FFFast Drupal backend

No kidding!

Pavel Prischepa

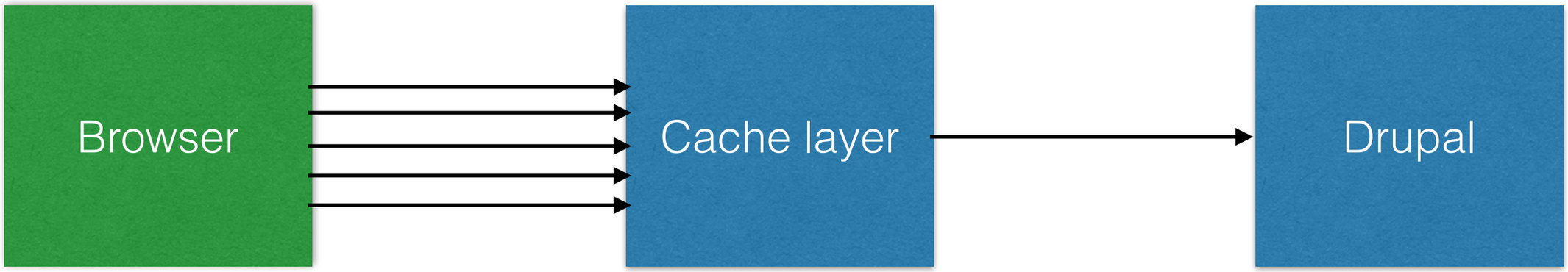
Pavel Prischepa

Partner at i20 Group
CEO at DrupalJedi

5 years of work with Drupal (and Drupal only) on an international level

- Development
- Project management
- Drupal PR, promotion
- Drupal audit
- Consulting

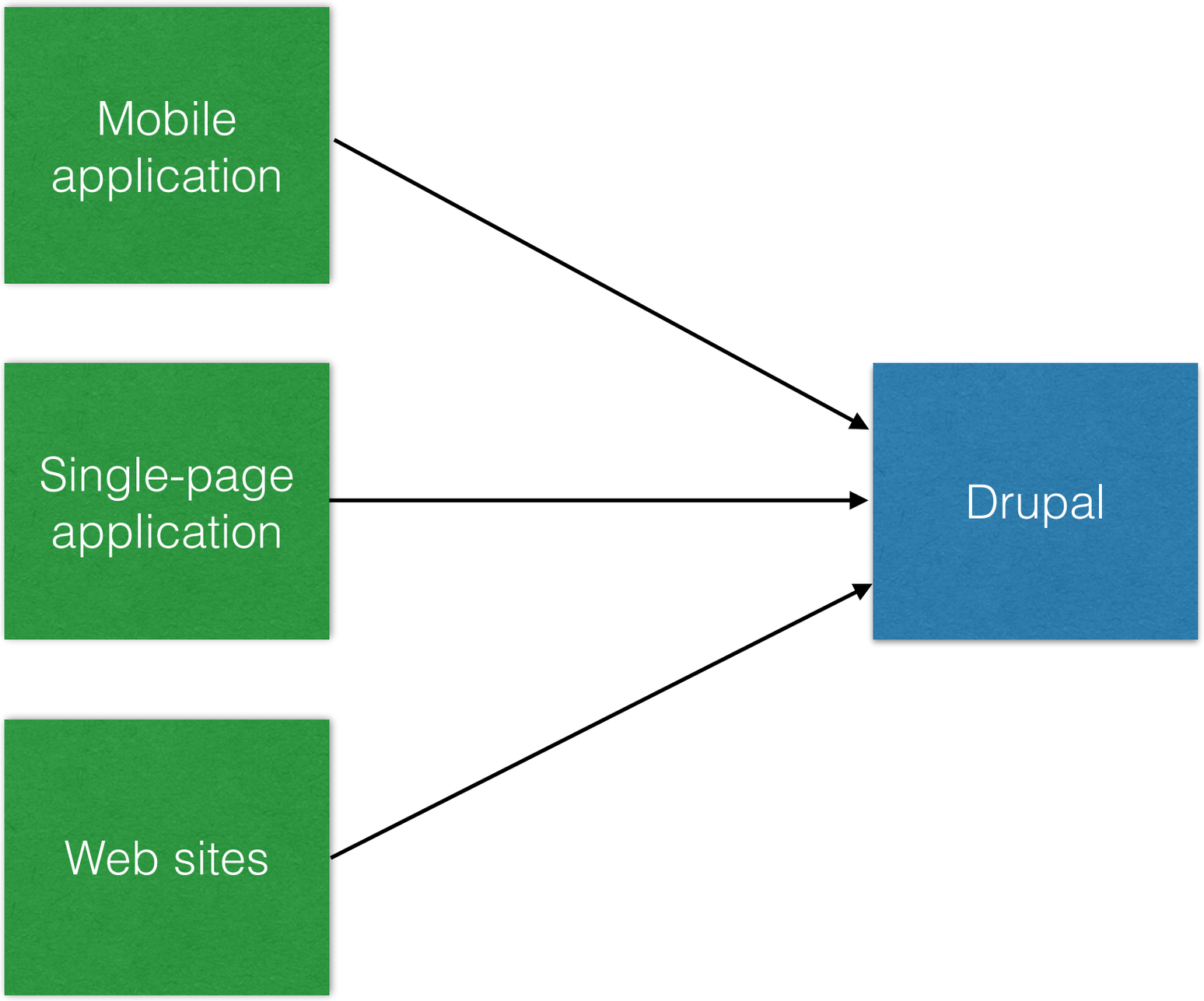
Is Drupal fast?



Real project

- 5 000 000 nodes
- 1 000 000 unique visitors per day
- 0,4 seconds per page
- Cache: MongoDB, Redis, NginX, Varnish
- Dynamic blocks: ESI, AJAX

Drupal as backend



Drupal as backend

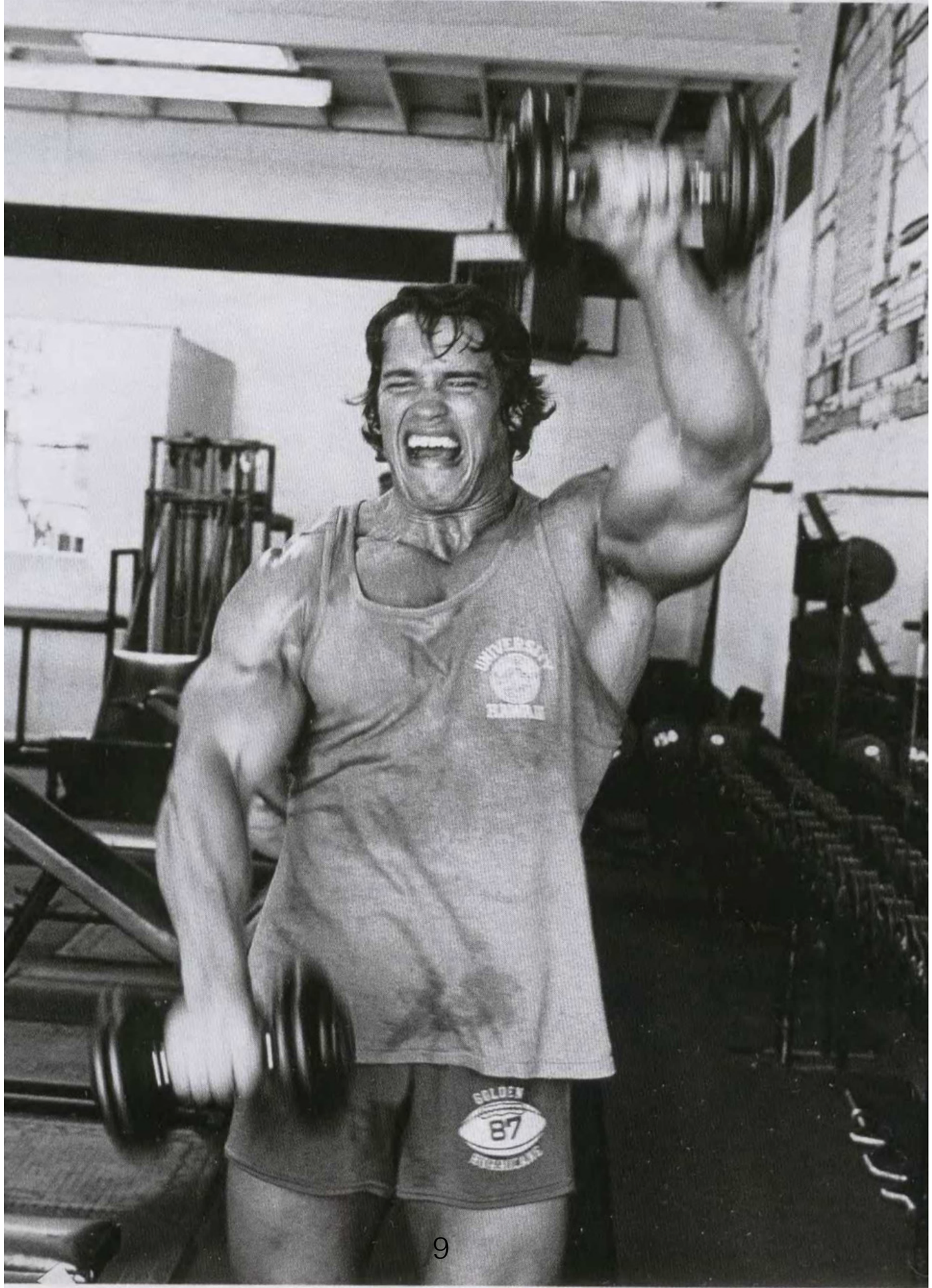
- DRUPAL_BOOTSTRAP_FULL for each request
- 30-100 database queries per request
- Slow server response

Silk Paints:

mobile application for drawing

Silk Paints

- Users draw paintings, save them to server
- Backend on Drupal (Services)
- 500 000 users
- **5-20** requests per second


UNIVERSAL
GOLDEN
87

Silk Paints:
launch of iOS version

- Likes
- Friend lists
- Pictures moderation
- **50-100** requests per second

Looking for an alternative to Drupal

- Node.js
- Python
- Zend

One does not simply use a new framework

- Change production process
- Train / hire people
- “Nabit’ shishki”
- Meanwhile...project is uncontrollable

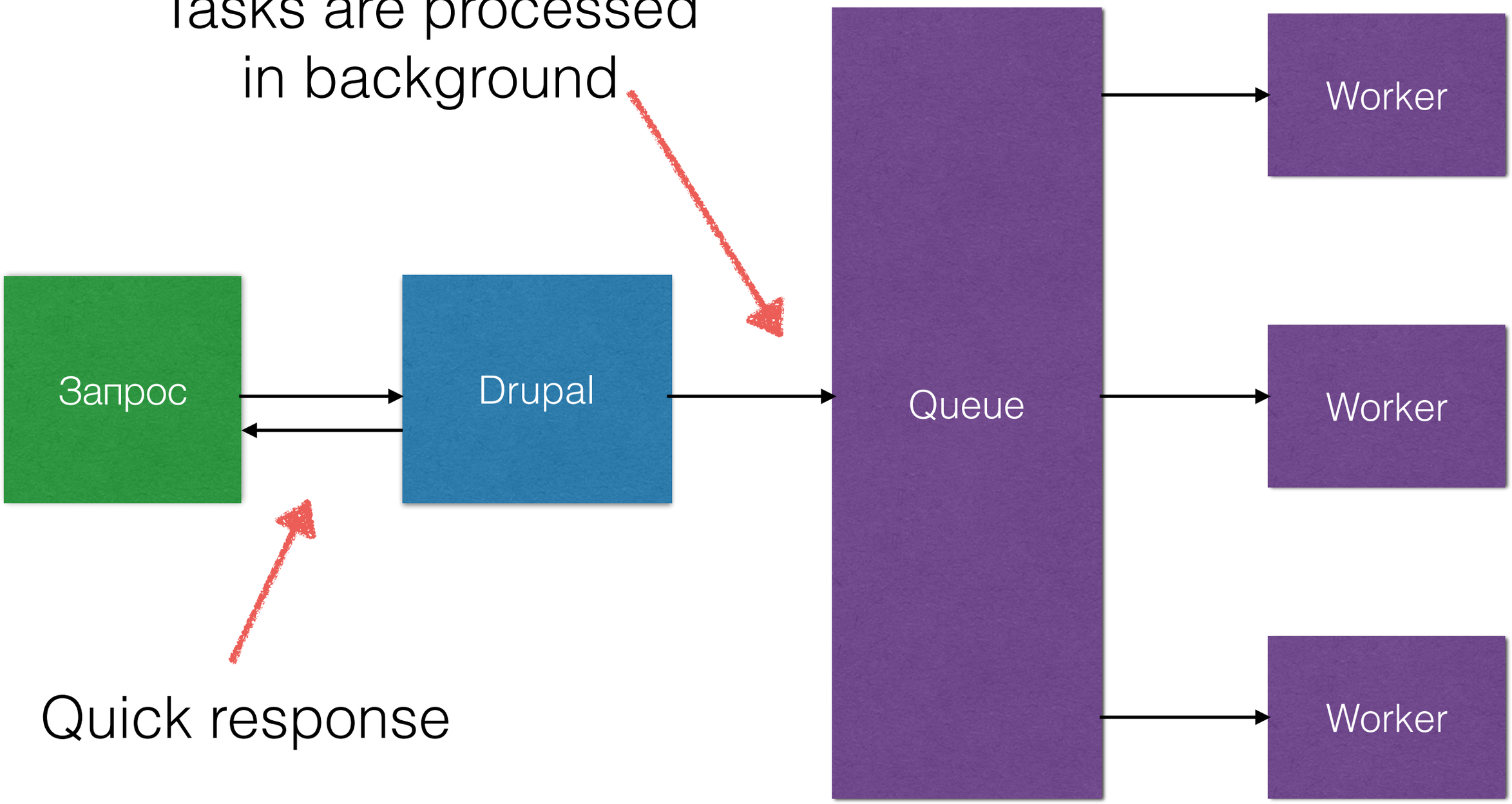
Back to Drupal

Back to Drupal

- Coding standards
- Config
- DB integration
- Cache
- Modules
- Design, Develop, Testing, Deploy processes

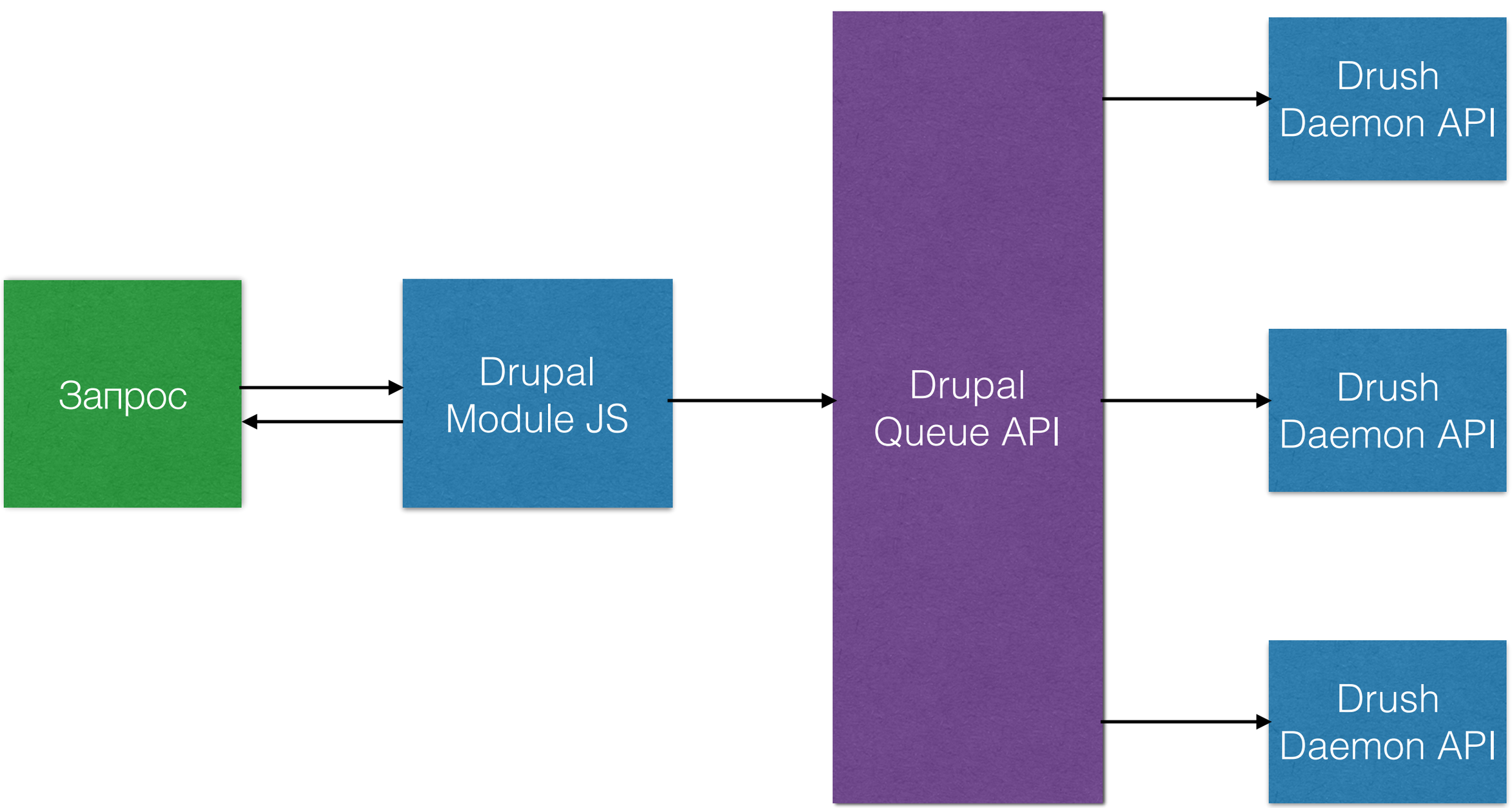
Back to Drupal

Tasks are processed
in background



Quick response

Back to Drupal



Silk Paints: result

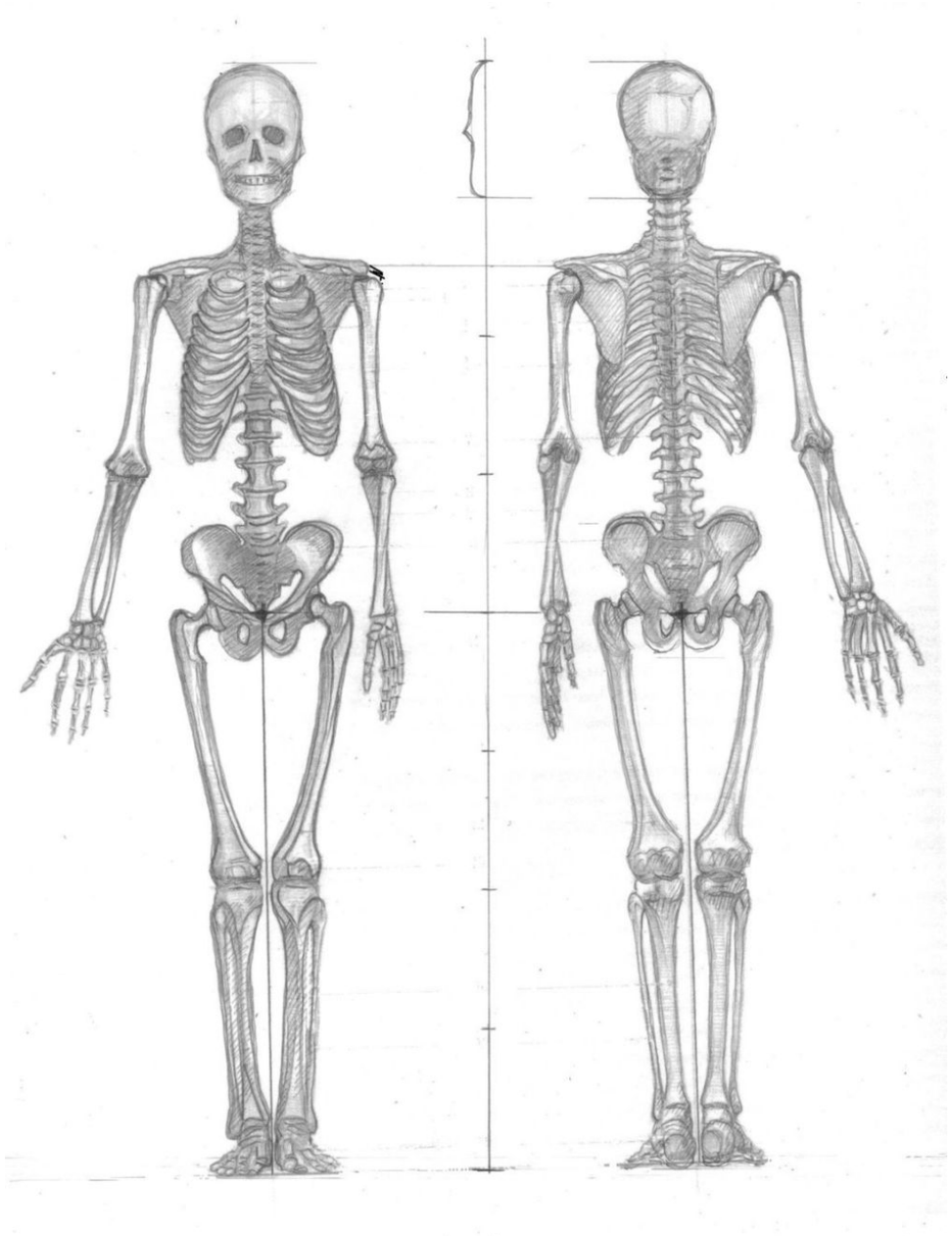
- 50 requests per second
- 5-10 requests to MySQL per request
- Response time less than 1 second
- Savings on development and support
- Predictability of development

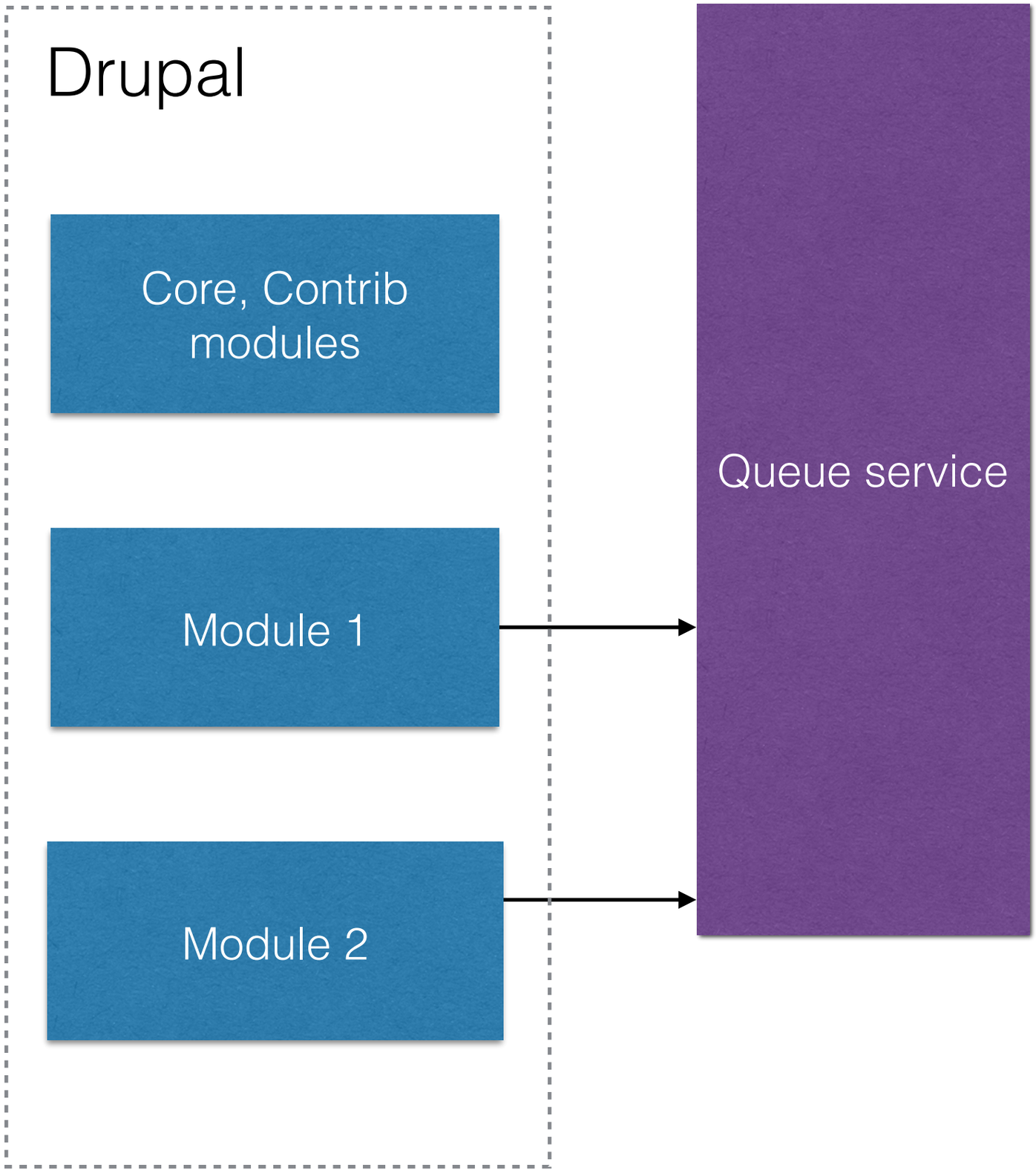
Drupal as backend

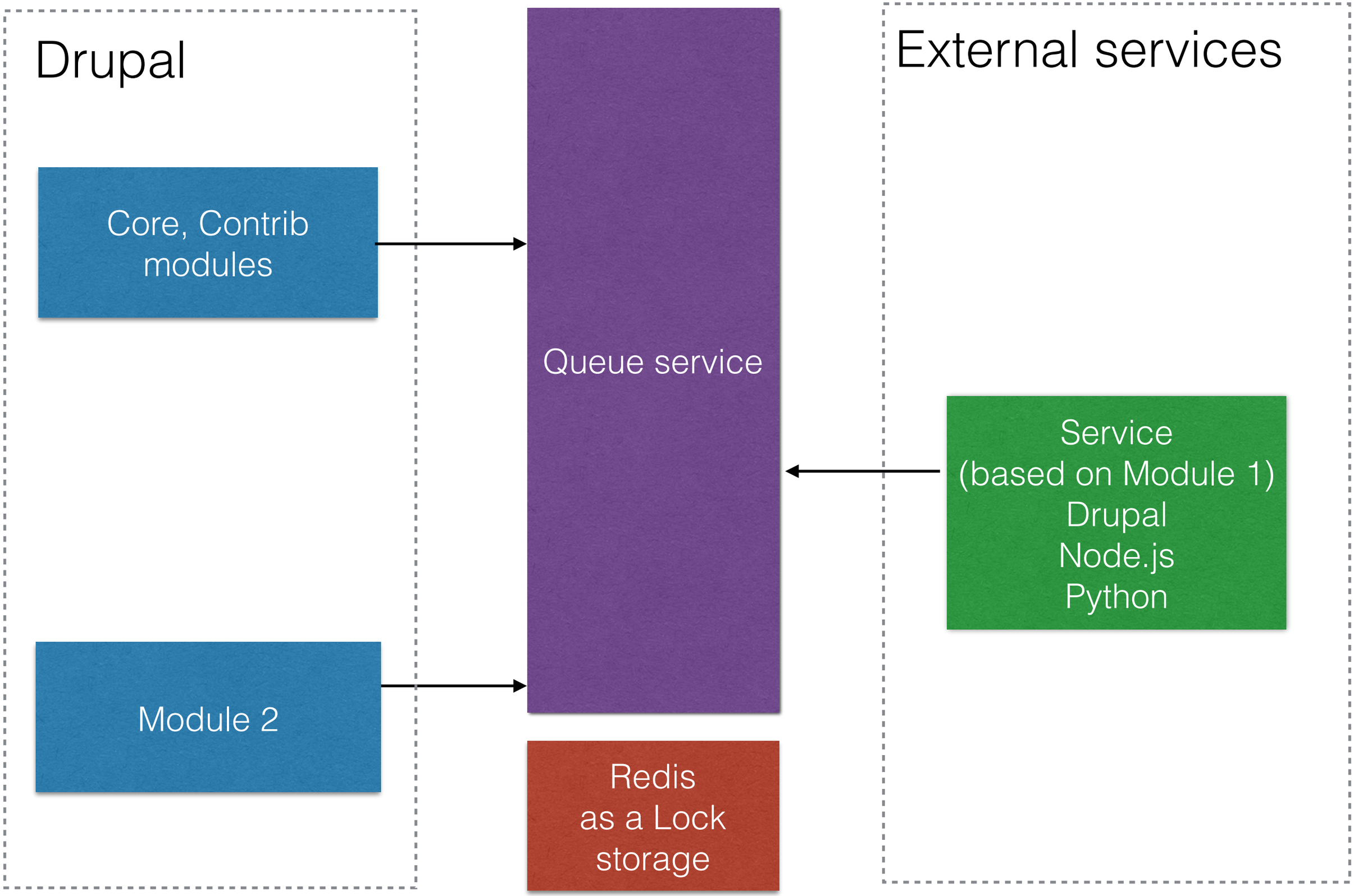
Best practice

Application architecture

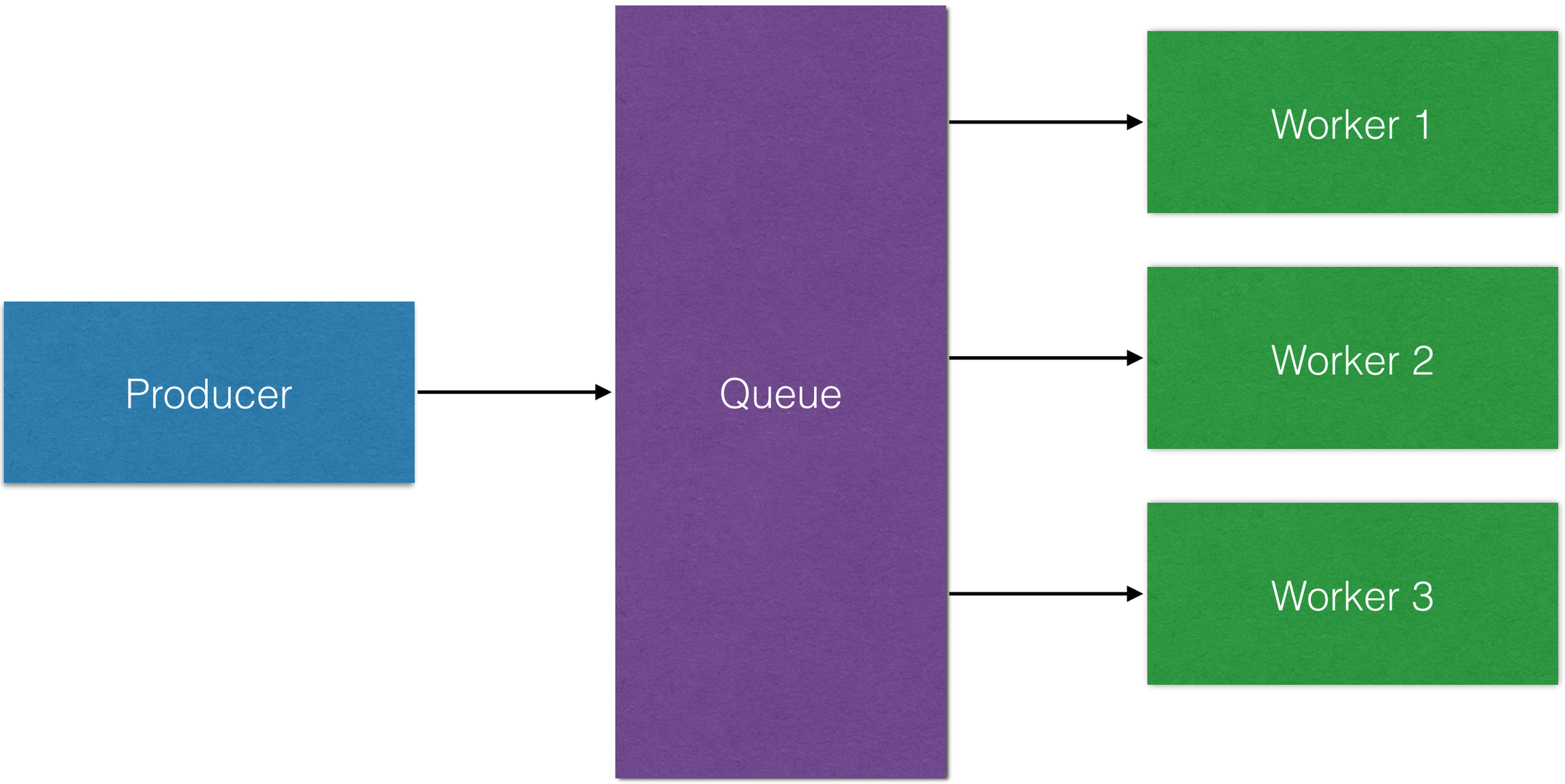
- Service Oriented Architecture (SOA)
- Queue service (middleware)



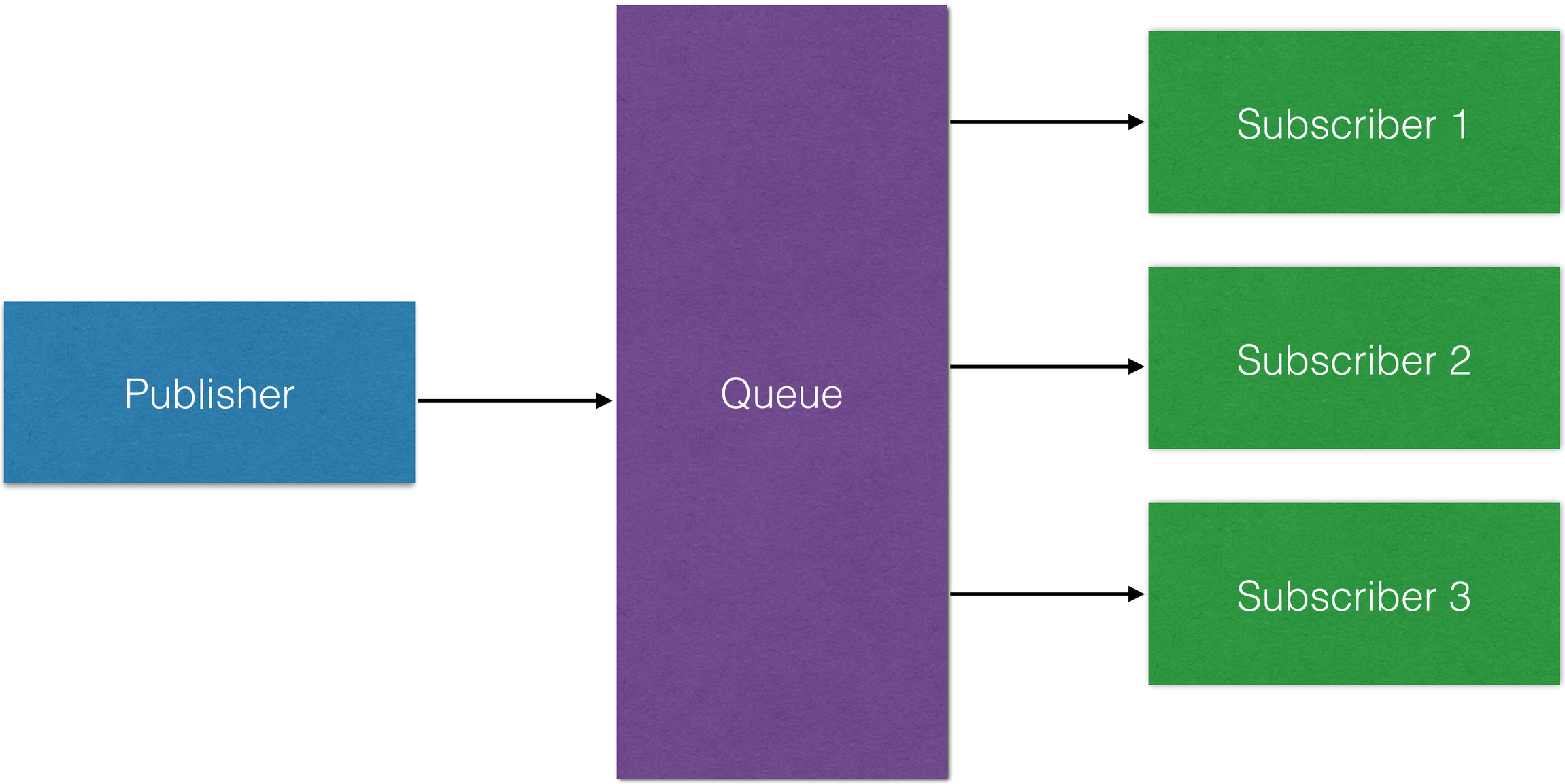

Drupal
Core, Contrib modules
Module 1
Module 2
Queue service


Drupal
Core, Contrib modules
Module 2
Queue service
Redis as a Lock storage
External services
Service (based on Module 1)
Drupal
Node.js
Python

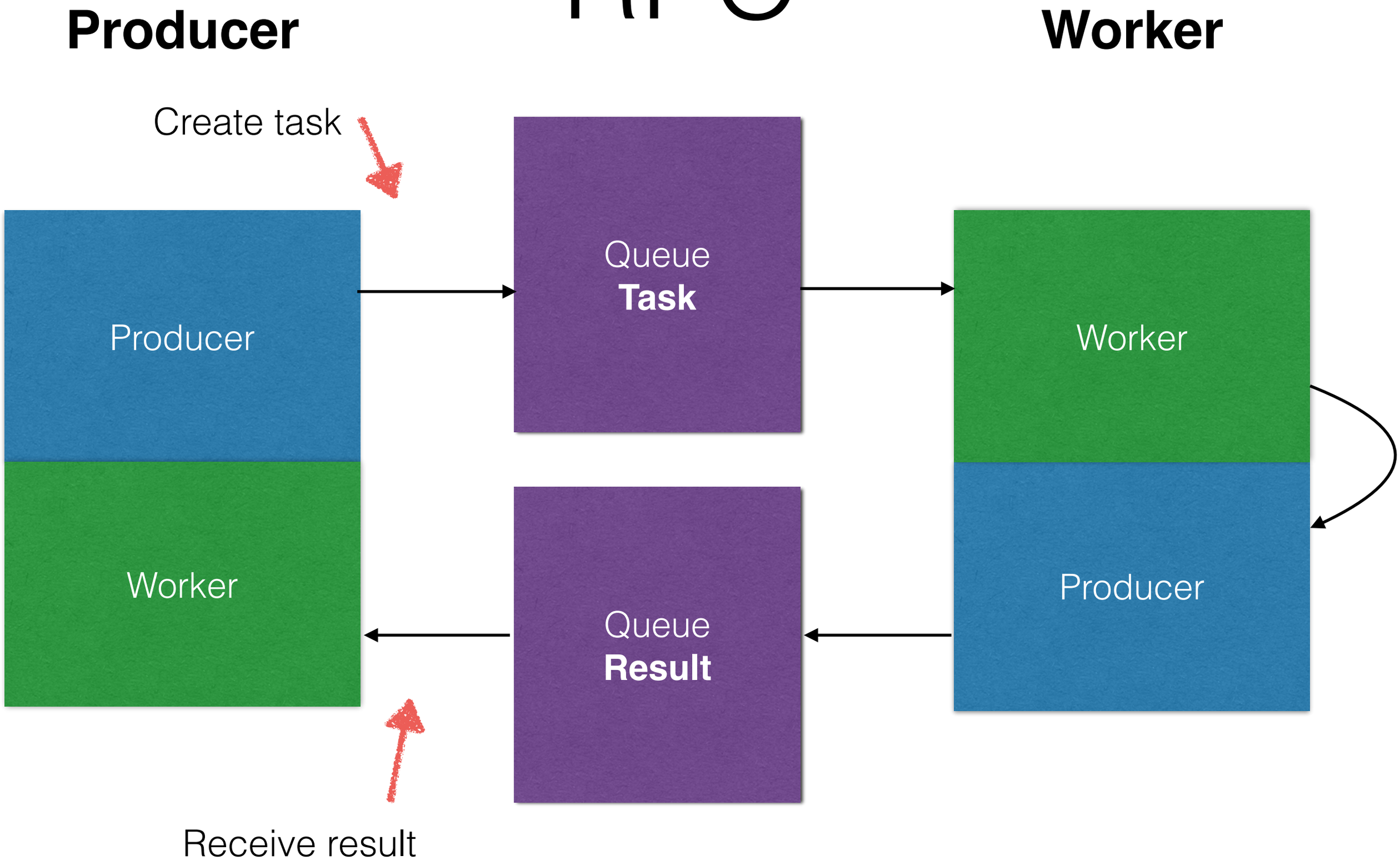
Work queues (worker per task)



Publish / subscribe
(one message for all subscribers)



RPC


Producer
Worker
Create task
Producer
Worker
Queue
Task
Queue
Result
Worker
Producer
Receive result

Best practice

What else?

Degradation of functionality

- If load increases, non-critical functionality could be disabled
- Several cases for disabling
- Approach allows pass the peak load successfully

Cache

- Application should work perfectly without cache
- Cache invalidation scheme
- “Cold cache” problem
- Efficient of cache using (hit/miss correlation)

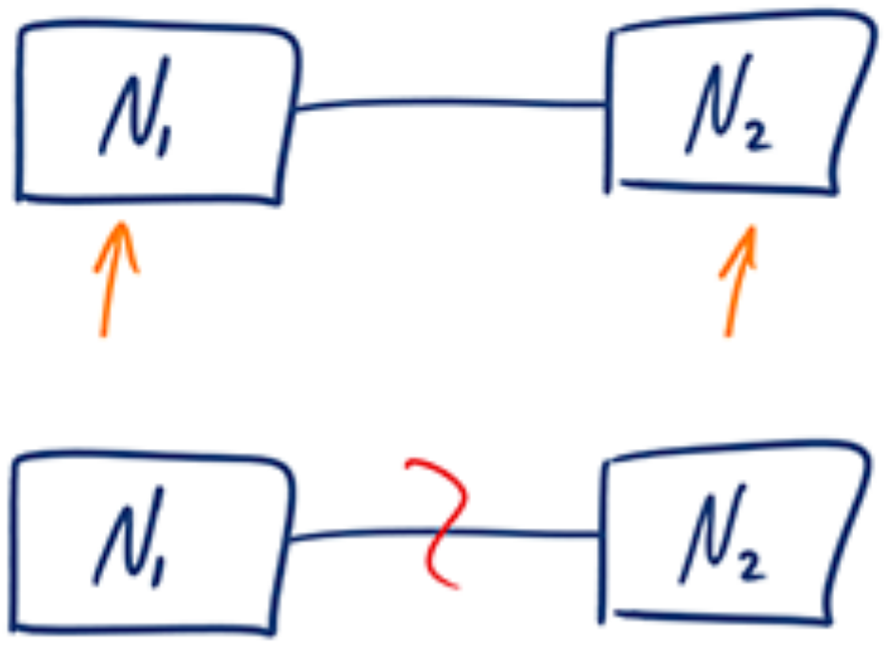
Database

- Take care of database
- Denormalize it!
- Check indexes
- Do not add other databases beforehand (like MongoDB)
- Migrate to Postgres? (…, NoSQL, memory storage)

Database
replica, partition, sharding

CAP Theorem:

Consistency
Availability
Partition tolerance



The most important

- The main causes of low performance - “krivie ruki”
- Use technologies that you know well
- Remember about business
- Optimize only what prevents sells

Thank you!

Pavel Prischepa

CEO at DrupalJedi
pprischepa@drupaljedi.com