

Dead Letters

Handling within Message Broker



Sergej Kurakin



Sergej Kurakin

Age: 36

Company: Oberlo

Position: Senior Developer

<http://careers.oberlo.com/>



Dead Letters

1. Message Queues
2. Poisonpill Message
3. Dead Letter Queue
4. Alternatives
5. What to do with Dead Letters?
6. Demos

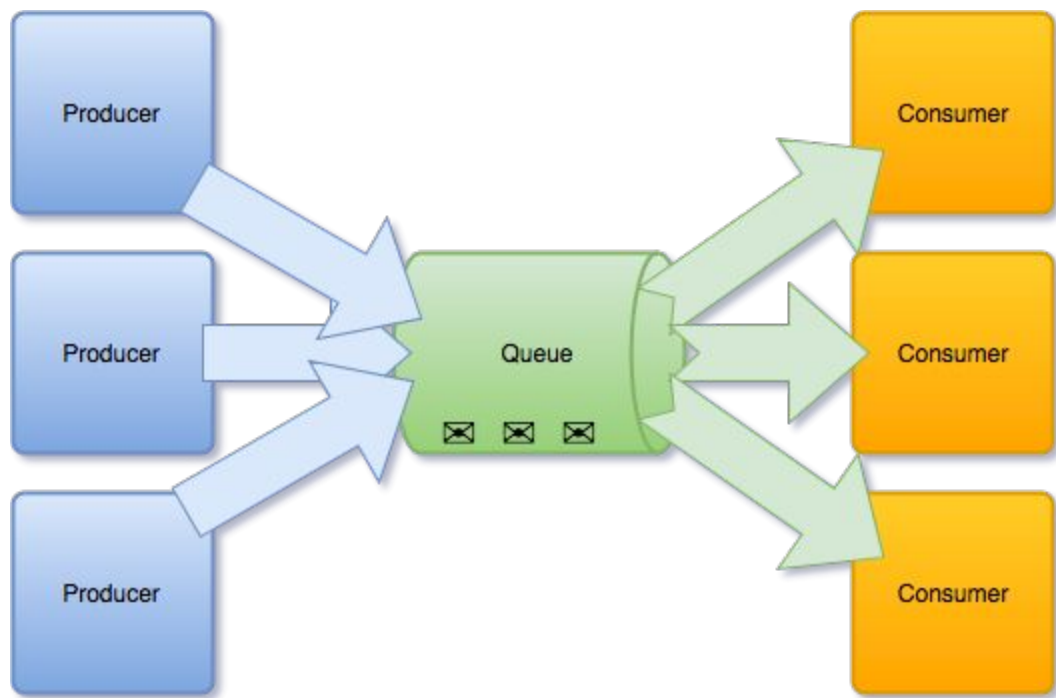
Message Queues



In computer science, message queues and mailboxes are software-engineering components used for inter-process communication, or for inter-thread communication within the same process.

Simple Explanation

- **Client (Sender, Producer)** - sends message to Message Server.
- **Message Server (Broker)** - Receives message from Client(s) and routes to proper Worker(s).
- **Worker (Receiver, Consumer)** - receives message and executes task.



Common Usage for WEB

- Background Tasks
- Parallel Tasks
- System Integration
- Interlanguage Communication
- Scaling
- Decoupling
- Traffic Shaping

Message Queue Software

- Amazon Simple Queue Service
- Apache ActiveMQ
- RabbitMQ
- Gearman
- Beanstalk
- IBM Integration Bus (WebSphere Message Broker)
- Microsoft Message Queuing
- Redis*
- RDBMS*
- Use “Search” for more ;-)

Poisonpill Message



Message that can't be processed.

Message that can bring down all your
Workers.

```
PHP Fatal error: Uncaught RuntimeException: Fail command received in
/home/skurakin/queuenodl/default/consumer.php:27
Stack trace:
#0 [internal function]: process_message(Object(PhpAmqpLib\Message\AMQPMessage))
#1
/home/skurakin/queuenodl/vendor/php-amqplib/php-amqplib/PhpAmqpLib/Channel/AMQPChannel.php(
986): call_user_func('process_message', Object(PhpAmqpLib\Message\AMQPMessage))
#2 [internal function]:
PhpAmqpLib\Channel\AMQPChannel->basic_deliver(Object(PhpAmqpLib\Wire\AMQPReader),
Object(PhpAmqpLib\Message\AMQPMessage))
#3
/home/skurakin/queuenodl/vendor/php-amqplib/php-amqplib/PhpAmqpLib/Channel/AbstractChannel.
php(202): call_user_func(Array, Object(PhpAmqpLib\Wire\AMQPReader),
Object(PhpAmqpLib\Message\AMQPMessage))
#4
/home/skurakin/queuenodl/vendor/php-amqplib/php-amqplib/PhpAmqpLib/Channel/AbstractChannel.
php(340): PhpAmqpLib\Channel\AbstractChannel->dispatch('60,60',
'\x07default\x00\x00\x00\x00\x00\x00\x00...', Object(PhpAmqpLib\Message\AMQPMessage))
#5 /home/skurakin/queuenodl/default/consumer.php(50): PhpAmqpLib in
/home/skurakin/queuenodl/default/consumer.php on line 27
```

Dead Letter Queue

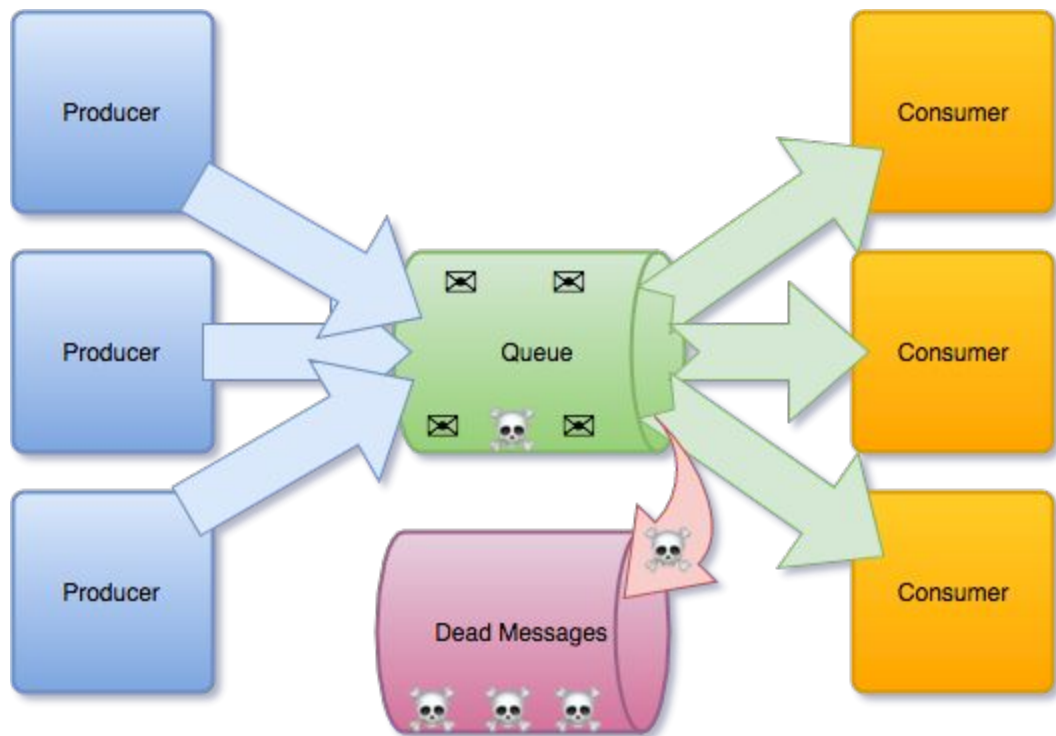


Queue where “bad” messages appear.

“Bad” messages

- The message is rejected (failed to process)
- The TTL for the message expires
- The queue length limit is exceeded
- Undeliverable message
- Other rules may apply

Other rules may apply - consult with your message queue manual!



Who supports Dead Letter Queues

- Amazon Simple Queue Service
- Apache ActiveMQ
- RabbitMQ
- IBM Integration Bus (WebSphere Message Broker)
- Microsoft Message Queuing
- Use “Search” for more ;-)

Alternatives



Alternatives to Dead Letter Queues

- Ignore bad messages
- Track statistics
- Log exception and store message payload
- Implement Dead Letter Queue manually

What to do with Dead Letters?



You decide!

- Ignore and clear Dead Letters Queue
- Analyze, fix your code and clear Dead Letters Queue
- Analyze, fix your code and requeue Dead Letters

Demos

```
skurakin@queuetest:~/queenodl/default$ php -f producer.php test
skurakin@queuetest:~/queenodl/default$ php -f producer.php test
skurakin@queuetest:~/queenodl/default$ php -f producer.php test
skurakin@queuetest:~/queenodl/default$ php -f producer.php test
skurakin@queuetest:~/queenodl/default$ php -f producer.php test
skurakin@queuetest:~/queenodl/default$ php -f producer.php fail
skurakin@queuetest:~/queenodl/default$ pwd
/home/skurakin/queenodl/default
skurakin@queuetest:~/queenodl/default$

AbstractChannel.php(340): PhpAmqpLib\Channel\AbstractChannel->dispatch('60,60', '\x07default\x00\x00\x00\x00\x00\x00\x00...', Object(PhpAmqpLib\Message\AMQPMessage))
#5 /home/skurakin/queenodl/default/consumer.php(50): PhpAmqpLib in /home/skurakin/queenodl/default/consumer.php on line 27
skurakin@queuetest:~/queenodl/default$

mqplib/php-amqplib/PhpAmqpLib/Channel/AbstractChannel.php(340): PhpAmqpLib\Channel\AbstractChannel->dispatch('60,60', '\x07default\x00\x00\x00\x00\x00\x00\x00...', Object(PhpAmqpLib\Message\AMQPMessage))
#5 /home/skurakin/queenodl/default/consumer.php(50): PhpAmqpLib in /home/skurakin/queenodl/default/consumer.php on line 27
skurakin@queuetest:~/queenodl/default$
```

[0] 0: bash* "queuetest" 18:03 27-Dec-17

All demos are based on RabbitMQ 3.7
and “php-amqplib”

No error checking

On ordinary queue

All consumers fail.

Message stays in queue.

System non-operational.

Reject, no requeue

On ordinary queue

Consumers recover from fail.

Message is lost, unless you handle it.

System operational.

Reject, requeue

On ordinary queue

Consumers recover from fail.

Message stays in same the queue first and gets rejected from all failing consumers.

System semi-operational or non-operational.

Reject, no requeue

On queue with Dead Letter
Exchange configured

Consumers recover from fail.

Message is moved into Dead Letter
Exchange.

System operational.

Dead Letters - Recap

1. Message Queues
2. Poisonpill Message
3. Dead Letter Queue
4. Alternatives
5. What to do with Dead Letters?
6. Demos

Links

https://en.wikipedia.org/wiki/Message_queue

<https://www.cloudamqp.com/blog/2014-12-03-what-is-message-queuing.html>

https://en.wikipedia.org/wiki/Dead_letter_queue

<https://www.rabbitmq.com/dlx.html>

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-dead-letter-queues.html>

Discussion

Sergej Kurakin

Work @mail: sergej@oberlo.com

Personal @mail: sergej@kurakin.info

<https://www.linkedin.com/in/sergejkurakin>

