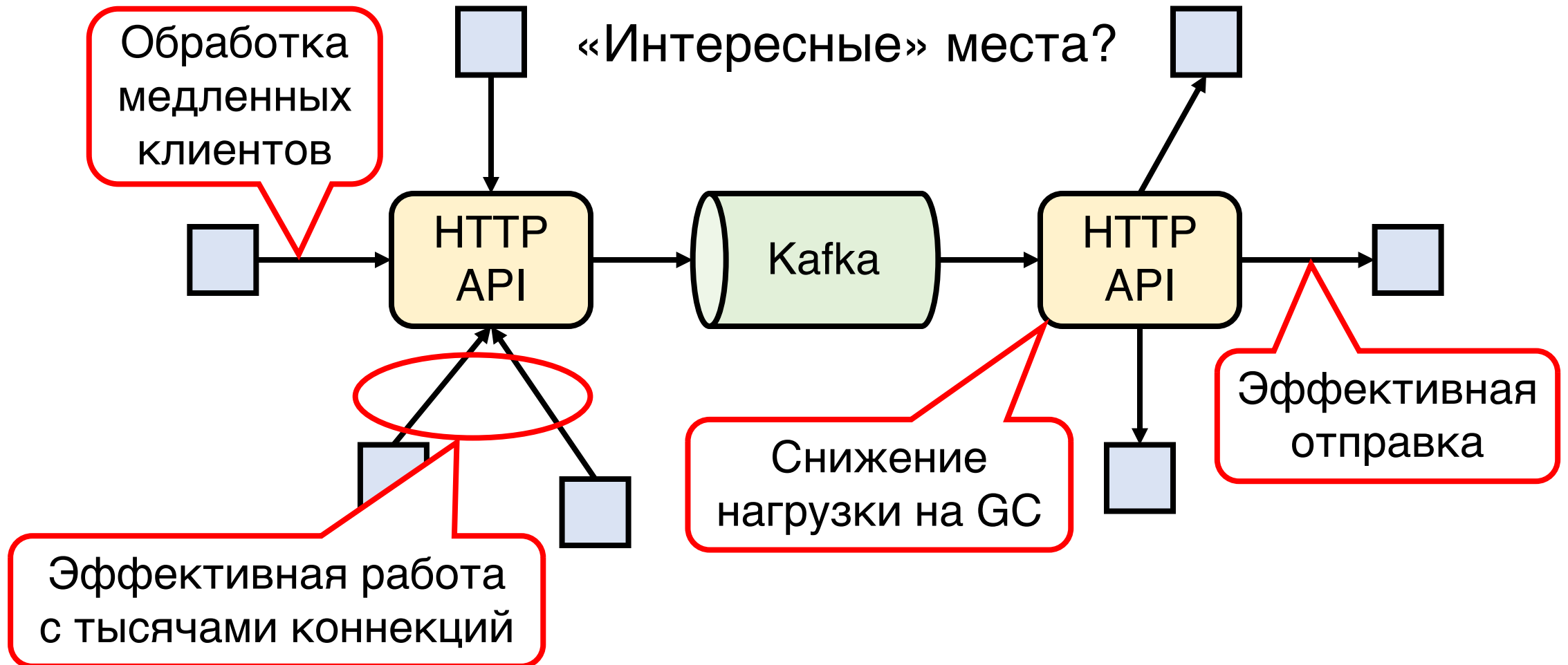
IO-intensive приложения



IO-intensive приложения



IO-intensive приложения



Undertow + XNIO

undertow-servlet
Servlet 3.1+

undertow-websockets-jsr
JSR-356 (Java API for websockets)

undertow-core

XNIO

Java NIO

EPoll (Linux)

KQueue (Mac, BSD)

Undertow + XNIO

```
Undertow server = Undertow.builder()
    .addHttpListener(8080, "localhost")
    .setHandler(Handlers.routing()
        .get("/hello/{any}", new MyHelloAnyHandler())
        .post("/echo", new MyEchoHandler())
    ).build();

server.start();
```

Undertow + XNIO

```
Undertow server = Undertow.builder()
    .addHttpListener(8080, "localhost")
    .setHandler(Handlers.routing()
        .get("/hello/{any}", new MyHelloAnyHandler())
        .post("/echo", new MyEchoHandler())
    ).build();

server.start();
```

Undertow + XNIO

```
Undertow server = Undertow.builder()
    .addHttpListener(8080, "localhost")
    .setHandler(Handlers.routing()
        .get("/hello/{any}", new MyHelloAnyHandler())
        .post("/echo", new MyEchoHandler())
    ).build();
server.start();
```

Undertow + XNIO

```
Undertow server = Undertow.builder()
    .addHttpListener(8080, "localhost")
    .setHandler(Handlers.routing()
        2 .get("/hello/{any}", new MyHelloAnyHandler())
        .post("/echo", new MyEchoHandler())
    ).build();
server.start();
1
```

Undertow + XNIO

```
Undertow server = Undertow.builder()
    .addHttpListener(8080, "localhost")
    .setHandler(Handlers.routing()
        2 .get("/hello/{any}", new MyHelloAnyHandler())
        .post("/echo", new MyEchoHandler()) 3
    ).build();
1 server.start();
```


Undertow + XNIO

Undertow + XNIO

java.nio.channels.spi.
SelectorProvider

Undertow + XNIO

sun.nio.ch.EPollSelectorProvider

java.nio.channels.spi.
SelectorProvider

Undertow + XNIO

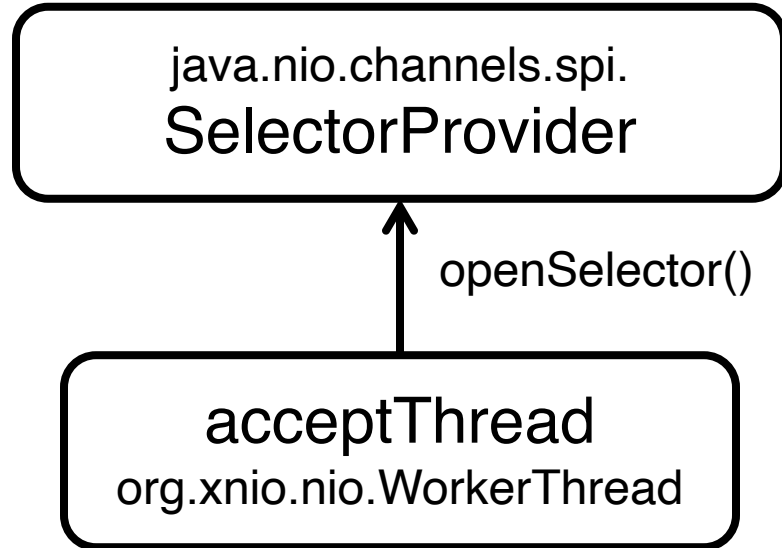
sun.nio.ch.EPollSelectorProvider

java.nio.channels.spi.
SelectorProvider

acceptThread
org.xnio.nio.WorkerThread

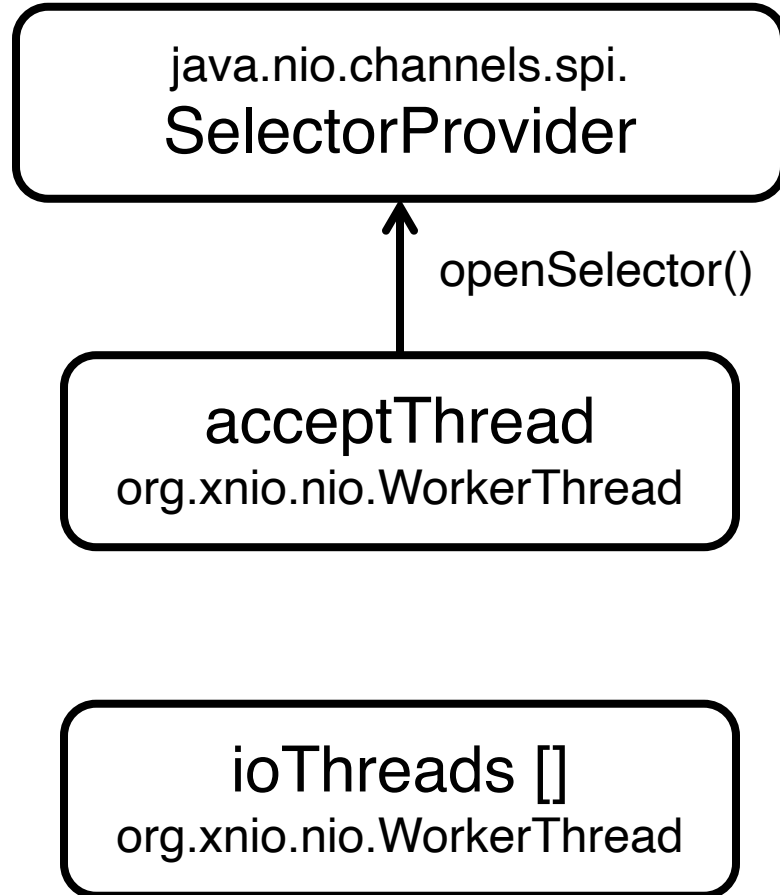
Undertow + XNIO

sun.nio.ch.EPollSelectorProvider



Undertow + XNIO

sun.nio.ch.EPollSelectorProvider



Undertow + XNIO

`sun.nio.ch.EPollSelectorProvider`

`java.nio.channels.spi.
SelectorProvider`

`openSelector()`

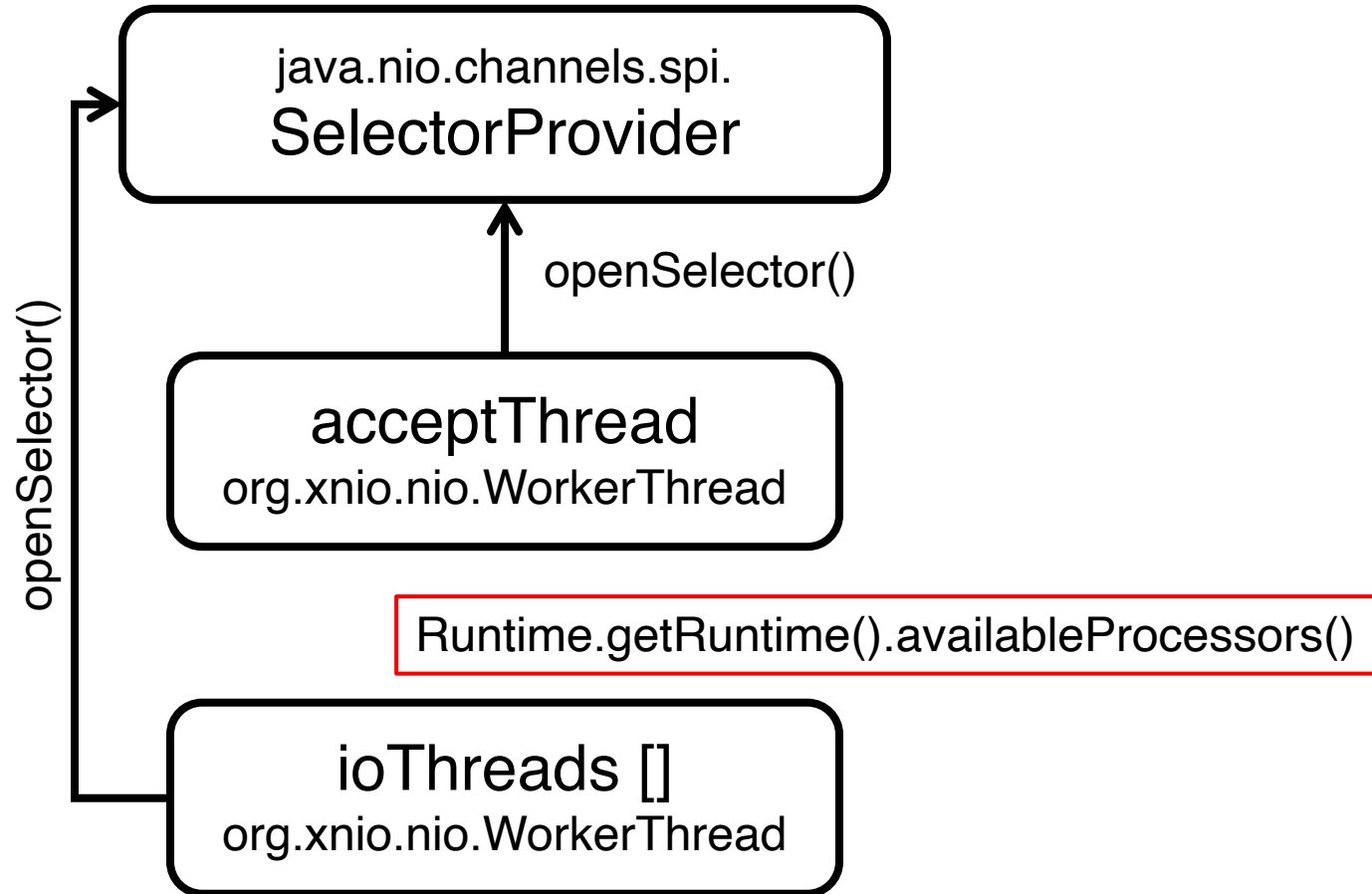
`acceptThread
org.xnio.nio.WorkerThread`

`Runtime.getRuntime().availableProcessors()`

`ioThreads []
org.xnio.nio.WorkerThread`

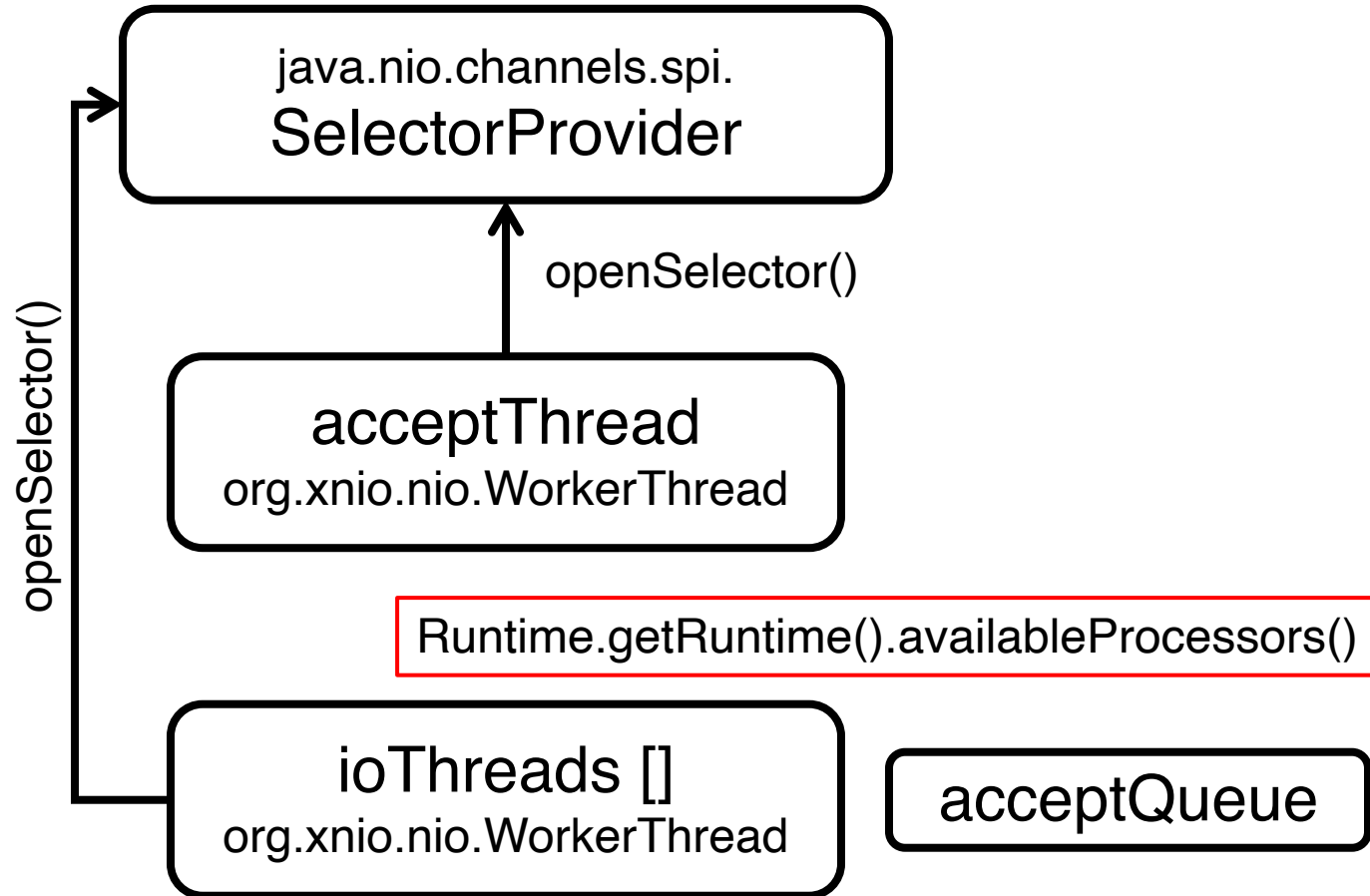
Undertow + XNIO

`sun.nio.ch.EPollSelectorProvider`



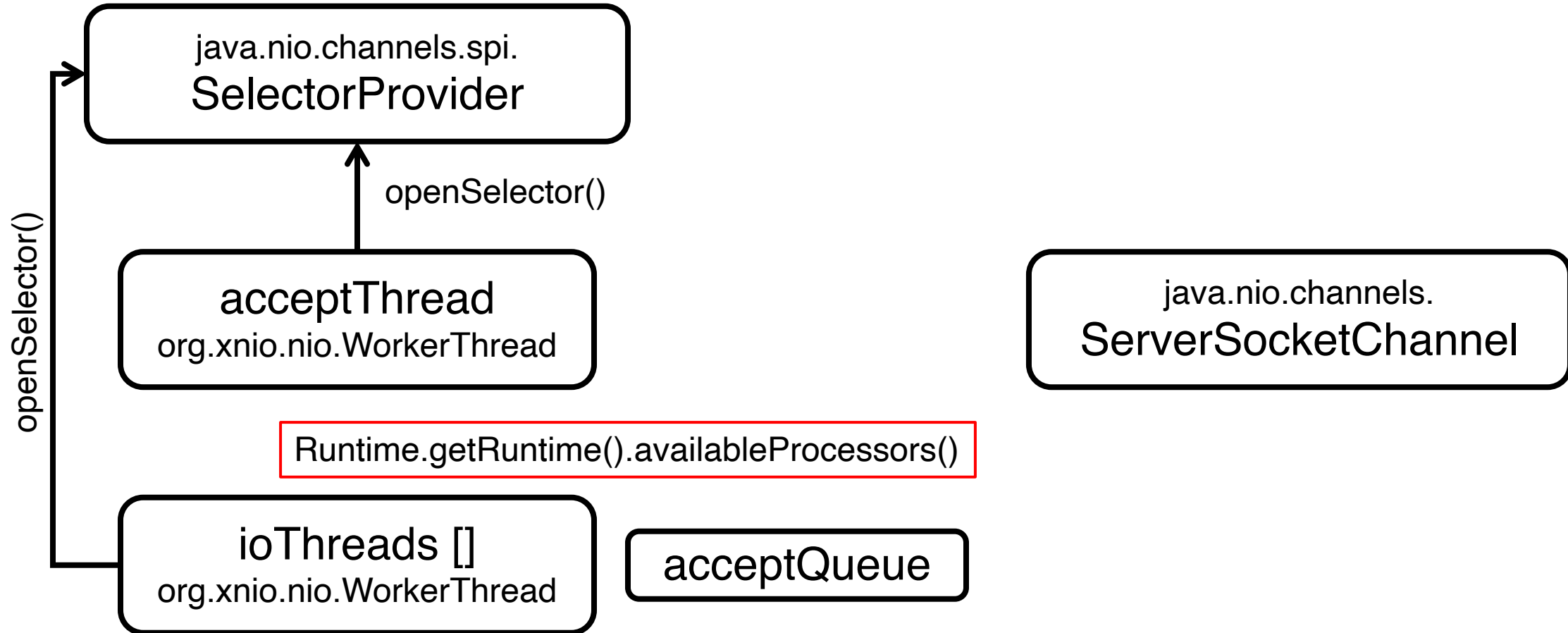
Undertow + XNIO

`sun.nio.ch.EPollSelectorProvider`



Undertow + XNIO

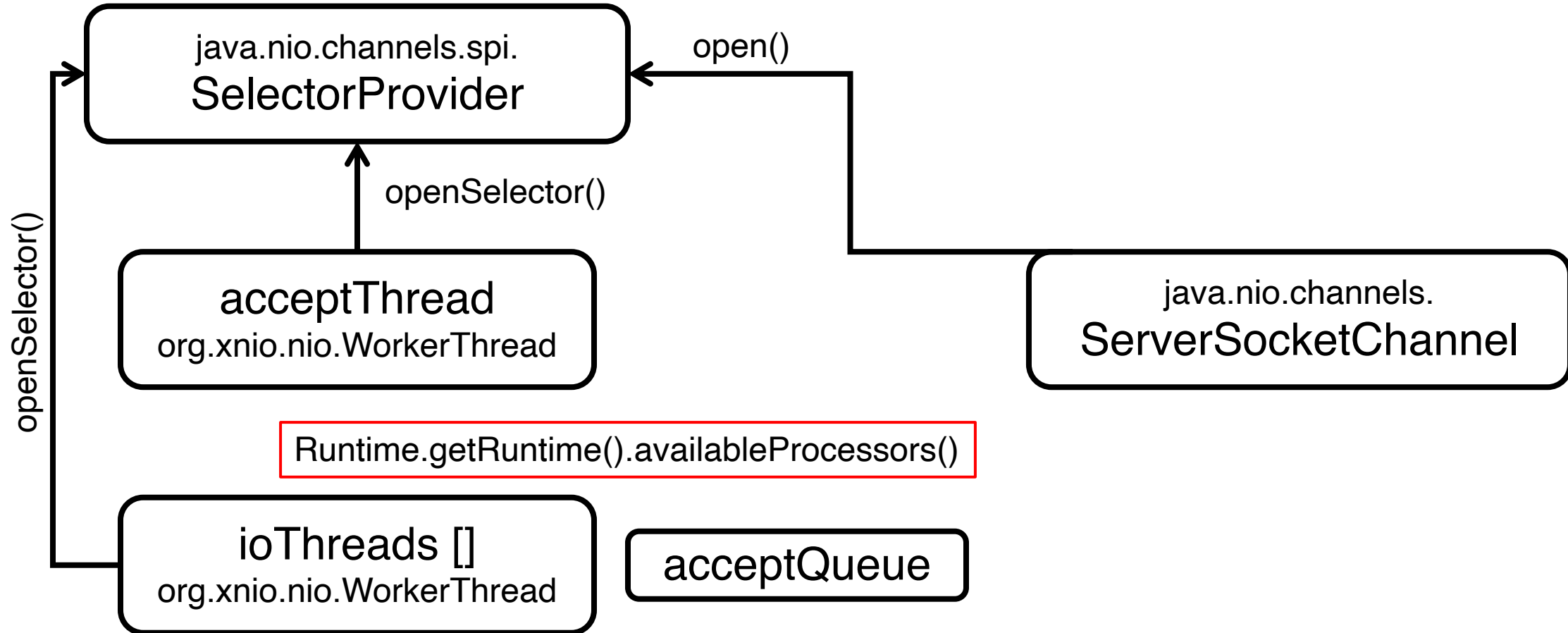
sun.nio.ch.EPollSelectorProvider



Runtime.getRuntime().availableProcessors()

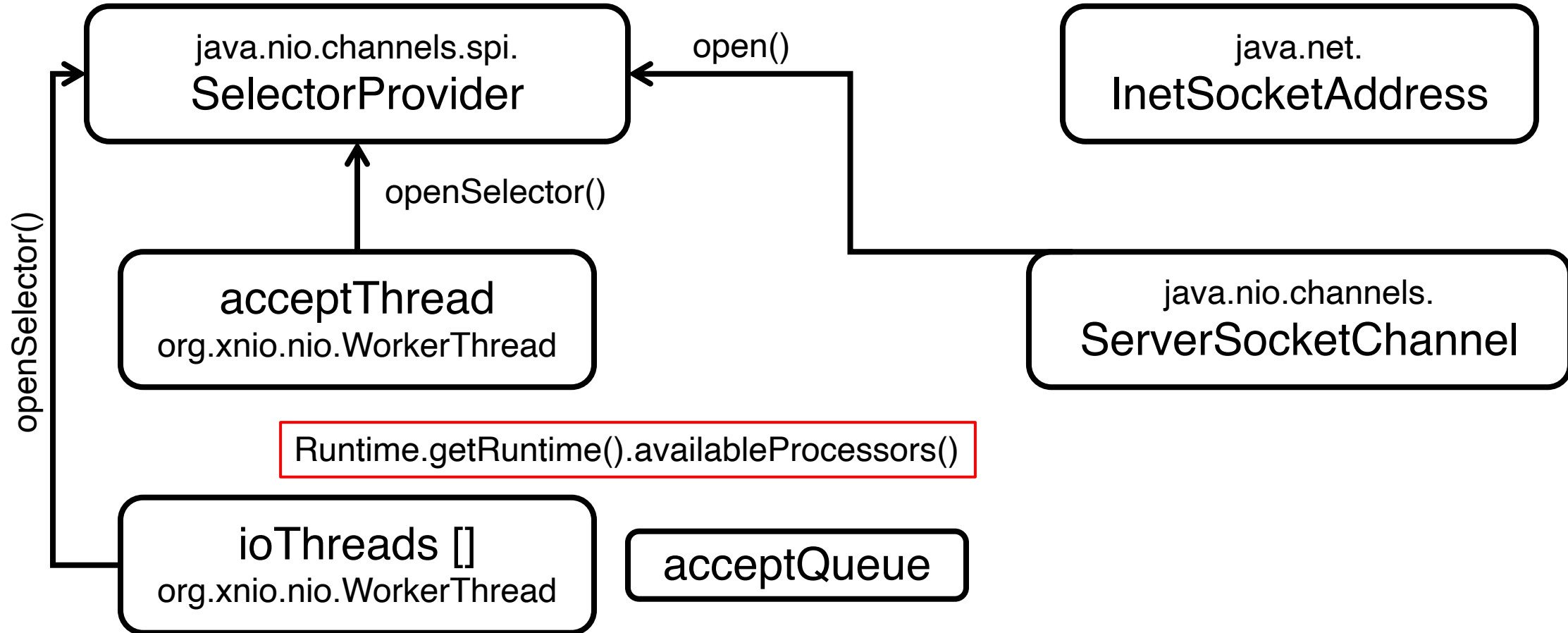
Undertow + XNIO

`sun.nio.ch.EPollSelectorProvider`



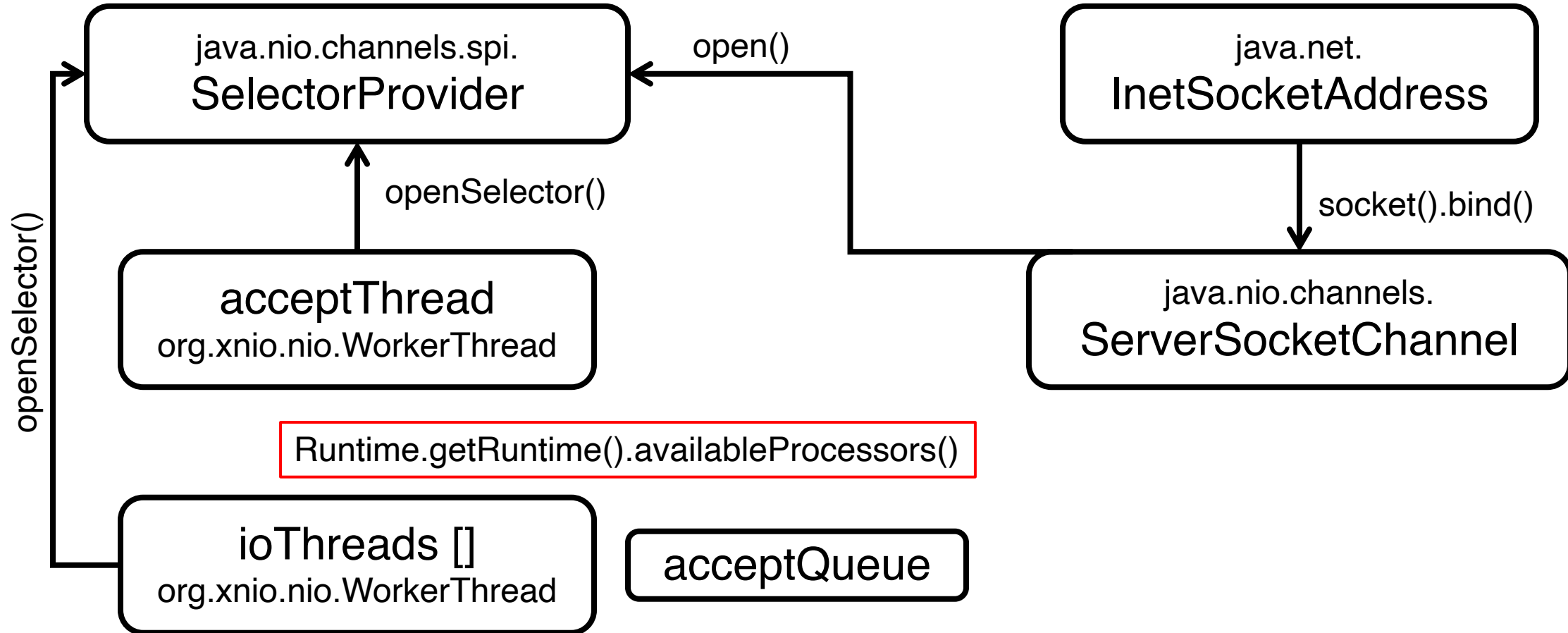
Undertow + XNIO

`sun.nio.ch.EPollSelectorProvider`



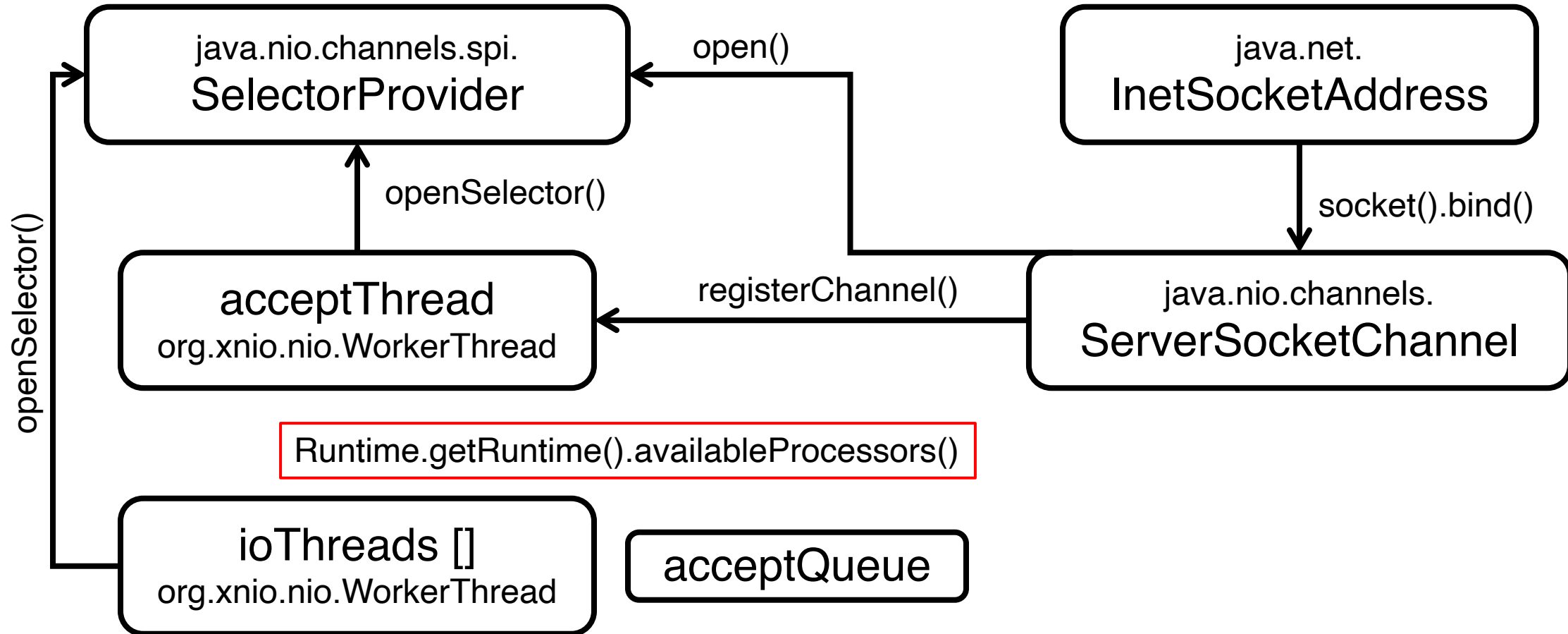
Undertow + XNIO

`sun.nio.ch.EPollSelectorProvider`



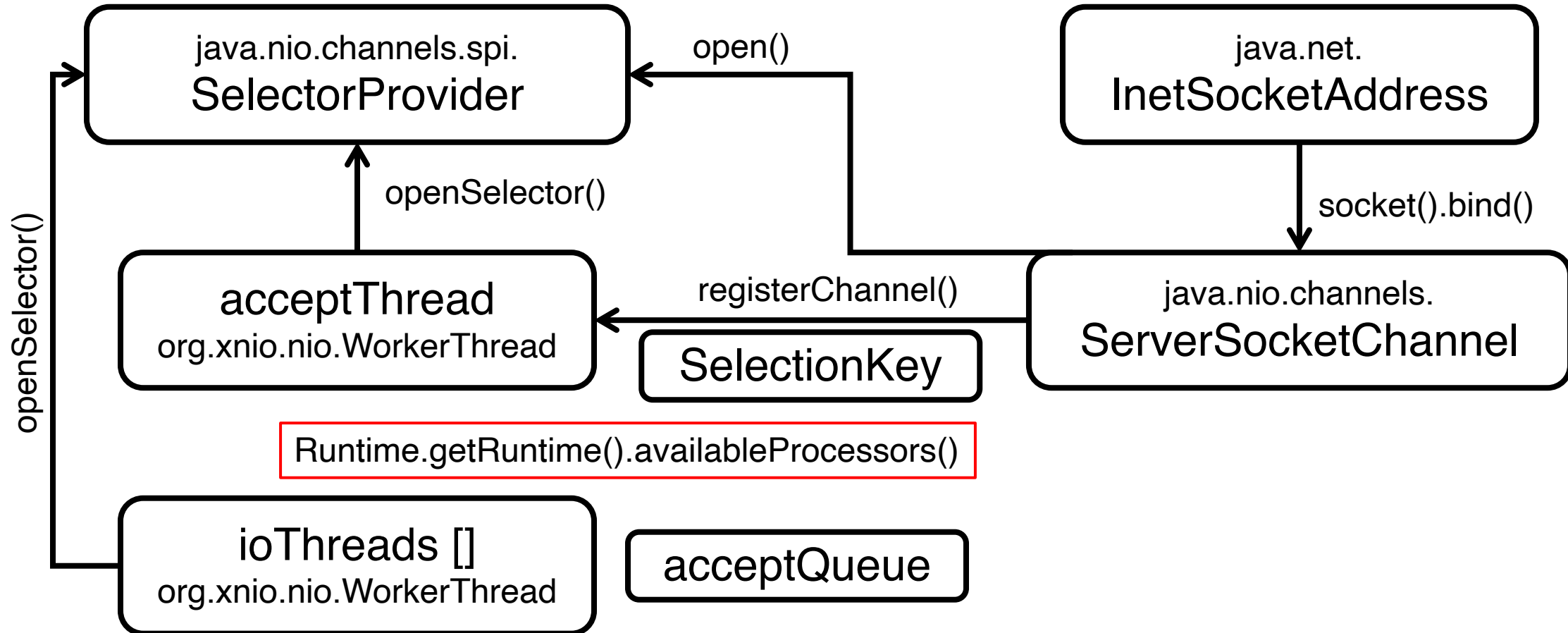
Undertow + XNIO

`sun.nio.ch.EPollSelectorProvider`



Undertow + XNIO

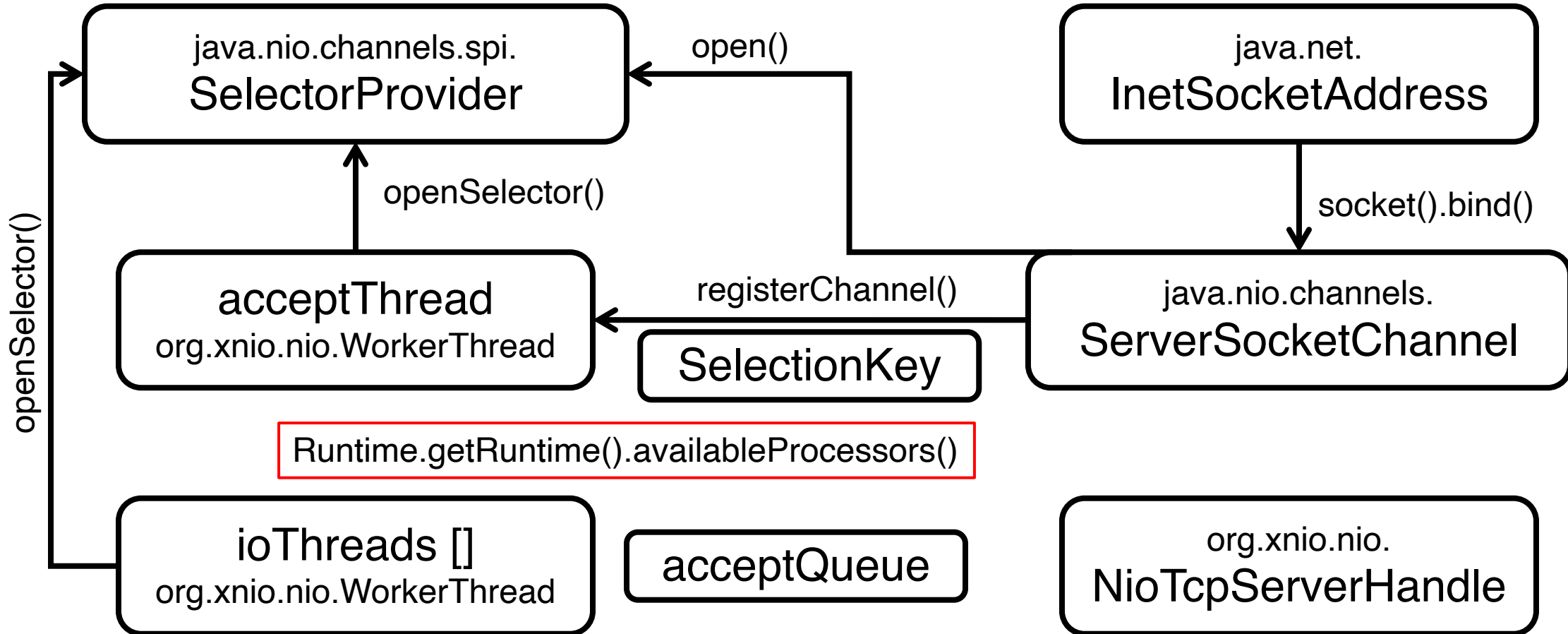
`sun.nio.ch.EPollSelectorProvider`



`Runtime.getRuntime().availableProcessors()`

Undertow + XNIO

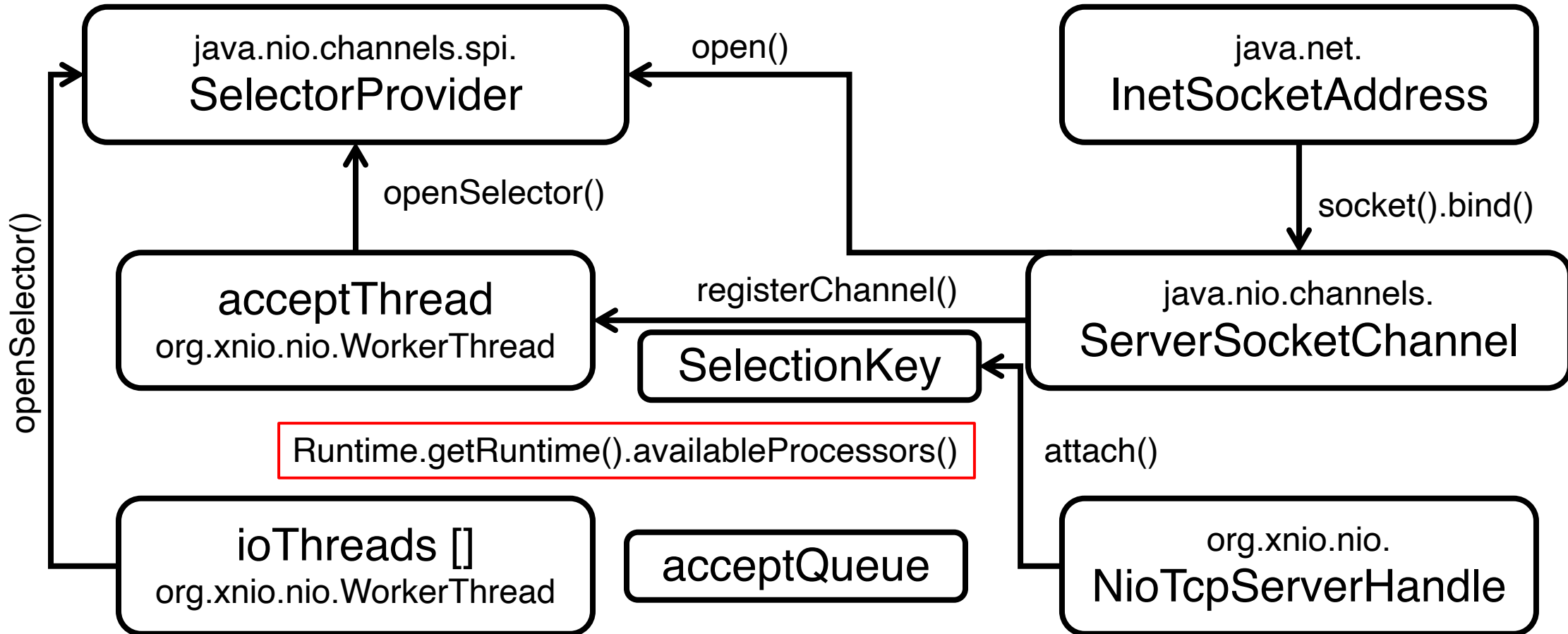
`sun.nio.ch.EPollSelectorProvider`



`Runtime.getRuntime().availableProcessors()`

Undertow + XNIO

`sun.nio.ch.EPollSelectorProvider`



Виды потоков

acceptThread

org.xnio.nio.WorkerThread

Runtime.getRuntime().availableProcessors()

ioThreads []

org.xnio.nio.WorkerThread

Виды потоков

acceptThread

org.xnio.nio.WorkerThread

Runtime.getRuntime().availableProcessors()

ioThreads []

org.xnio.nio.WorkerThread

taskPool

j.u.c.ScheduledExecutorService

Виды потоков

acceptThread
org.xnio.nio.WorkerThread

`Runtime.getRuntime().availableProcessors()`

ioThreads []
org.xnio.nio.WorkerThread

`ioThreads * 8`

taskPool
j.u.c.ScheduledExecutorService

Undertow + XNIO

```
Undertow server = Undertow.builder()
    .addHttpListener(8080, "localhost")
    .setHandler(Handlers.routing()
        2 .get("/hello/{any}", new MyHelloAnyHandler())
        .post("/echo", new MyEchoHandler()) 3
    ).build();
server.start(); 1
```

Undertow + XNIO

```
Undertow server = Undertow.builder()
    .addHttpListener(8080, "localhost")
    .setHandler(Handlers.routing()
        2 .get("/hello/{any}", new MyHelloAnyHandler())
        .post("/echo", new MyEchoHandler()) 3
    ).build();
    1
server.start();
```

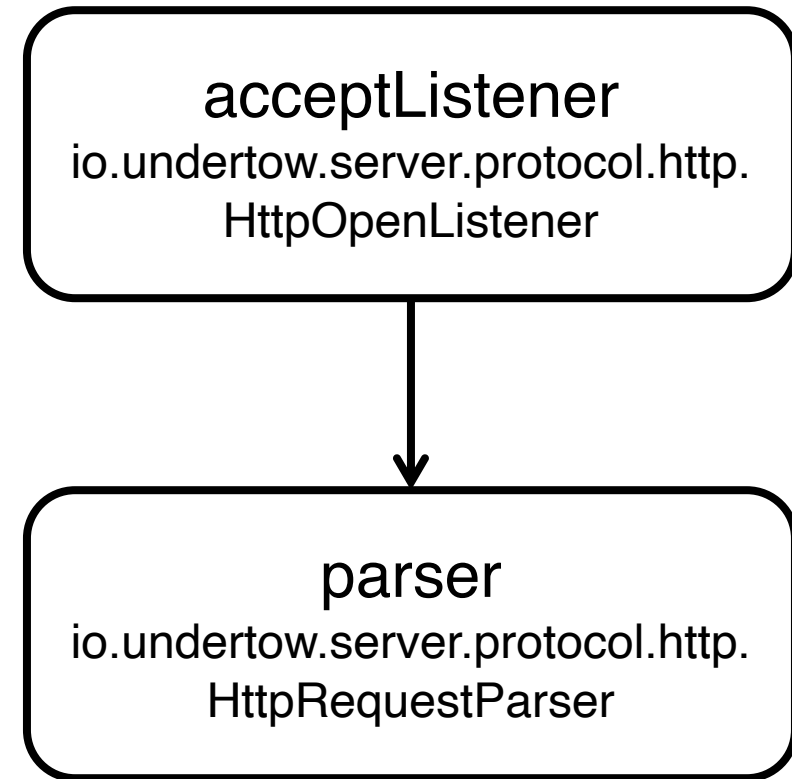
Undertow + XNIO

`acceptListener`

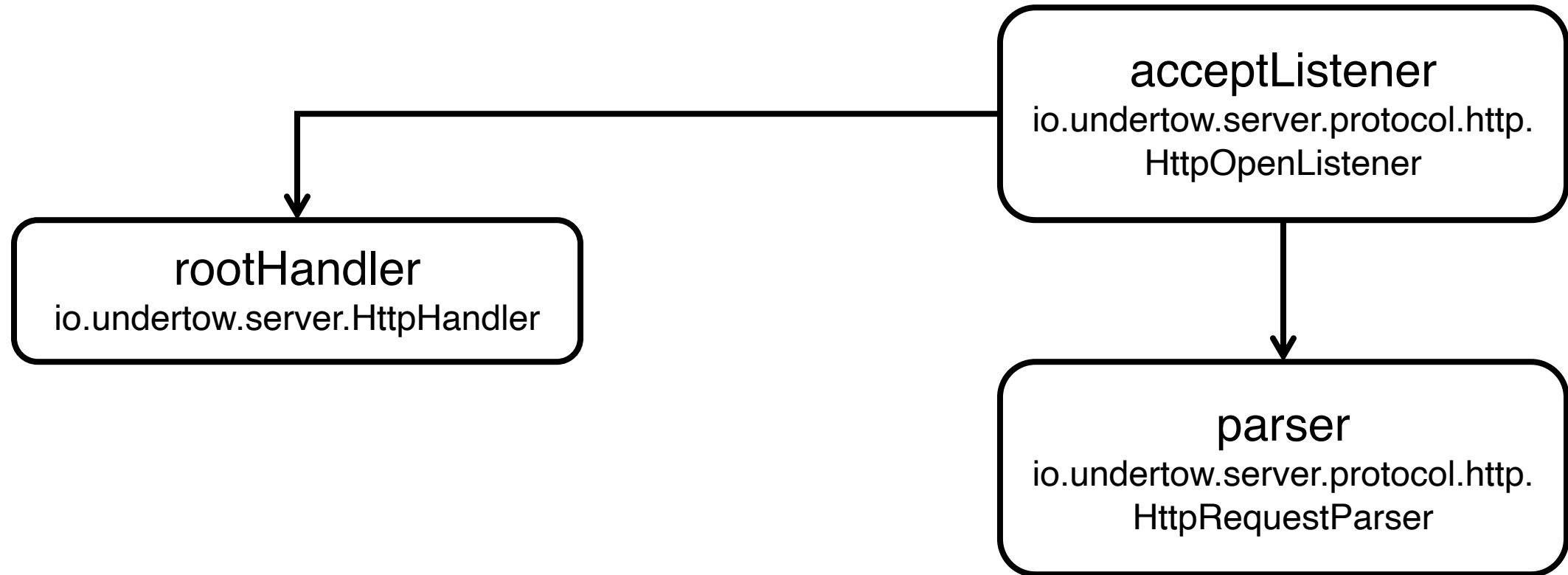
`io.undertow.server.protocol.http.`

`HttpOpenListener`

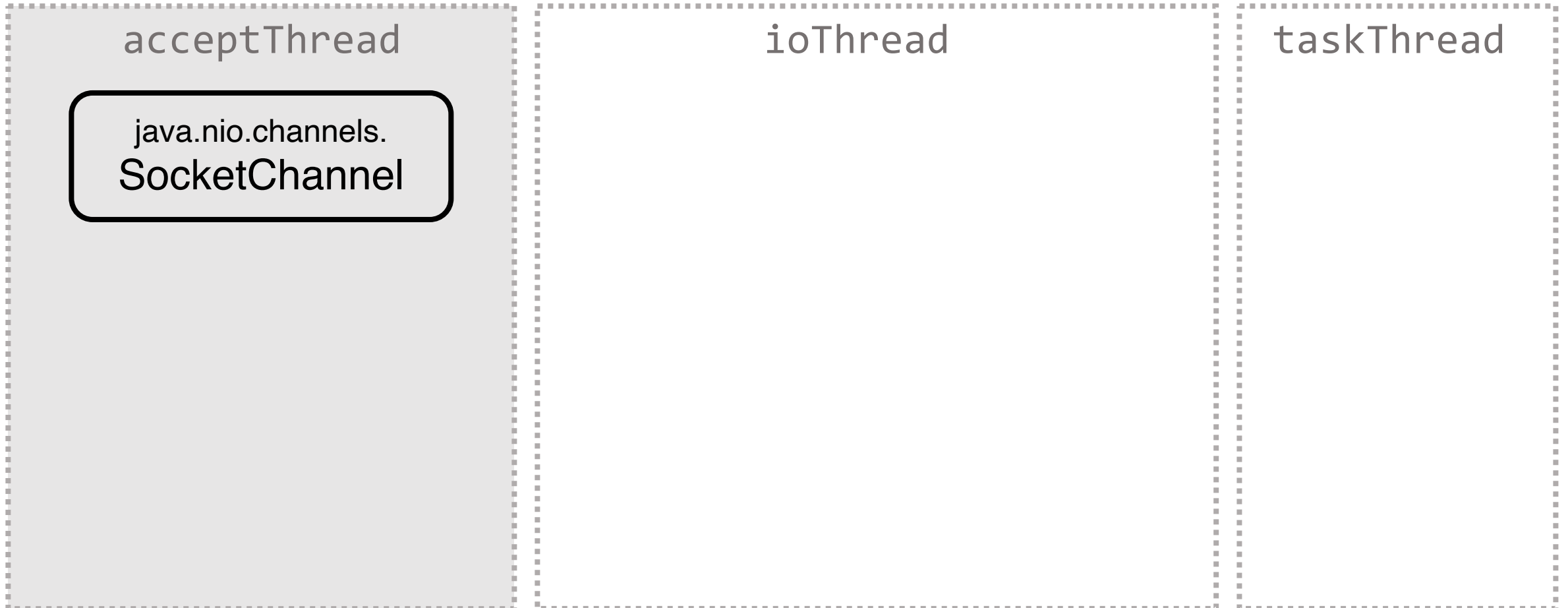
Undertow + XNIO



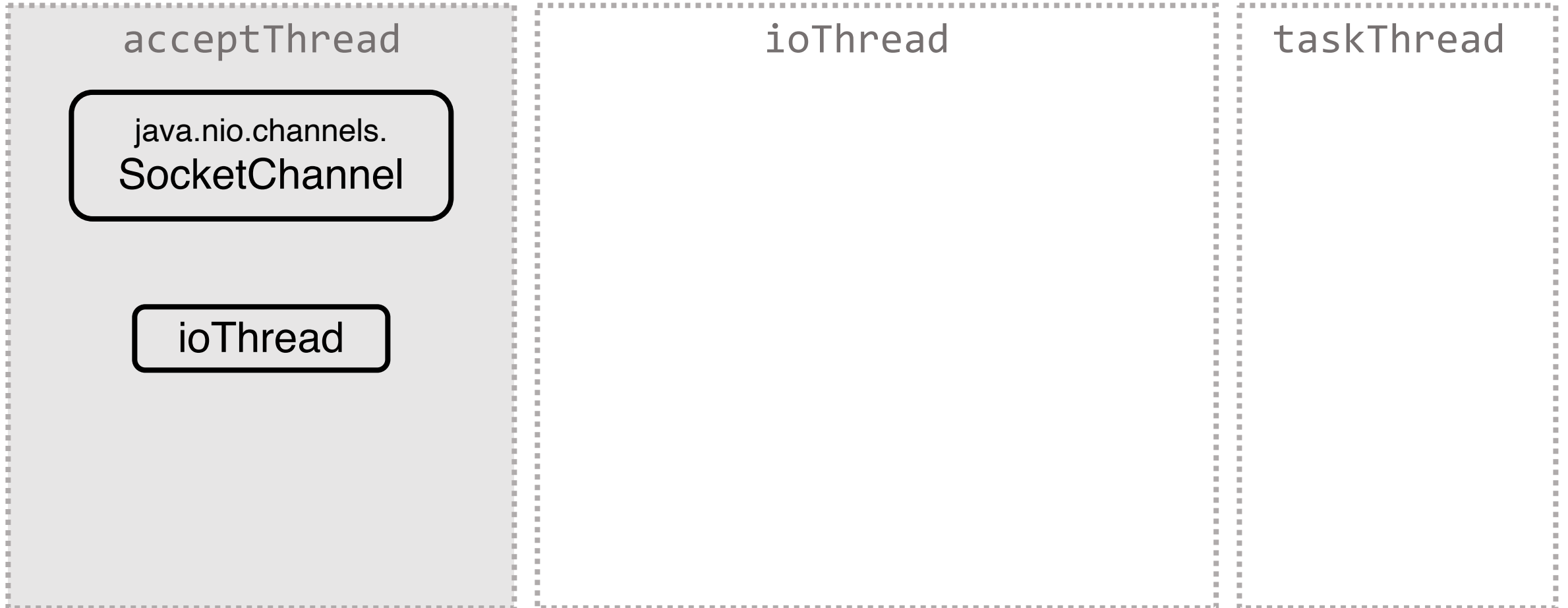
Undertow + XNIO



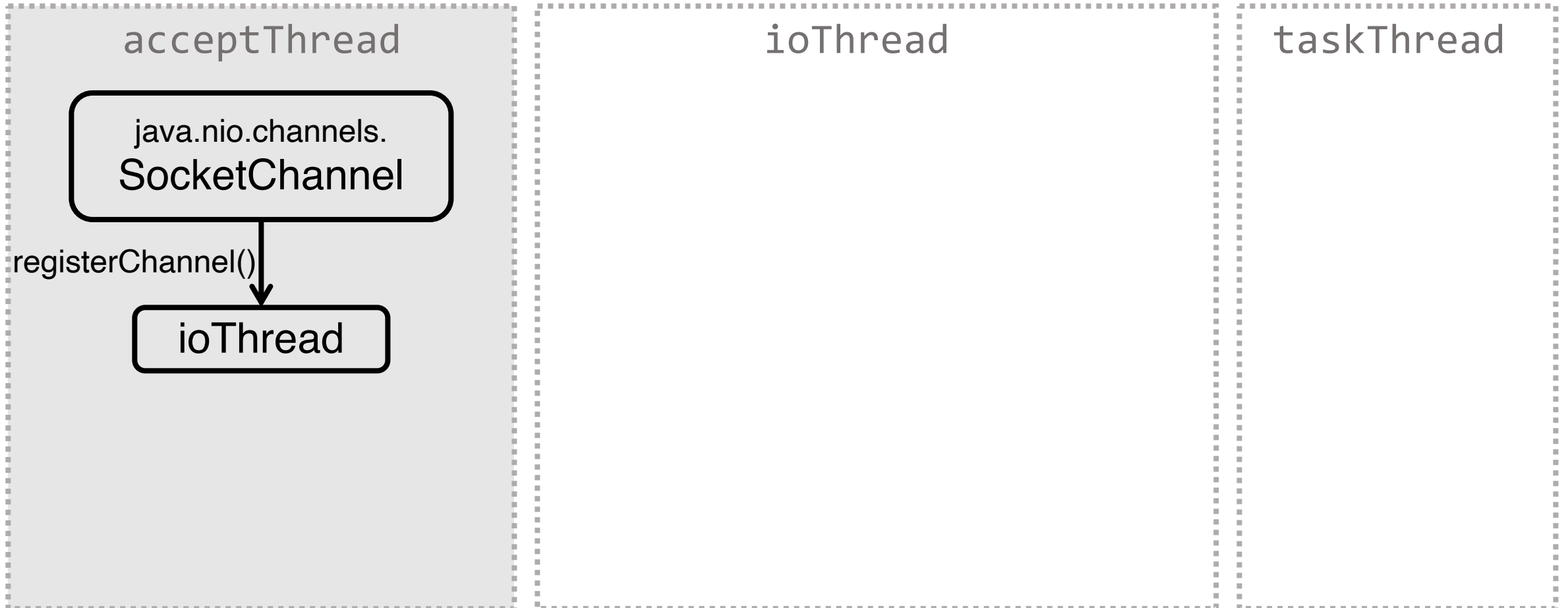
Обработка запроса



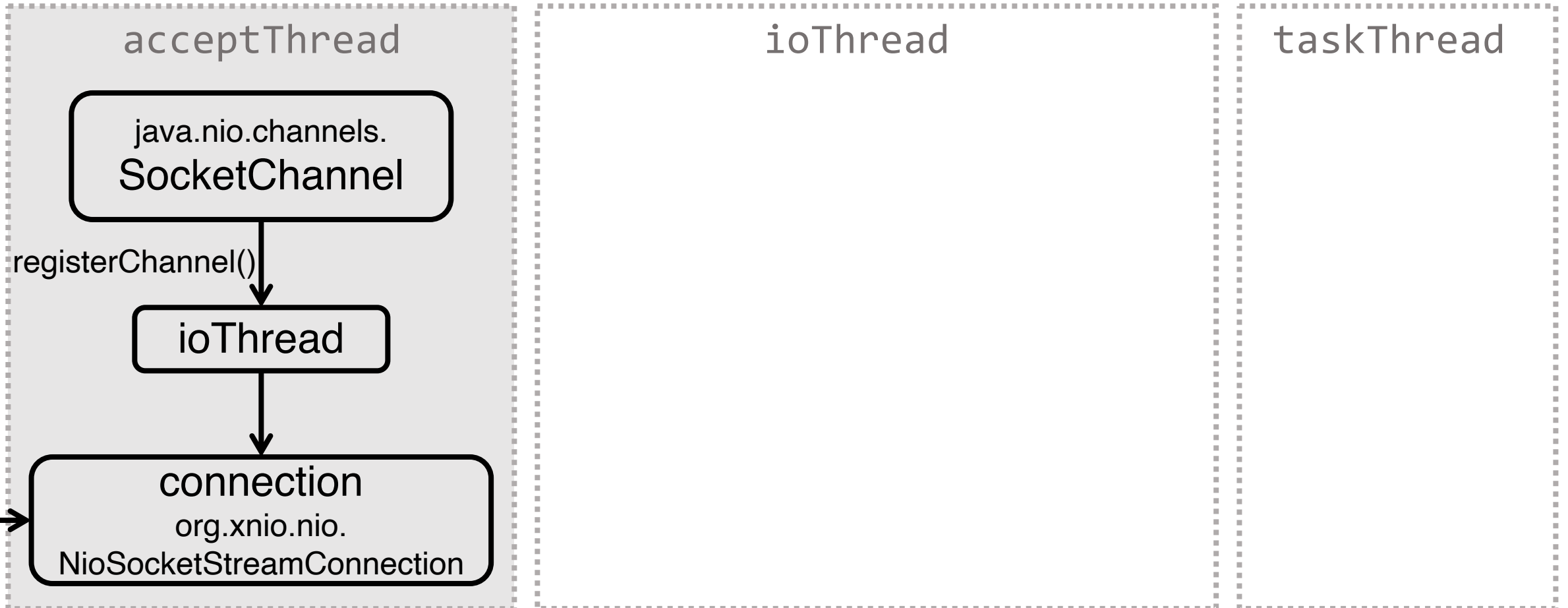
Обработка запроса



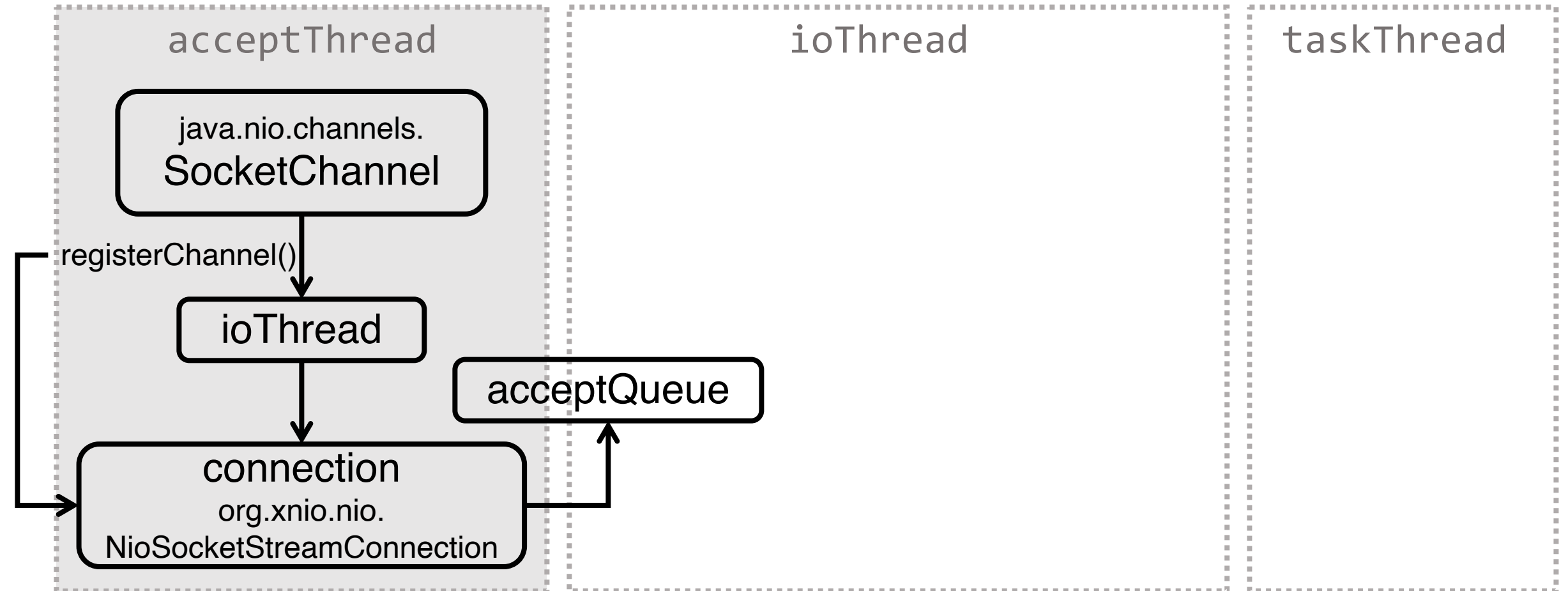
Обработка запроса



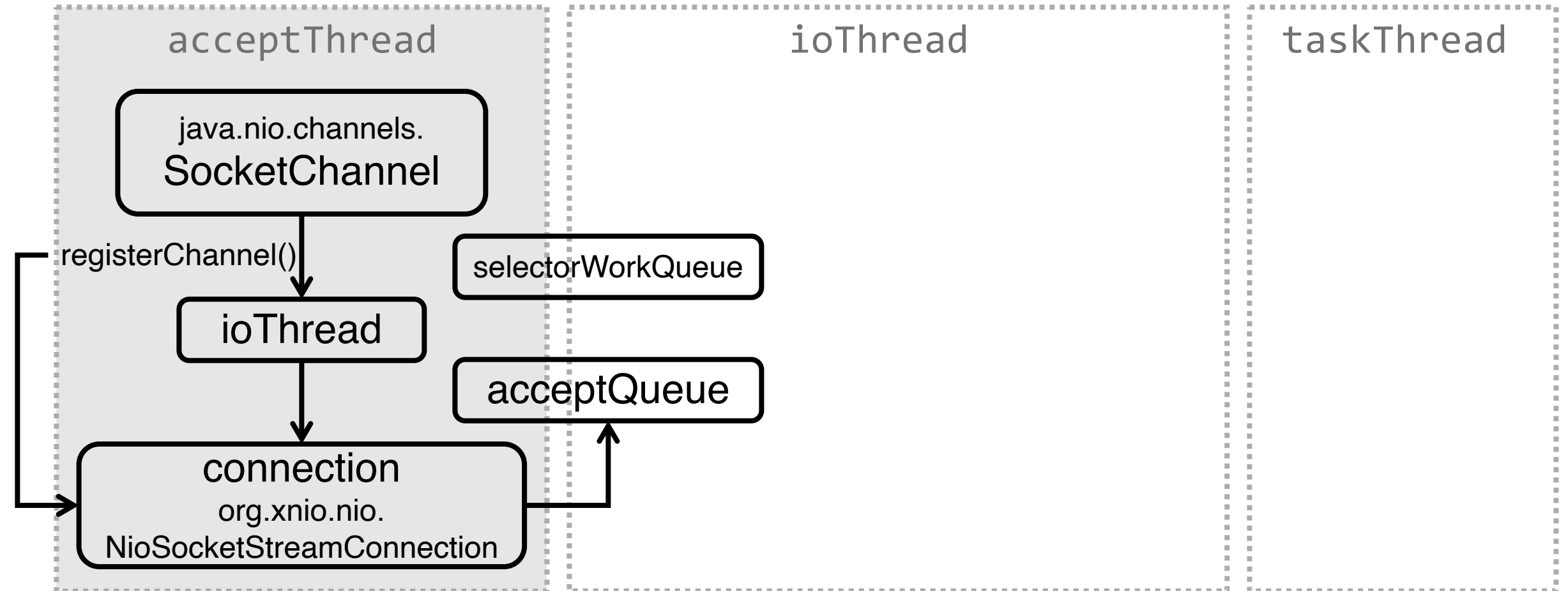
Обработка запроса



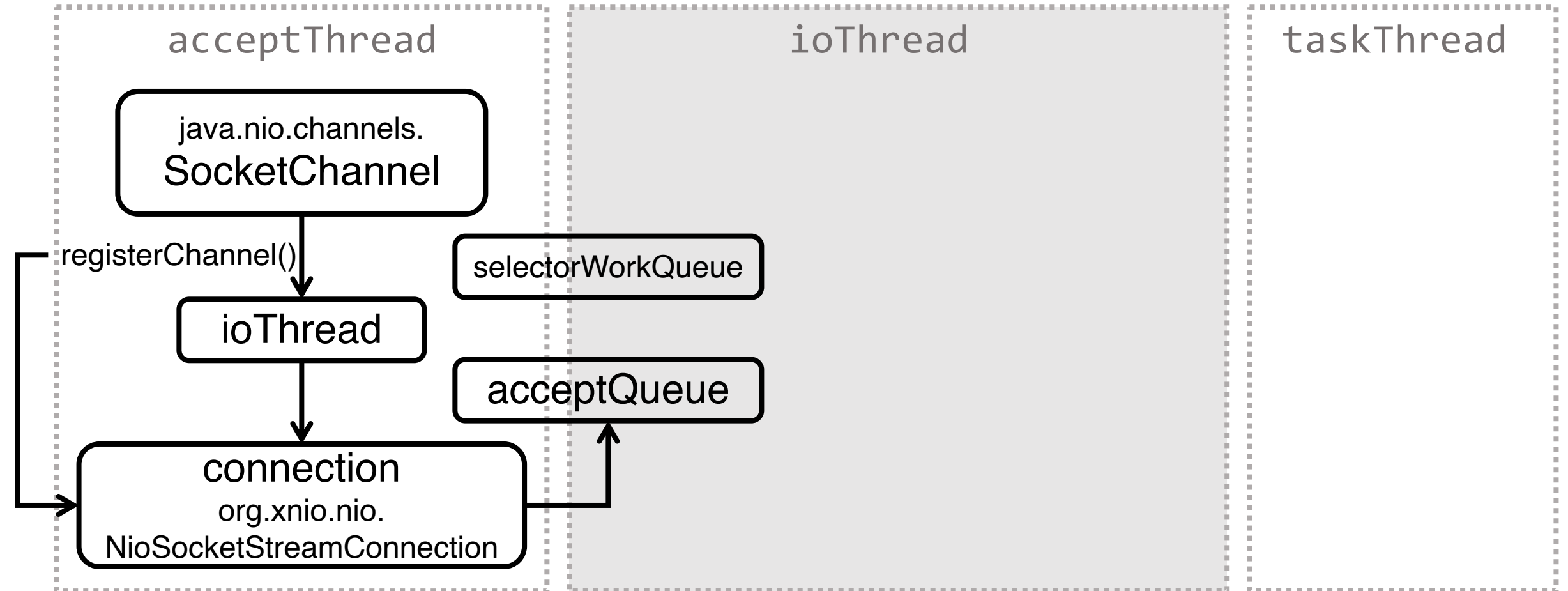
Обработка запроса



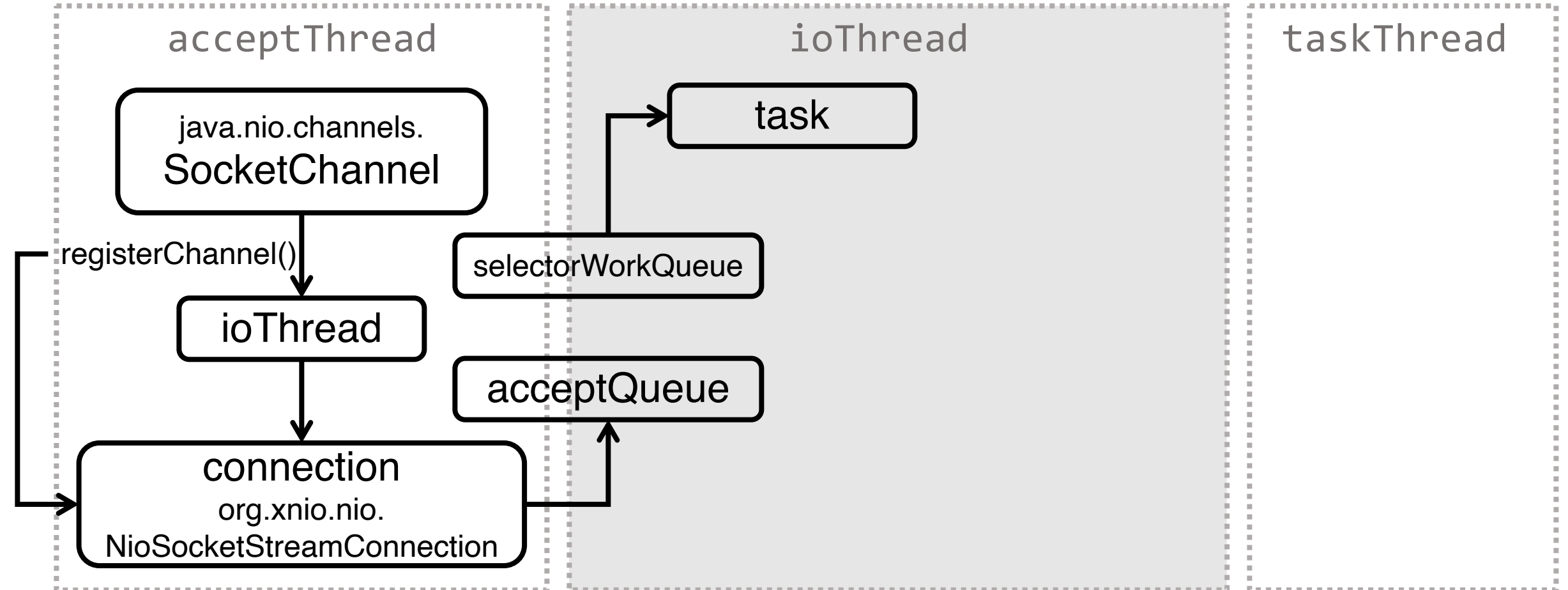
Обработка запроса



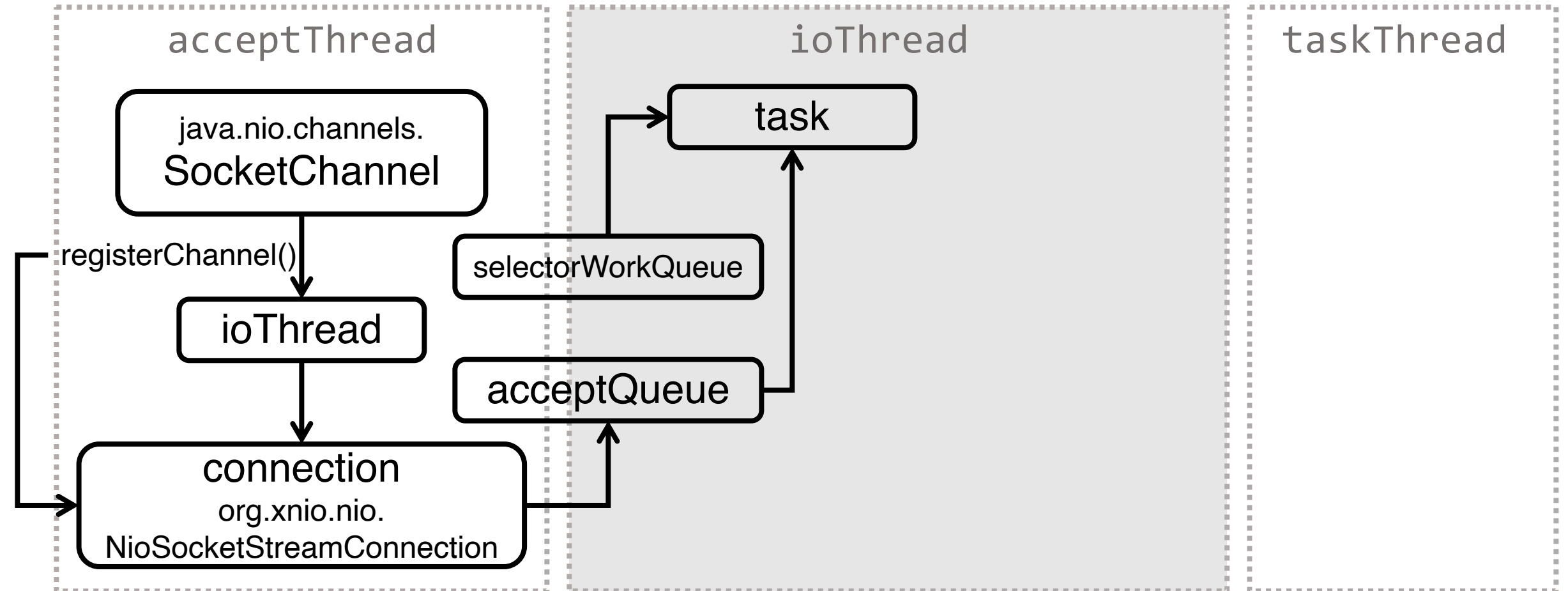
Обработка запроса



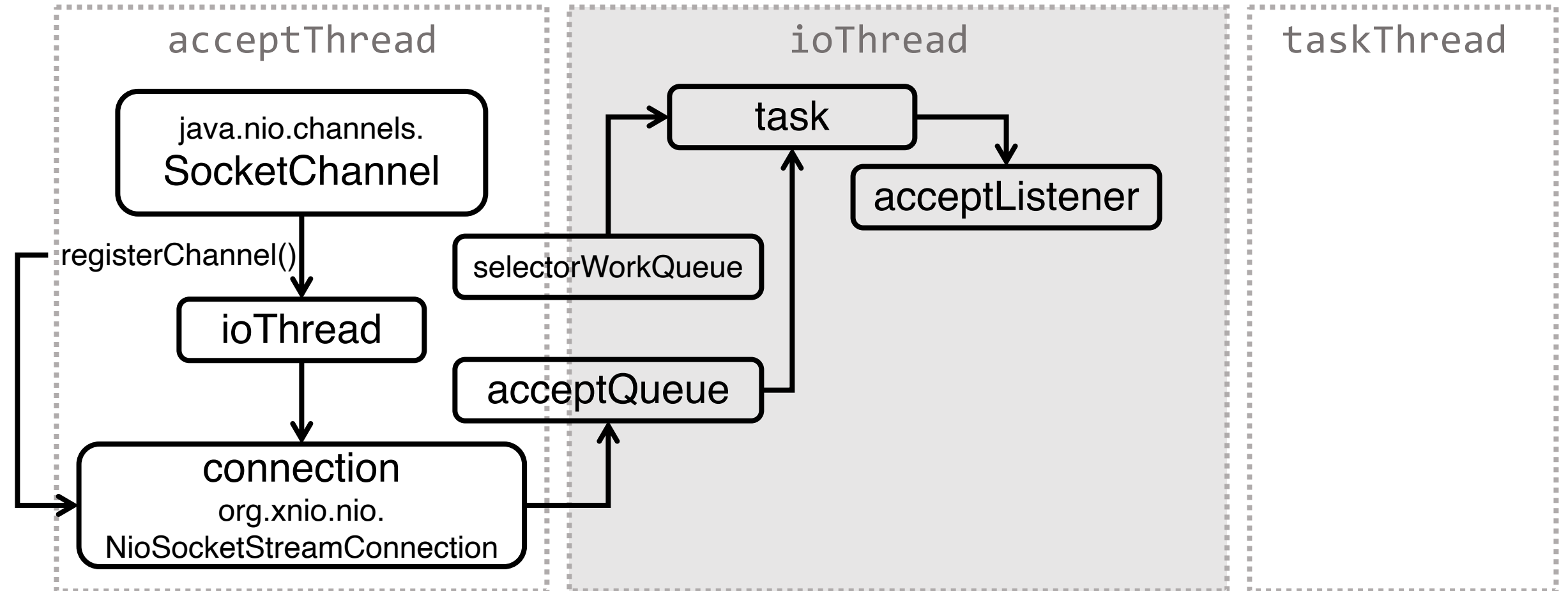
Обработка запроса



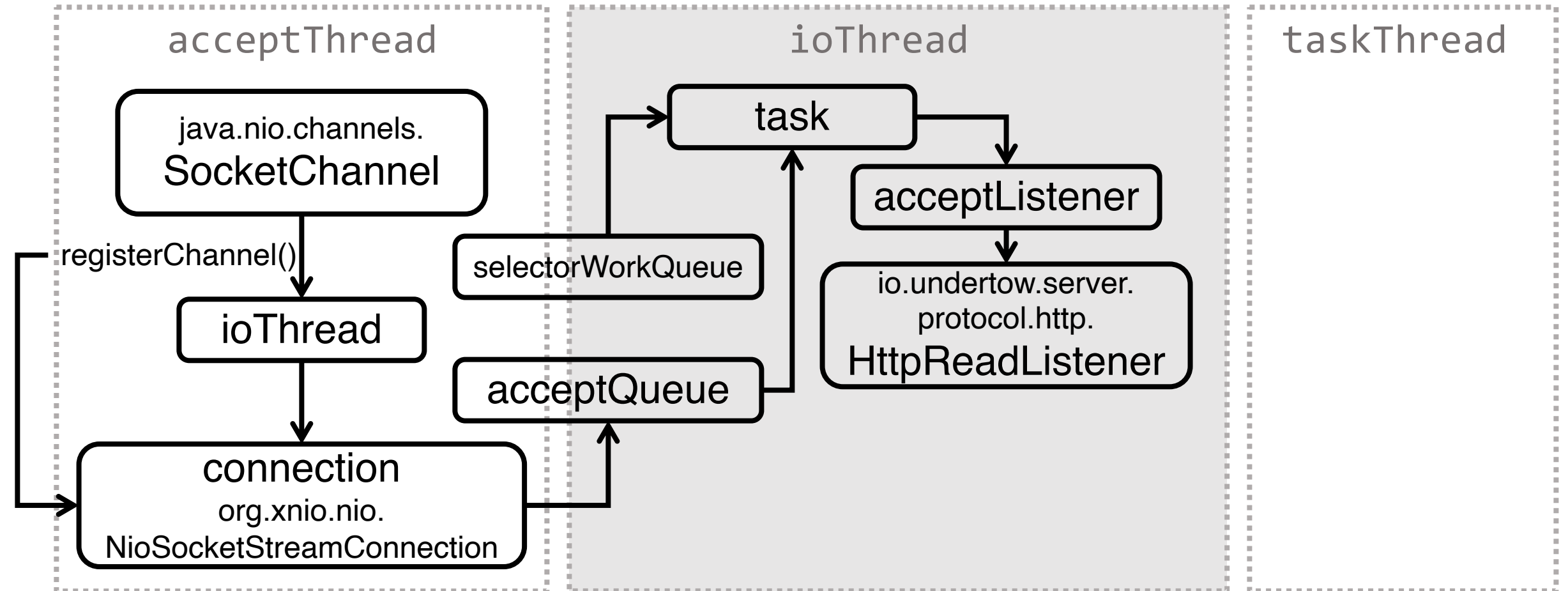
Обработка запроса



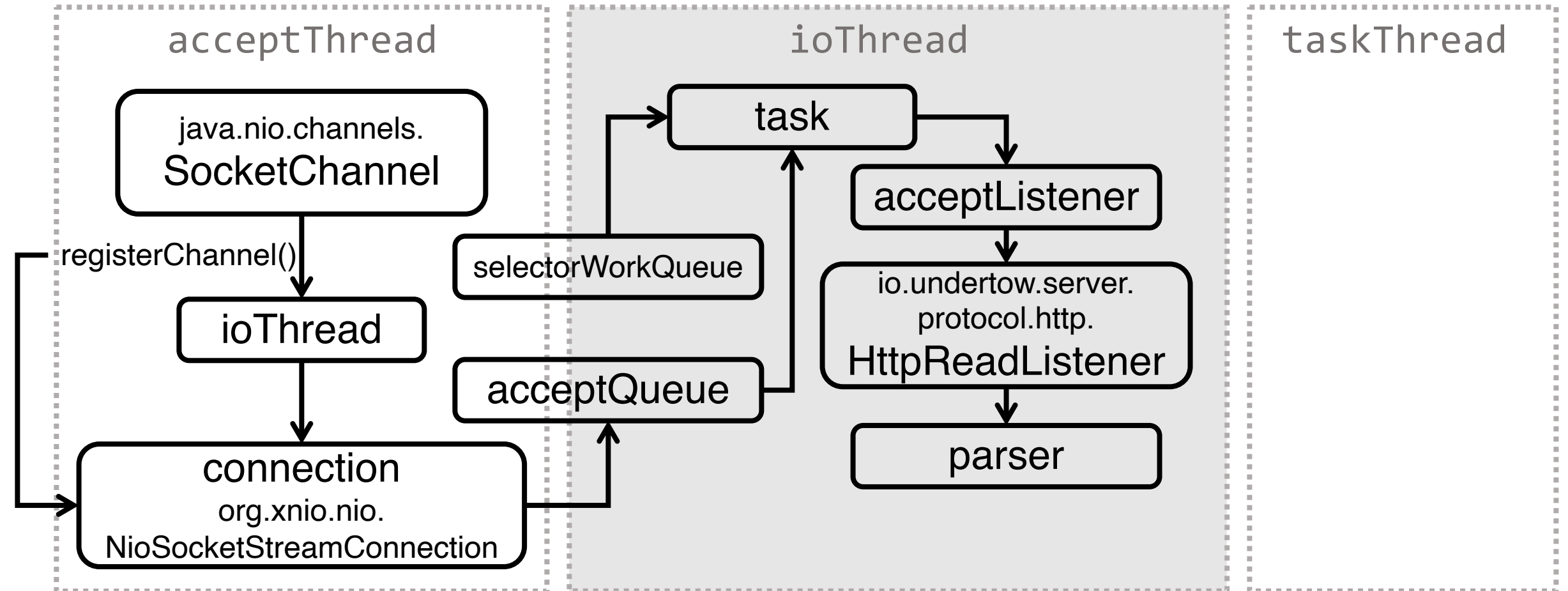
Обработка запроса



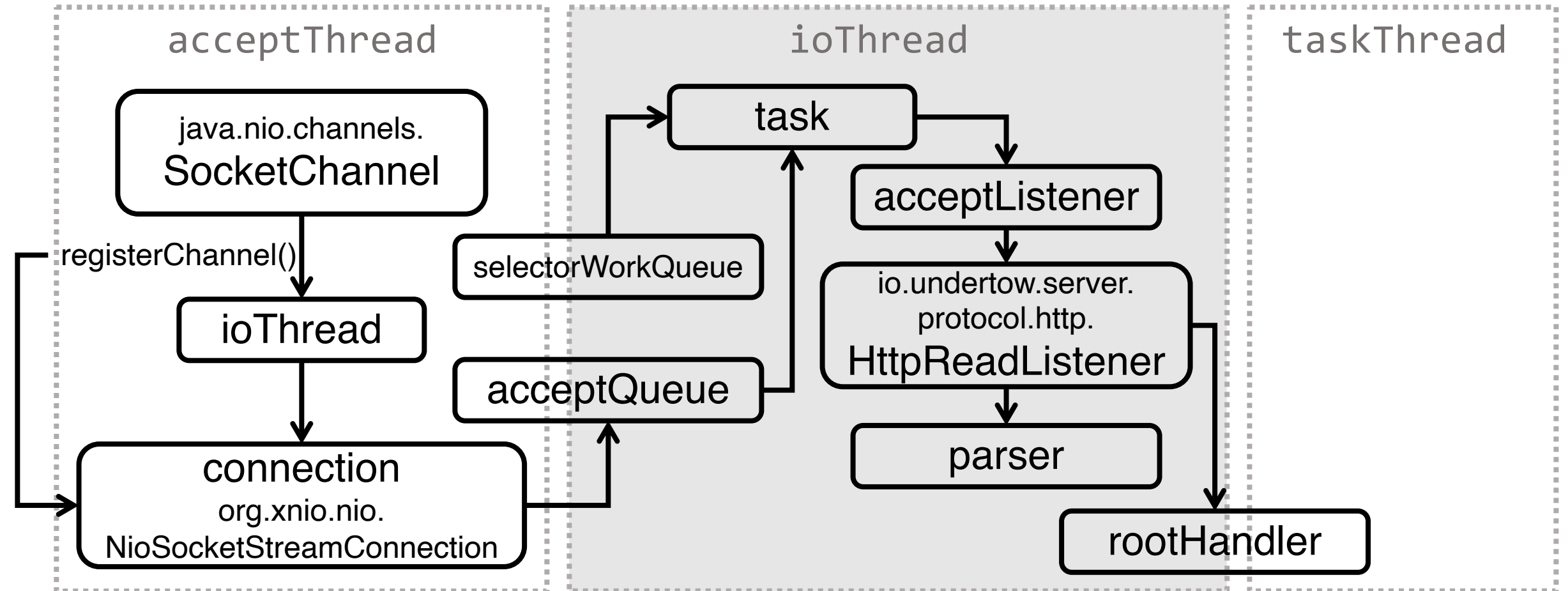
Обработка запроса



Обработка запроса



Обработка запроса



HttpHandler

```
public interface HttpHandler {  
  
    /**  
     * Handle the request.  
     *  
     * @param exchange the HTTP request/response exchange  
     *  
     */  
    void handleRequest(HttpServerExchange exchange)  
                        throws Exception;  
}
```

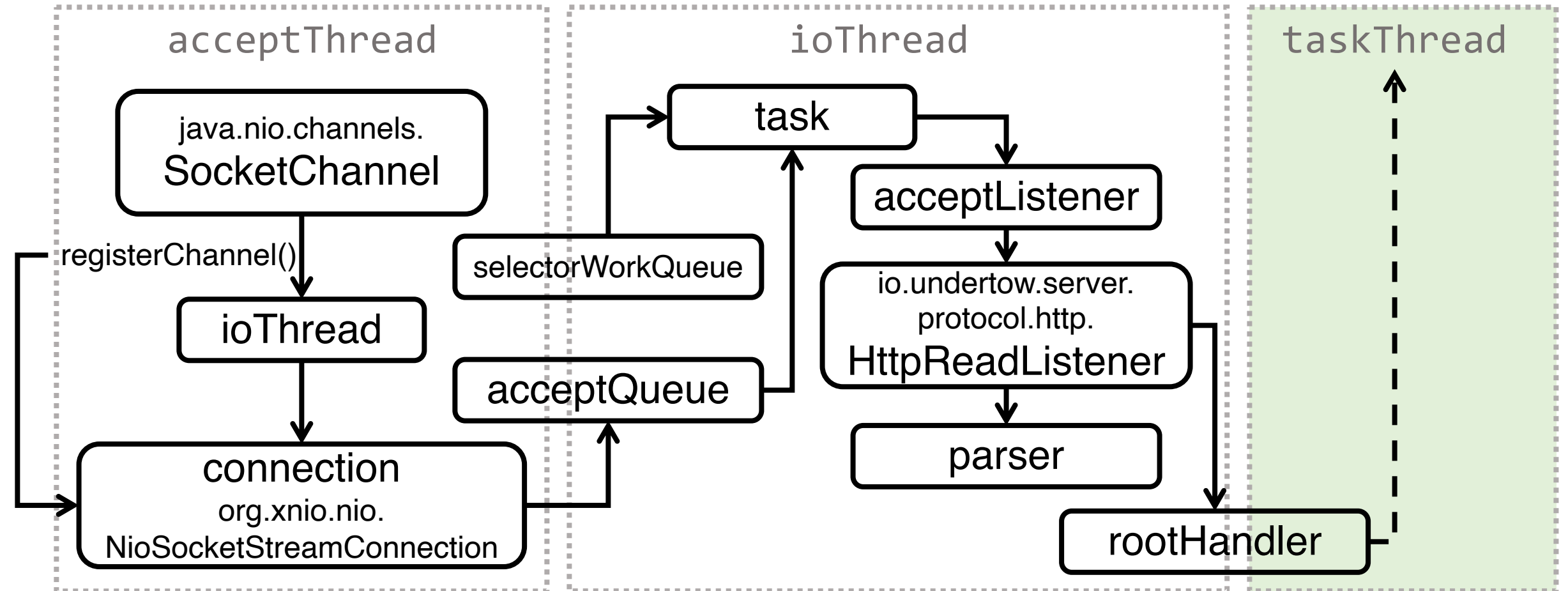
HttpServerExchange

HttpServerExchange

```
public HttpServerExchange dispatch(  
    final Runnable runnable) {  
    /* ... */  
}
```

```
public HttpServerExchange dispatch(  
    final Executor executor,  
    final Runnable runnable) {  
    /* ... */  
}
```

Обработка запроса



HttpServerExchange

```
public Receiver getRequestReceiver() {  
    /* ... */  
    return new AsyncReceiverImpl(this);  
}
```

```
public Sender getResponseSender() {  
    /* ... */  
    return new AsyncSenderImpl(this);  
}
```

HttpServerExchange

```
public Receiver getRequestReceiver() {  
    /* ... */  
    return new AsyncReceiverImpl(this);  
}
```

```
public Sender getResponseSender() {  
    /* ... */  
    return new AsyncSenderImpl(this);  
}
```

AsyncReceiverImpl

AsyncReceiverImpl

```
public void receiveFullBytes(  
    final FullBytesCallback callback,  
    final ErrorCallback errorCallback) {  
    /* ... */  
}  
public void receivePartialBytes(  
    final PartialBytesCallback callback,  
    final ErrorCallback errorCallback) {  
    /* ... */  
}
```

AsyncReceiverImpl

```
public void receiveFullBytes(  
    final FullBytesCallback callback,  
    final ErrorCallback errorCallback) {  
    /* ... */  
}  
  
public void receivePartialBytes(  
    final PartialBytesCallback callback,  
    final ErrorCallback errorCallback) {  
    /* ... */  
}
```

AsyncReceiverImpl

```
exchange.getRequestReceiver().receiveFullBytes(  
    (ex, bytes) -> { /* ioThread */  
        /* bytes -> runnable */  
        ex.dispatch(runnable);  
    },  
    (ex, exception) -> {  
        ex.setStatusCode(StatusCodes.INTERNAL_SERVER_ERROR);  
        ex.endExchange();  
    }  
);
```


AsyncReceiverImpl

```
exchange.getRequestReceiver().receiveFullBytes(  
    (ex, bytes) -> { /* ioThread */  
        /* bytes -> runnable */  
        ex.dispatch(runnable);  
    },  
    (ex, exception) -> {  
        ex.setStatusCode(StatusCode.INTERNAL_SERVER_ERROR);  
        ex.endExchange();  
    }  
);
```

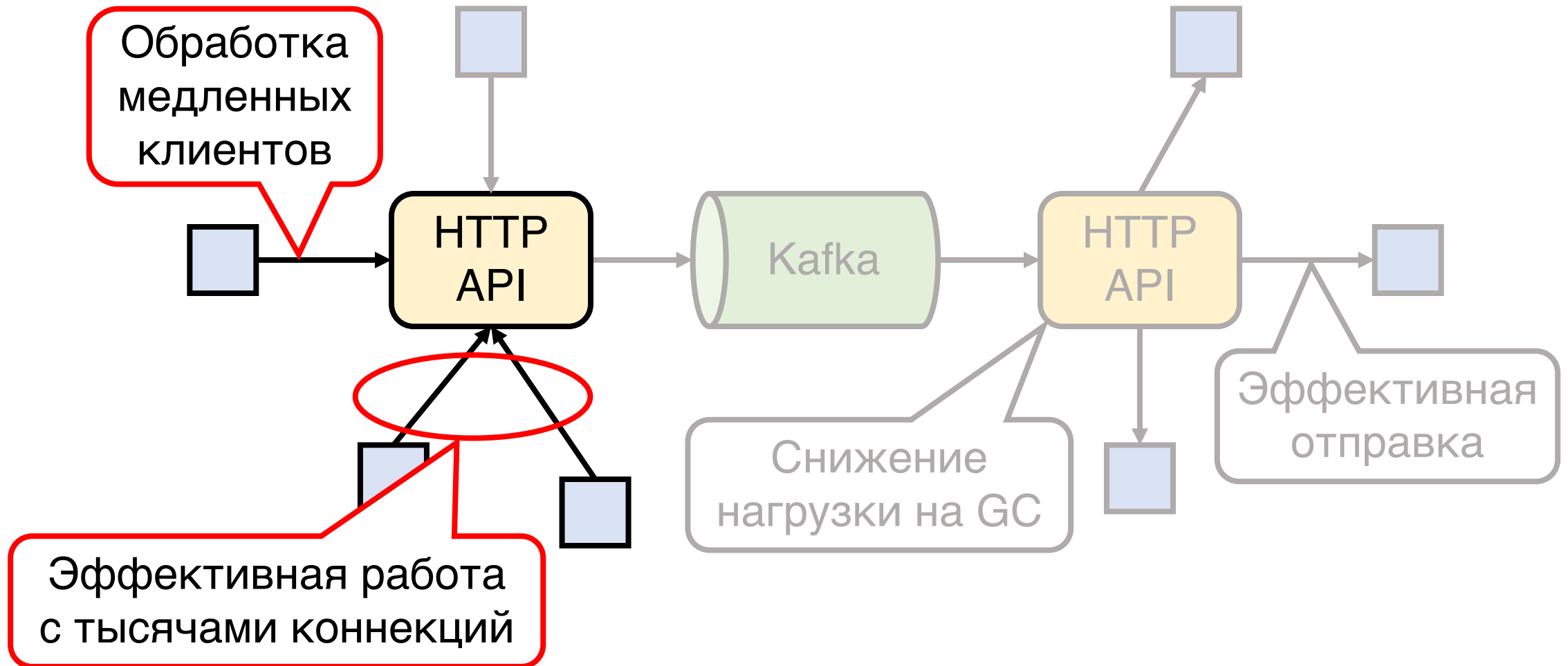
AsyncReceiverImpl

```
exchange.getRequestReceiver().receiveFullBytes(  
    (ex, bytes) -> { /* ioThread */  
        /* bytes -> runnable */  
        ex.dispatch(runnable);  
    },  
    (ex, exception) -> {  
        ex.setStatusCode(StatusCode.INTERNAL_SERVER_ERROR);  
        ex.endExchange();  
    }  
);
```

AsyncReceiverImpl

```
exchange.getRequestReceiver().receiveFullBytes(  
    (ex, bytes) -> { /* ioThread */  
        /* bytes -> runnable */  
        ex.dispatch(runnable);  
    },  
    (ex, exception) -> {  
        ex.setStatusCode(StatusCode.INTERNAL_SERVER_ERROR);  
        ex.endExchange();  
    }  
);
```

Производительность и оптимизации



HttpServerExchange

```
public Receiver getRequestReceiver() {  
    /* ... */  
    return new AsyncReceiverImpl(this);  
}
```

```
public Sender getResponseSender() {  
    /* ... */  
    return new AsyncSenderImpl(this);  
}
```

AsyncSenderImpl

AsyncSenderImpl

```
public void send(  
    final ByteBuffer buffer,  
    final IoCallback callback) {  
    /* ... */  
}  
  
public void send(  
    final ByteBuffer[] buffer,  
    final IoCallback callback) {  
    /* ... */  
}
```

AsyncSenderImpl

```
public void send(  
    final ByteBuffer buffer,  
    final IoCallback callback) {  
    /* ... */  
}  
  
public void send(  
    final ByteBuffer[] buffer,  
    final IoCallback callback) {  
    /* ... */  
}
```


AsyncSenderImpl

```
public void send(  
    final ByteBuffer buffer,  
    final IoCallback callback) {  
    /* ... */  
}  
  
public void send(  
    final ByteBuffer[] buffer,  
    final IoCallback callback) {  
    /* ... */  
}
```

AsyncSenderImpl.send()

```
do {  
    long res = channel.write(buffer);  
    written += res;  
    if (res == 0) {  
        this.buffer = buffer;  
        this.callback = callback;  
        /* ... */  
        channel.getWriter().set(writeListener);  
        channel.resumeWrites();  
        return;  
    }  
} while (written < total);
```

AsyncSenderImpl.send()

```
do {  
    long res = channel.write(buffer);  
    written += res;  
    if (res == 0) {  
        this.buffer = buffer;  
        this.callback = callback;  
        /* ... */  
        channel.getWriter().set(writeListener);  
        channel.resumeWrites();  
        return;  
    }  
} while (written < total);
```

AsyncSenderImpl.send()

```
do {  
    long res = channel.write(buffer);  
    written += res;  
    if (res == 0) {  
        this.buffer = buffer;  
        this.callback = callback;  
        /* ... */  
        channel.getWriter().set(writeListener);  
        channel.resumeWrites();  
        return;  
    }  
} while (written < total);
```

AsyncSenderImpl.send()

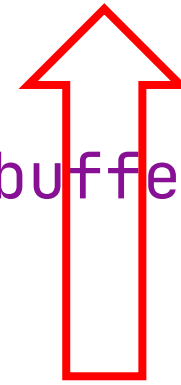
```
do {
    long res = channel.write(buffer);
    written += res;
    if (res == 0) {
        this.buffer = buffer;
        this.callback = callback;
        /* ... */
        channel.getWriter().set(writeListener);
        channel.resumeWrites();
        return;
    }
} while (written < total);
```

AsyncSenderImpl.writeListener

```
public void handleEvent(final StreamSinkChannel ch) {  
    try {  
        long toWrite = Buffers.remaining(buffer);  
        long written = 0;  
        while (written < toWrite) {  
            long res = ch.write(buffer, 0, buffer.length);  
            written += res;  
            if (res == 0) { return; }  
        }  
        /* ... */  
    } catch (IOException e) {  
        /* ... */  
    }  
}
```

AsyncSenderImpl.writeListener

```
public void handleEvent(final StreamSinkChannel ch) {  
    try {  
        long toWrite = Buffers.remaining(buffer);  
        long written = 0;  
        while (written < toWrite) {  
            long res = ch.write(buffer, 0, buffer.length);  
            written += res;  
            if (res == 0) { return; }  
        }  
        /* ... */  
    } catch (IOException e) {  
        /* ... */  
    }  
}
```



ByteBuffer[]

AsyncSenderImpl.writeListener

```
public void handleEvent(final StreamSinkChannel ch) {
    try {
        long toWrite = Buffers.remaining(buffer);
        long written = 0;
        while (written < toWrite) {
            long res = ch.write(buffer, 0, buffer.length);
            written += res;
            if (res == 0) { return; }
        }
        /* ... */
    } catch (IOException e) {
        /* ... */
    }
}
```

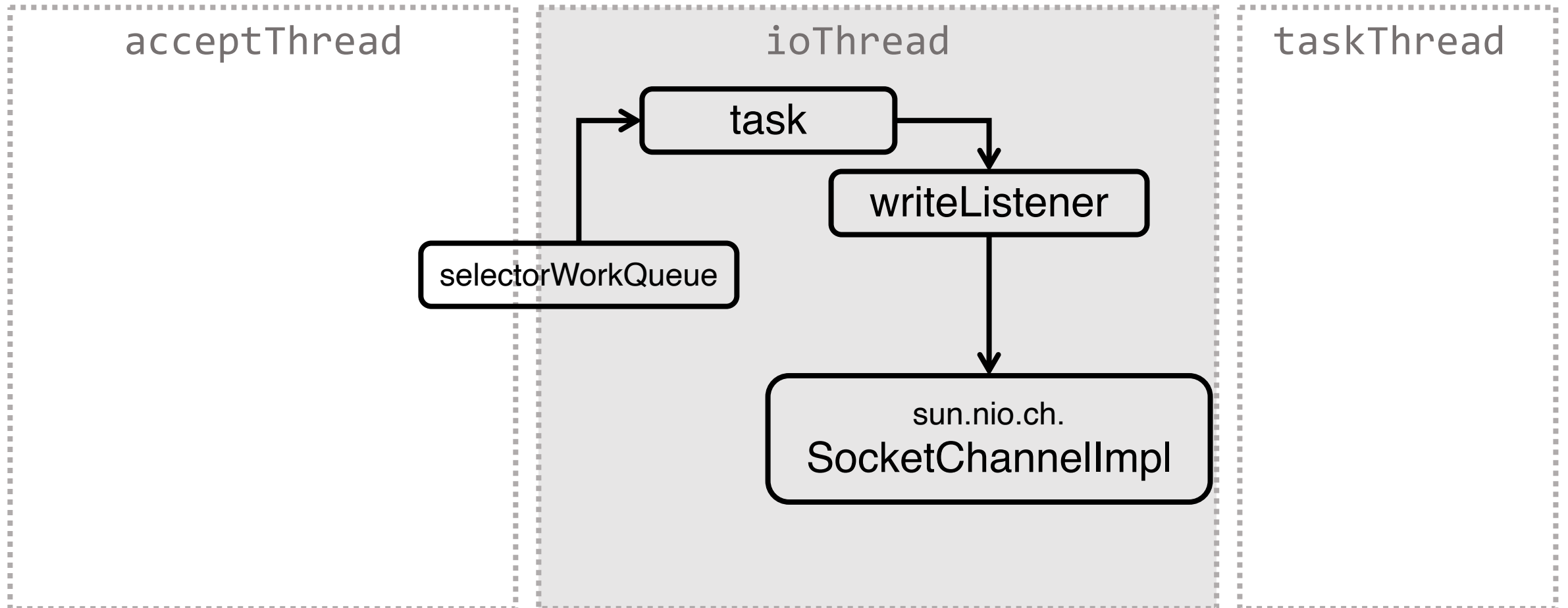

AsyncSenderImpl.send()

```
do {  
    long res = channel.write(buffer);  
    written += res;  
    if (res == 0) {  
        this.buffer = buffer;  
        this.callback = callback;  
        /* ... */  
        channel.getWriter().set(writeListener);  
        channel.resumeWrites();  
        return;  
    }  
} while (written < total);
```

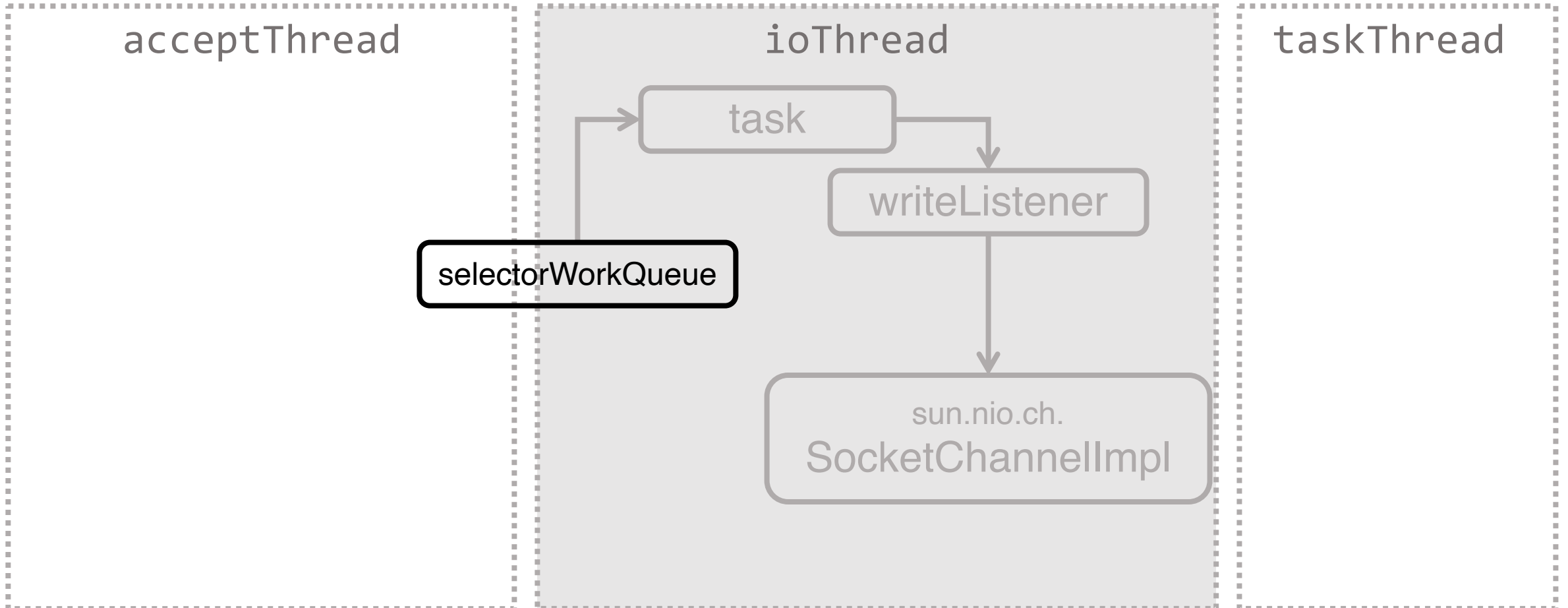
AsyncSenderImpl.send()

```
do {
    long res = channel.write(buffer);
    written += res;
    if (res == 0) {
        this.buffer = buffer;
        this.callback = callback;
        /* ... */
        channel.getWriter().set(writeListener);
        channel.resumeWrites();
        return;
    }
} while (written < total);
```

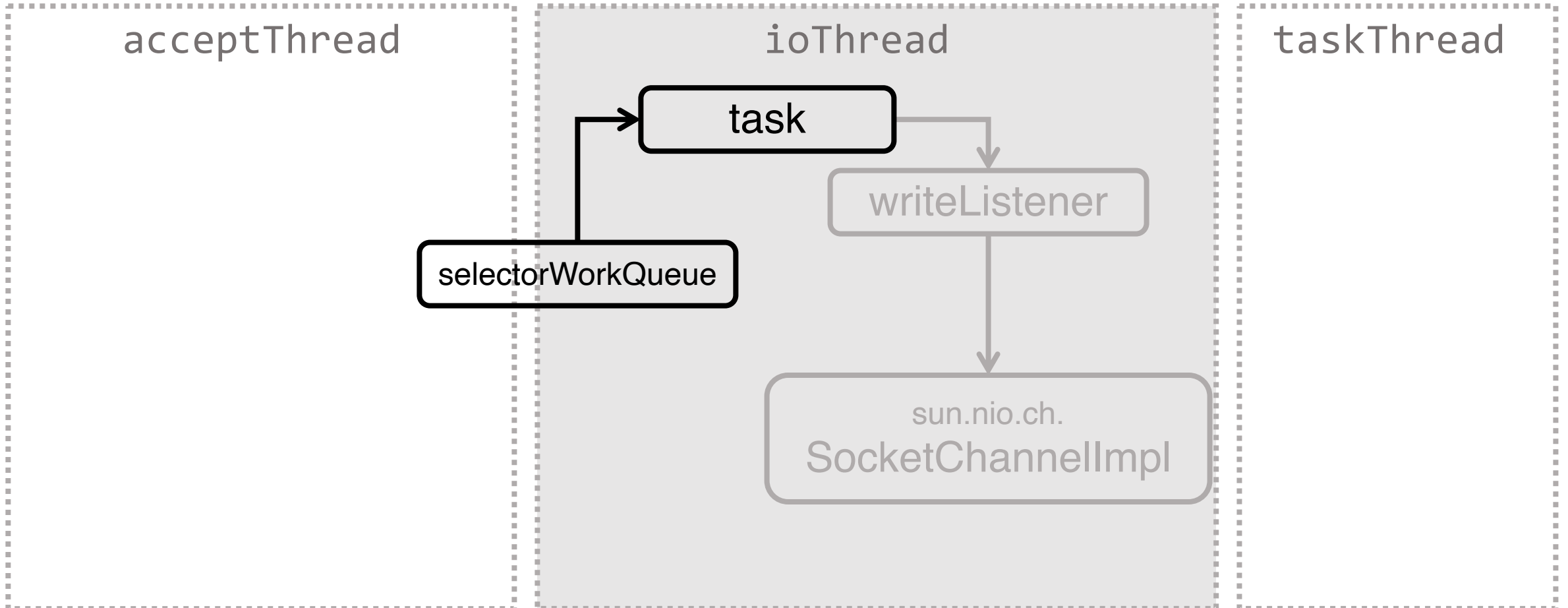
Обработка запроса



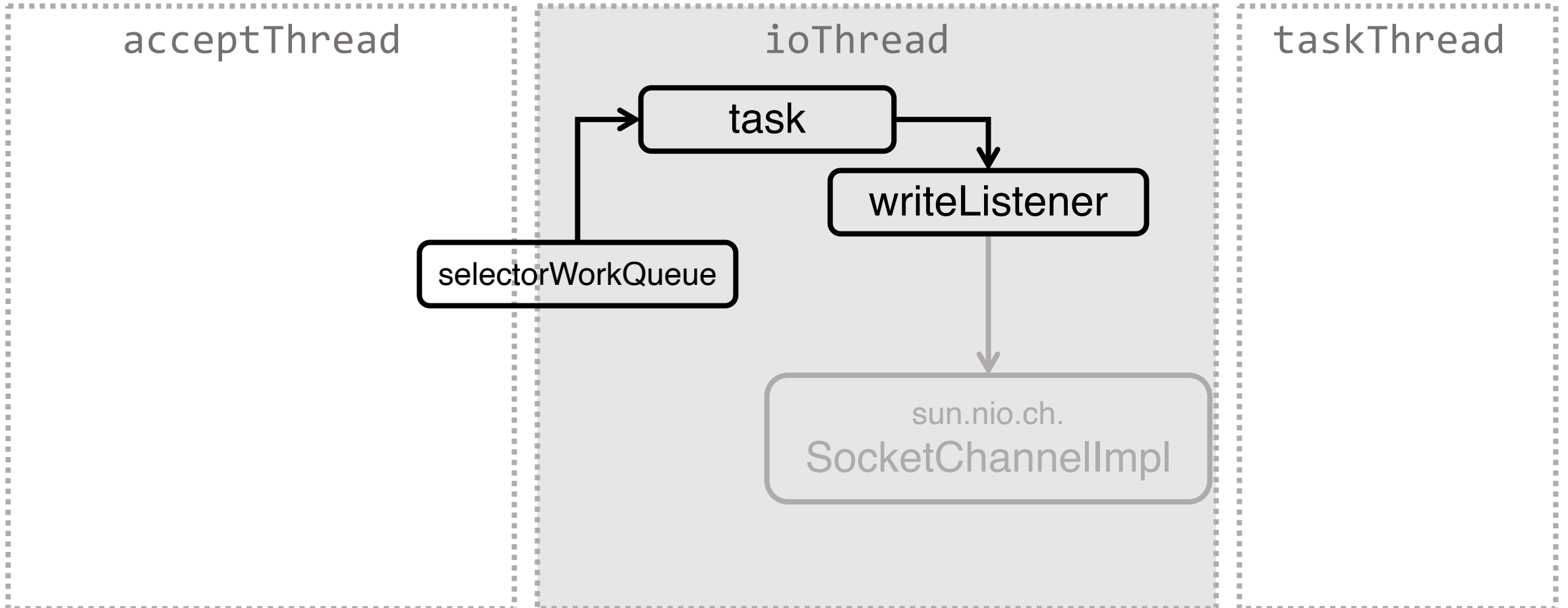
Обработка запроса



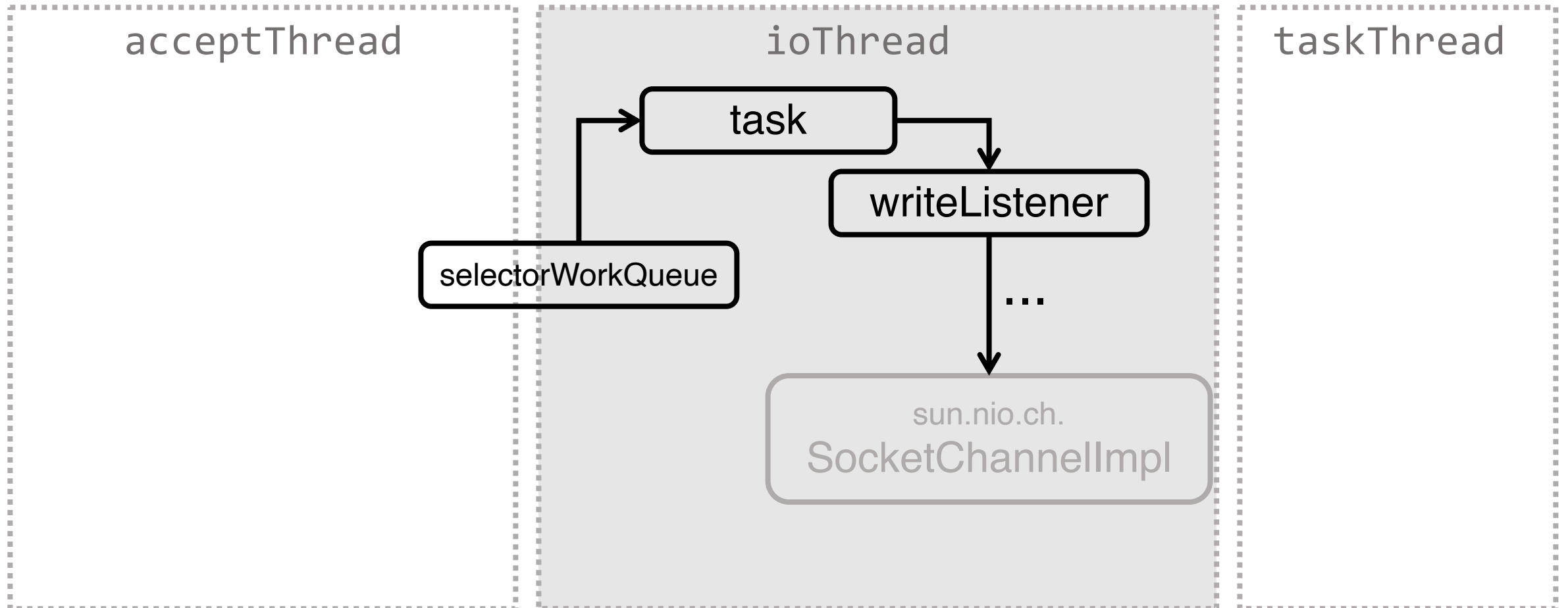
Обработка запроса



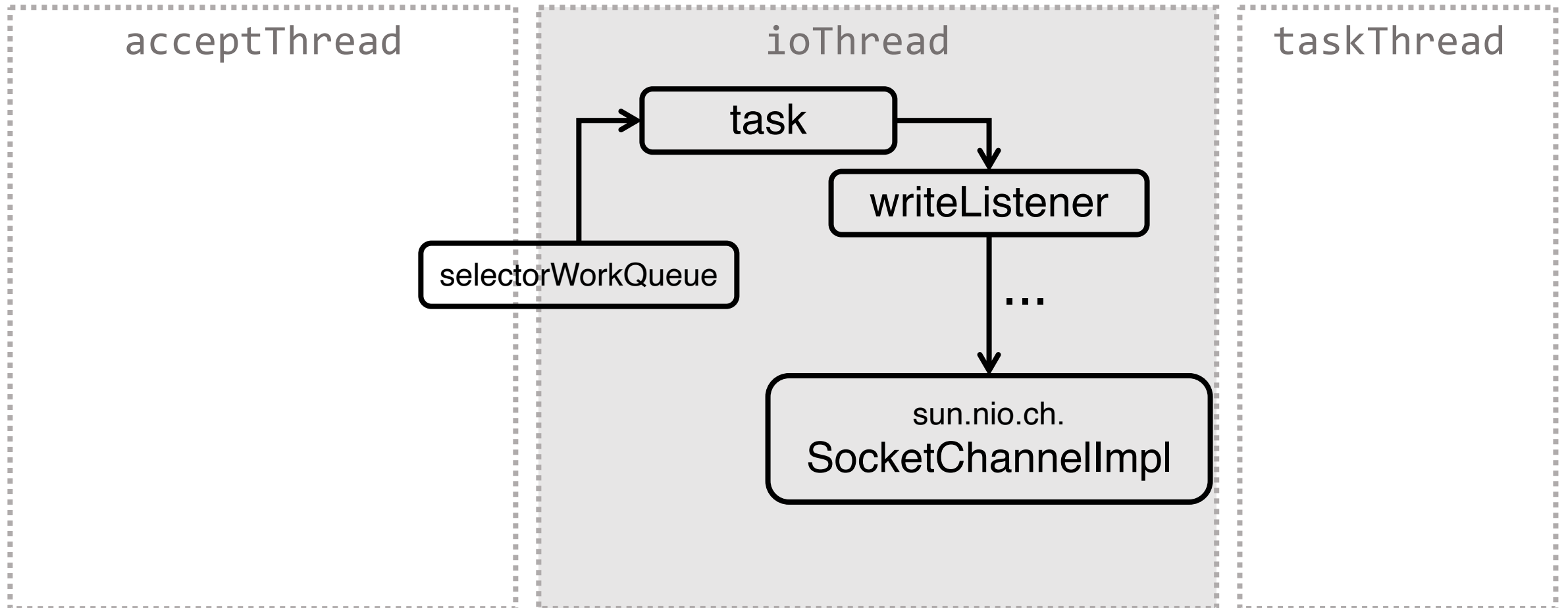
Обработка запроса



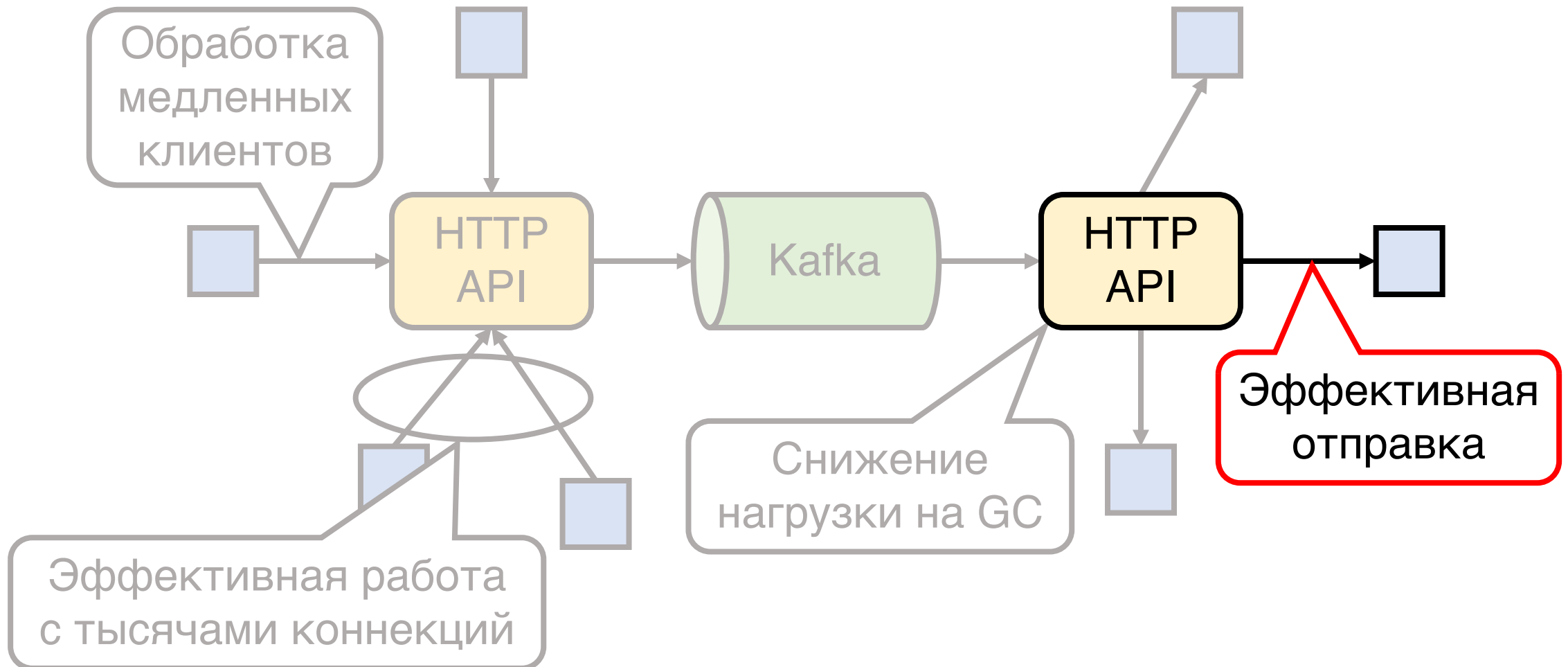
Обработка запроса



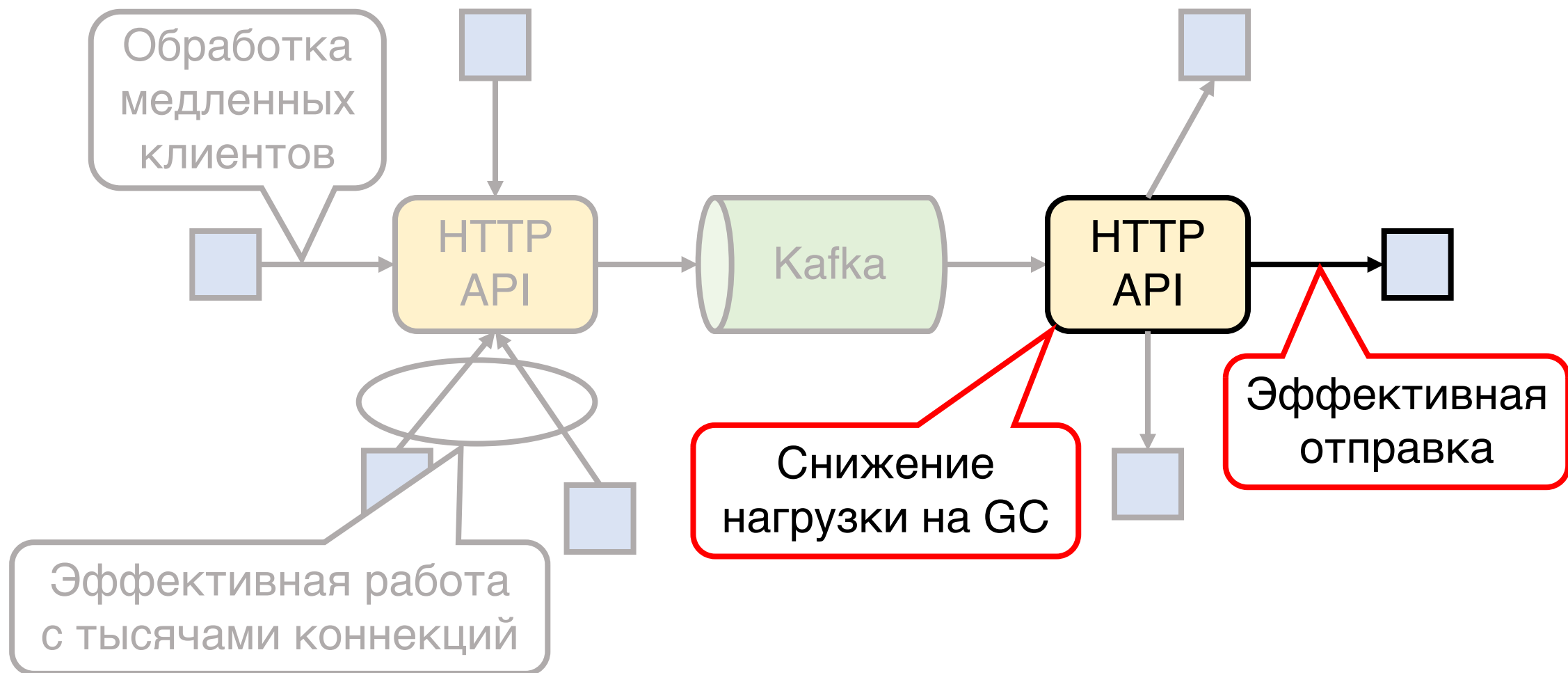
Обработка запроса



Производительность и оптимизации



Производительность и оптимизации



sun.nio.ch.SocketChannelImpl

```
public long write(  
    ByteBuffer[] srcs,  
    int offset,  
    int length) {  
    /* ... */  
    IOUtil.write(fd, srcs, offset, length, nd);  
    /* ... */  
}
```

sun.nio.ch.IOUtil

```
if (!(buf instanceof DirectBuffer)) {  
    ByteBuffer shadow = Util.getTemporaryDirectBuffer(rem);  
    shadow.put(buf);  
    shadow.flip();  
    /* ... */  
    buf = shadow;  
    /* ... */  
}
```

sun.nio.ch.IOUtil

```
if (!(buf instanceof DirectBuffer)) {  
    ByteBuffer shadow = Util.getTemporaryDirectBuffer(rem);  
    shadow.put(buf);  
    shadow.flip();  
    /* ... */  
    buf = shadow;  
    /* ... */  
}
```

sun.nio.ch.IOUtil

```
if (!(buf instanceof DirectBuffer)) {  
    ByteBuffer shadow = Util.getTemporaryDirectBuffer(rem);  
    shadow.put(buf);  
    shadow.flip();  
    /* ... */  
    buf = shadow;  
    /* ... */  
}
```

ru.kontur.vostok.hercules.util.

ByteBufferPool

ru.kontur.vostok.hercules.util.

ByteBufferPool

```
exchange.getResponseSender().send(  
    buffer,  
    new io.undertow.io.IoCallback() {  
        @Override  
        public void onComplete(  
            HttpServerExchange exchange,  
            Sender sender) {  
            ByteBufferPool.release(buffer);  
            exchange.endExchange();  
        }  
        /* ... */  
    });
```


ru.kontur.vostok.hercules.util.

ByteBufferPool

```
exchange.getResponseSender().send(  
    buffer,  
    new io.undertow.io.IOException() {  
        @Override  
        public void onComplete(  
            HttpServerExchange exchange,  
            Sender sender) {  
            ByteBufferPool.release(buffer);  
            exchange.endExchange();  
        }  
        /* ... */  
    });
```

ru.kontur.vostok.hercules.util.

ByteBufferPool

```
exchange.getResponseSender().send(
    buffer,
    new io.undertow.io.IOException() {
        @Override
        public void onComplete(
            HttpServerExchange exchange,
            Sender sender) {
            ByteBufferPool.release(buffer);
            exchange.endExchange();
        }
        /* ... */
    });
```

Производительность и оптимизации

Производительность и оптимизации

- HttpRequestParser
- HttpString

io.undertow.server.protocol.http. HttpRequestParser

```
public abstract class HttpRequestParser
```

io.undertow.server.protocol.http. HttpRequestParser

```
public abstract class HttpRequestParser
```

```
public class HttpRequestParser$$generated  
    extends HttpRequestParser
```

io.undertow.server.protocol.http. HttpRequestParser

```
770 @
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796

protected final void handleHttpVerb(ByteBuffer var1, ParseState var2, HttpServerExchange var3) throws BadRequestEx
    boolean var10;
    if (!var1.hasRemaining()) {
        var10 = false;
    } else {
        int var4;
        int var5;
        HttpString var6;
        byte[] var8;
        label156: {
            StringBuilder var7;
            label162: {
                byte var10000;
                HttpString var10003;
                label154: {
                    label163: {
                        label152: {
                            var7 = var2.stringBuilder;
                            if ((var4 = var2.parseState) != 0) {
                                var5 = var2.pos;
                                var6 = var2.current;
                                var8 = var2.currentBytes;
                                switch (var4) {
                                    case -2:
                                        break label163;
                                    case -1:
                                        break label154;
                                }
                            }
                        }
                    }
                }
            }
        }
    }
}
```

io.undertow.util.HttpString

```
/**
 * An HTTP case-insensitive Latin-1 string.
 */
public final class HttpString implements
    Comparable<HttpString>, Serializable {
    private final byte[] bytes;
    private final transient int hashCode;
    /**
     * For well known header to make comparison fast
     */
    private final int orderInt;
    private transient String string;
    /* ... */
}
```


io.undertow.util.HttpString

```
/**
 * An HTTP case-insensitive Latin-1 string.
 */
public final class HttpString implements
    Comparable<HttpString>, Serializable {
    private final byte[] bytes;
    private final transient int hashCode;
    /**
     * For well known header to make comparison fast
     */
    private final int orderInt;
    private transient String string;
    /** ... */
```

io.undertow.util.HttpString

```
/**
 * An HTTP case-insensitive Latin-1 string.
 */
public final class HttpString implements
    Comparable<HttpString>, Serializable {
    private final byte[] bytes;
    private final transient int hashCode;
    /**
     * For well known header to make comparison fast
     */
    private final int orderInt;
    private transient String string;
    /* ... */
}
```

io.undertow.util.HttpString

0x41 A

0x42 B

0x43 C

0x44 D

0x45 E

0x46 F

...

0x57 W

0x58 X

0x59 Y

0x5A Z

io.undertow.util.HttpString

0x41	A	0x61	a
0x42	B	0x62	b
0x43	C	0x63	c
0x44	D	0x64	d
0x45	E	0x65	e
0x46	F	0x66	f
...			
0x57	W	0x77	w
0x58	X	0x78	x
0x59	Y	0x79	y
0x5A	Z	0x7A	z

io.undertow.util.HttpString

0x41	A	0x61	a
0x42	B	0x62	b
0x43	C	0x63	c
0x44	D	0x64	d
0x45	E	0x65	e
0x46	F	0x66	f
...			
0x57	W	0x77	w
0x58	X	0x78	x
0x59	Y	0x79	y
0x5A	Z	0x7A	z

```
private static int higher(byte b) {  
    return b &  
        (b >= 'a' && b <= 'z' ? 0xDF : 0xFF);  
}
```

io.undertow.util.HttpString

0x41	A	0x61	a
0x42	B	0x62	b
0x43	C	0x63	c
0x44	D	0x64	d
0x45	E	0x65	e
0x46	F	0x66	f
...			
0x57	W	0x77	w
0x58	X	0x78	x
0x59	Y	0x79	y
0x5A	Z	0x7A	z

```
private static int higher(byte b) {  
    return b &  
        (b >= 'a' && b <= 'z' ? 0xDF : 0xFF);  
}
```

io.undertow.util.HttpString

```
/**
 * An HTTP case-insensitive Latin-1 string.
 */
public final class HttpString implements
    Comparable<HttpString>, Serializable {
    private final byte[] bytes;
    private final transient int hashCode;
    /**
     * For well known header to make comparison fast
     */
    private final int orderInt;
    private transient String string;
    /* ... */
}
```

io.undertow.util.HttpString

```
/**
 * An HTTP case-insensitive Latin-1 string.
 */
public final class HttpString implements
    Comparable<HttpString>, Serializable {
    private final byte[] bytes;
    private final transient int hashCode;
    /**
     * For well known header to make comparison fast
     */
    private final int orderInt;
    private transient String string;
    /* ... */
}
```


io.undertow.util.HttpString

```
public static final HttpString ACCEPT
    = new HttpString(ACCEPT_STRING, 1);

public static final HttpString ACCEPT_CHARSET
    = new HttpString(ACCEPT_CHARSET_STRING, 2);

public static final HttpString ACCEPT_ENCODING
    = new HttpString(ACCEPT_ENCODING_STRING, 3);
```

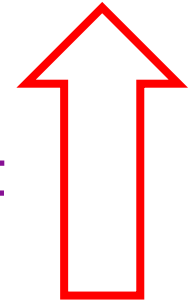
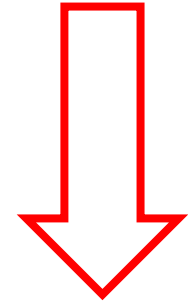
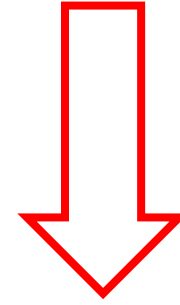
io.undertow.util.HttpString

```
public static final HttpString ACCEPT  
    = new HttpString(ACCEPT_STRING, 1);
```

```
public static final HttpString ACCEPT_CHARSET  
    = new HttpString(ACCEPT_CHARSET_STRING, 2);
```

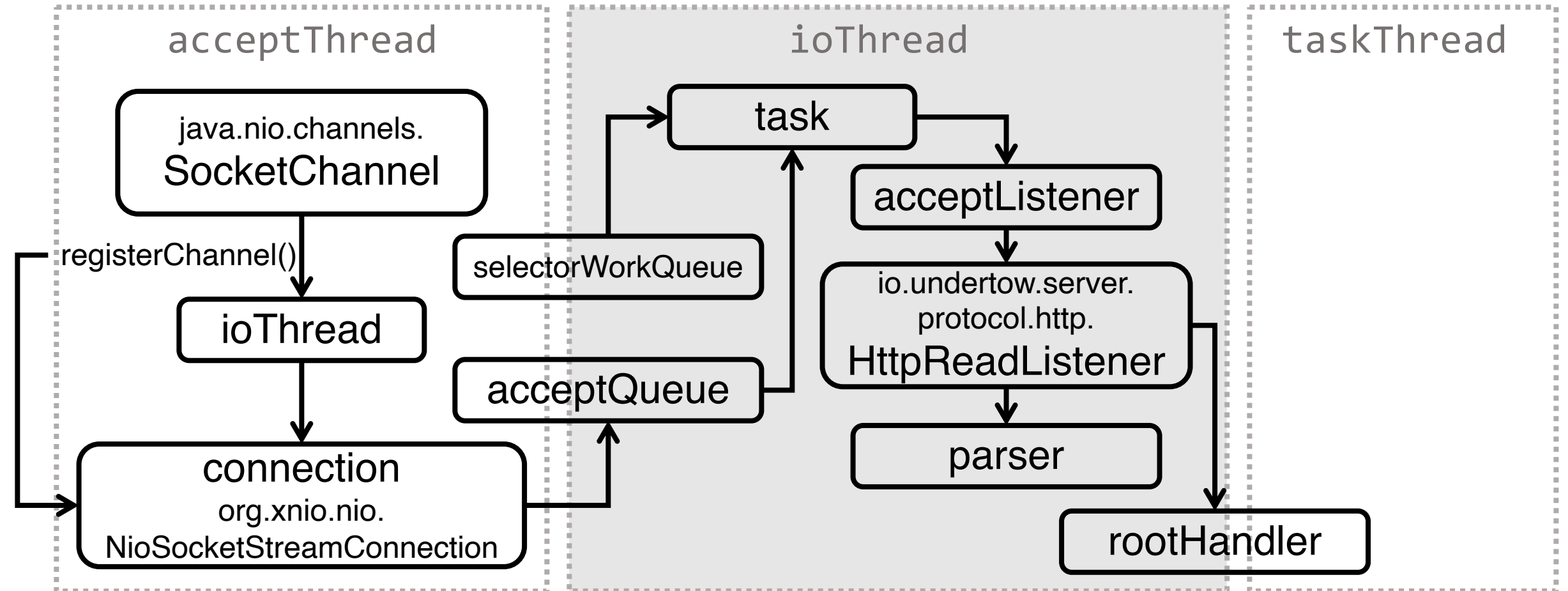
```
public static final HttpString ACCEPT_ENCODING  
    = new HttpString(ACCEPT_ENCODING_STRING, 3);
```

HttpString.orderInt

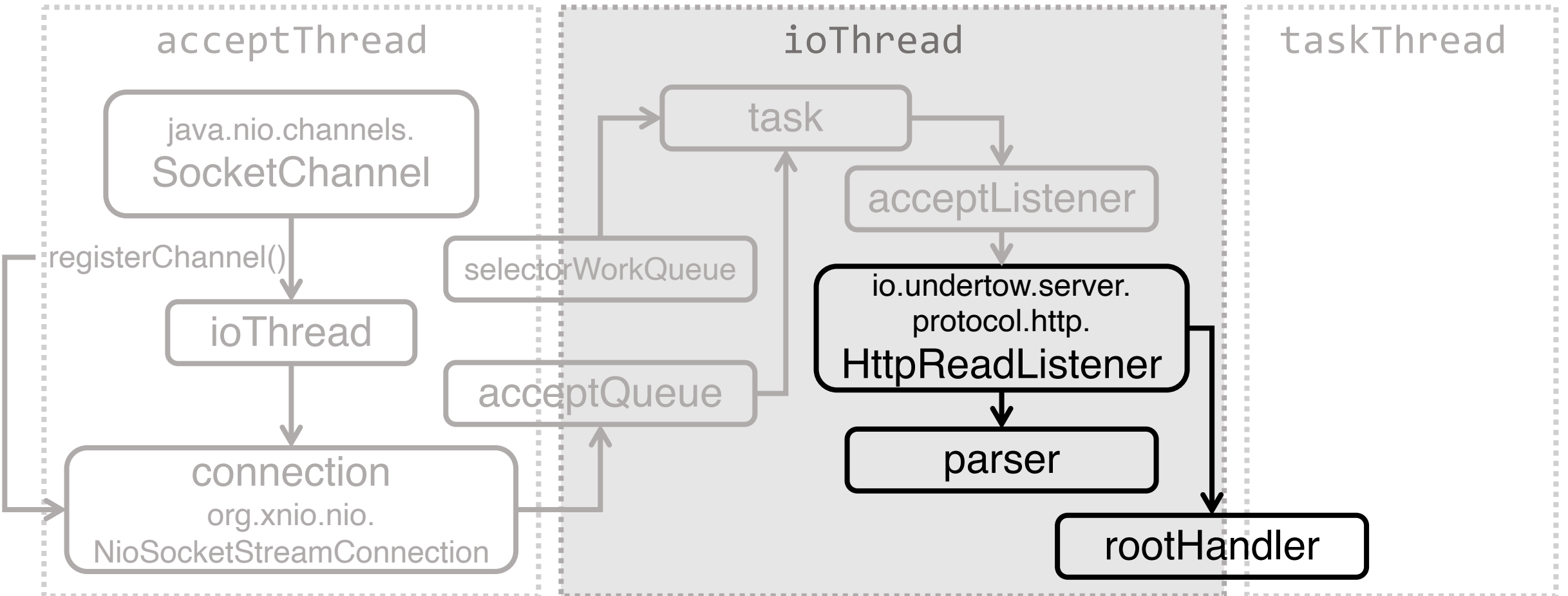


Коварный трейдинг

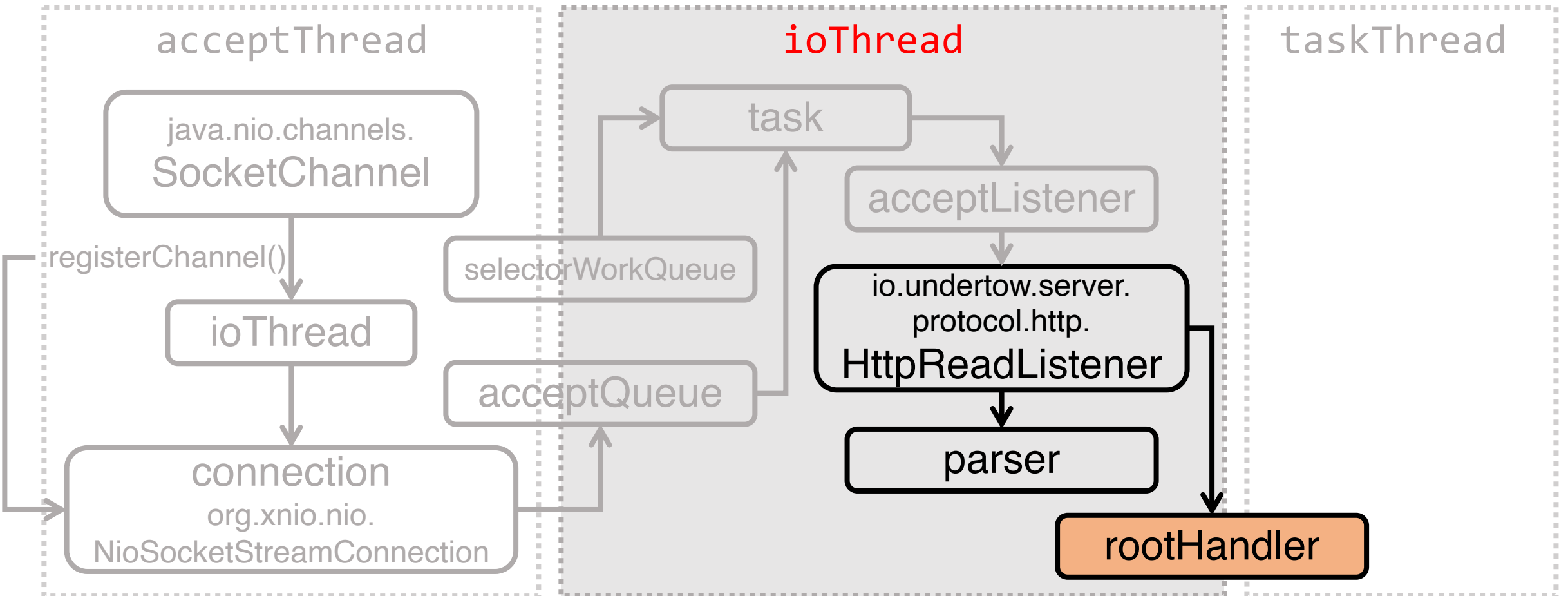
Коварный трединг



Коварный трединг

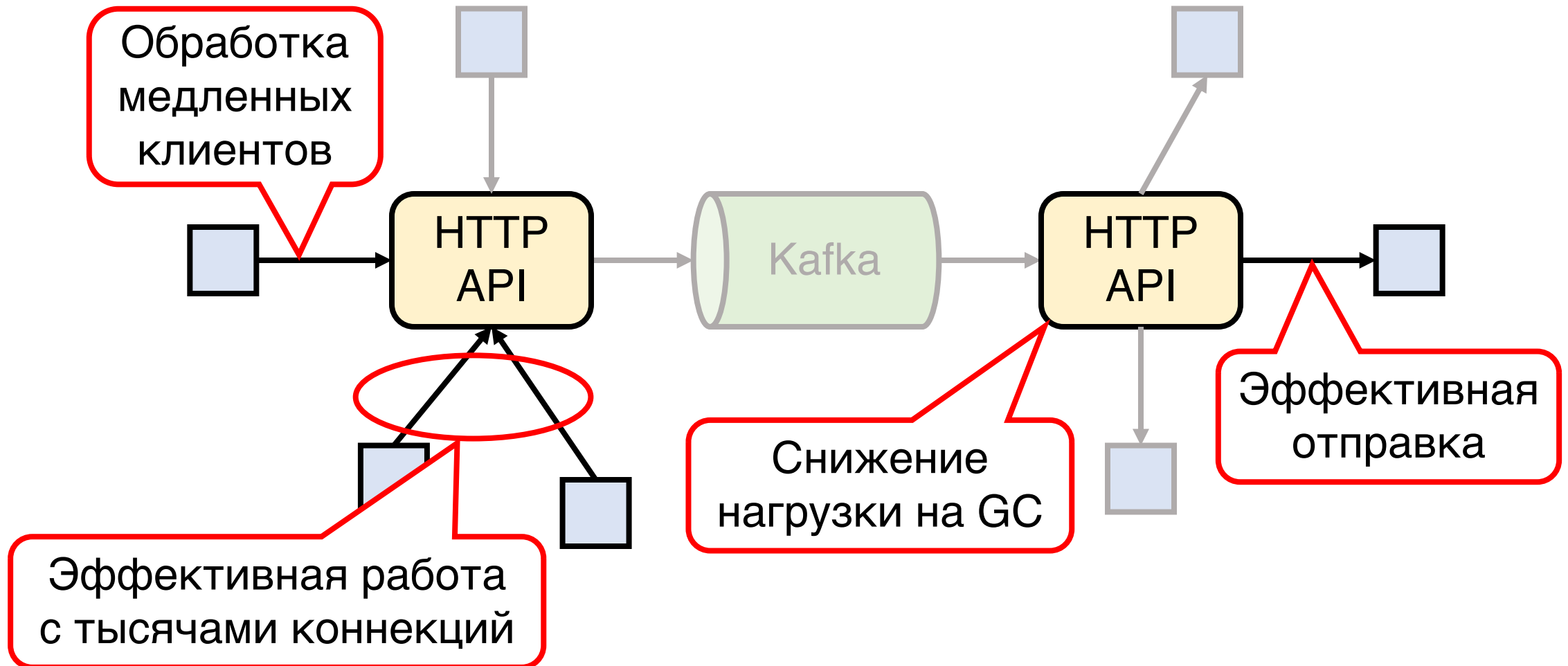


Коварный трединг



Выводы

Выводы



Q/A

Другие доклады и материалы:

https://tg.me/chnl_GregoryKoshelev

