Welcome to

10. Transactions and idempotency

KEA System Integration F2020 10 ECTS

Henrik Kramselund Jereminsen hkj@zencurity.com @kramse

Slides are available as PDF, kramse@Github

10-transactions-idempotency-system-integration.tex in the repo security-courses

This weeks Agenda in system integration

- Follow the plan:
  `https://zencurity.gitbook.io/kea-it-sikkerhed/system-integration/lektionsplan`
- Work on the hand-in assignment I: Describe the system environment for an organisation
- Plan for May 4.
  I will go through the subjects from the book
-

Goals for today

Todays goals:

- Talk about selecting technologies
- Camel books chapters 12-13
- Talk about the hand-inn assignment

Photo by Thomas Galler on Unsplash

Time schedule

- 08:30 2x 45 min with 10min break
  Documentation in Enterprises
  Open discussion how to select technologies
- 10:15 2x 45 min with 10min break
  Camel ch 12-13
  Chapter 12: Transactions and idempotency
  Chapter 13: Parallel processing
- 12:30 2x 45min with 10min break
  Hand-inn assignment hints and walk-through
- 14:15 45 min
  Chatting, doing exercises, questions about Linux

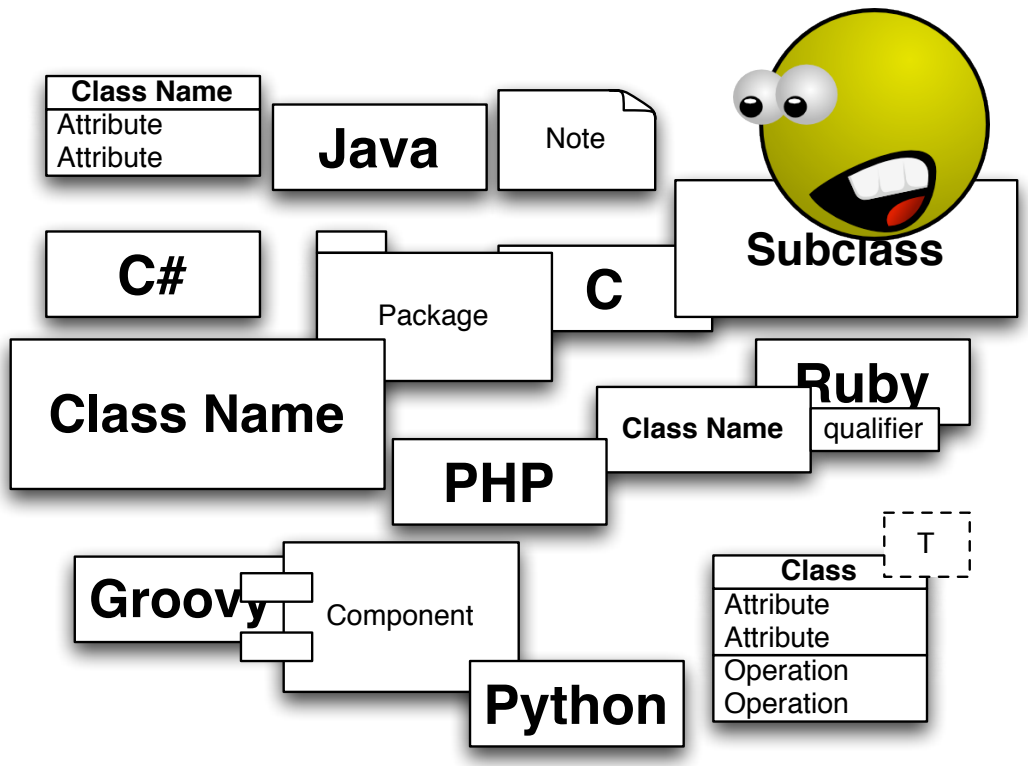
Plan for today

- How to select technologies in an organisation
- Documentation in Enterprises
- Transactions and idempotency
- Parallel processing with Camel
- Hand-inn assignment hints and walk-through

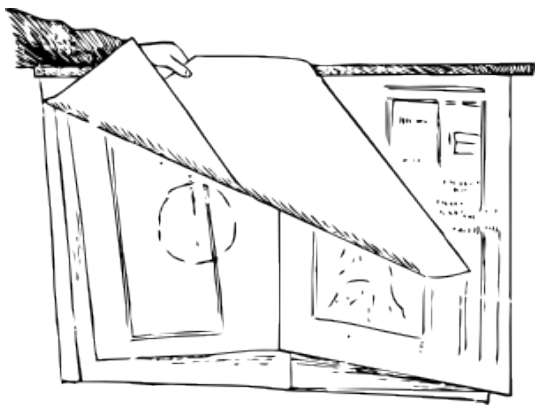
Exercises

- Open Discussion about selecting technologies
- Camel book and Java stuff

Selecting Technologies for your enterprise



Why talk about selecting technologies

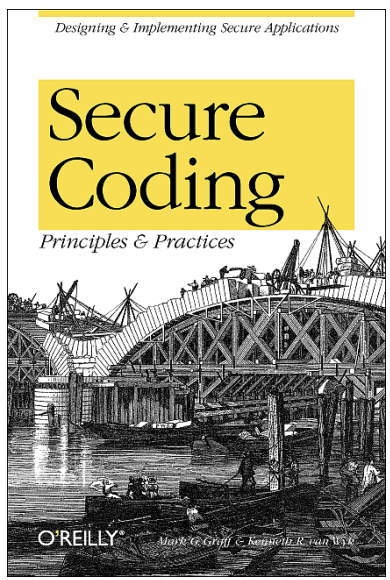


- A big part of systems integration it to make sure applications can work together
- Data interchange
- Running systems require skills, many different technologies, many humans needed
- Managing complexity with many systems become harder

Later today we will discuss this subject more with the hand-in assigment

Secure Infrastructures starts with architecture and design

Secure Coding: Principles and Practices af Mark G. Graff, Kenneth R. Van Wyk 2003



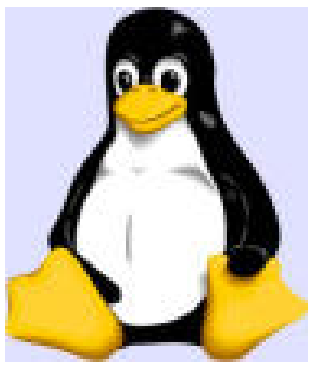
Architecture and design while you are thinking about the application

Implementation while you are writing the application

Operations After the application is in production

Approx. 200 pages, but very dense with information.

Operating Systems



- Applications need to run within some controlled system
- What is an operating system today?
- Is Docker an operating system? What is Docker?

Use the Modern Operating Systems

Newer versions of Microsoft Windows, Mac OS X and Linux

- Buffer overflow protection
- Stack protection, non-executable stack
- Heap protection, non-executable heap
- *Randomization of parameters* stack gap m.v.

Note: these still have errors and bugs, but are better than older versions

Check end-of-life and when updates will stop for each version

OpenBSD has shown the way in many cases
`http://www.openbsd.org/papers/`

Always try to make life worse and more costly for attackers

Building Secure Infrastructures

We did an exercise last time, starting to build a DMZ for servers

A real-life setup of an infrastructure from scratch can be daunting!

You need:

- Policies
- Procedures
- Incident Response

Running systems which require

- Configurations
- Settings
- Supporting infrastructure – networks
- Supporting infrastructure – logging, dashboarding, monitoring

Building something *secure* is **hard work!**

Existing infrastructures

or even worse you inherited an infrastructure

Multiple networks, with different vendors, rules

Multiple generations of services, applications, technologies

Built over decades

Varying to no documentation

Organizational challenges

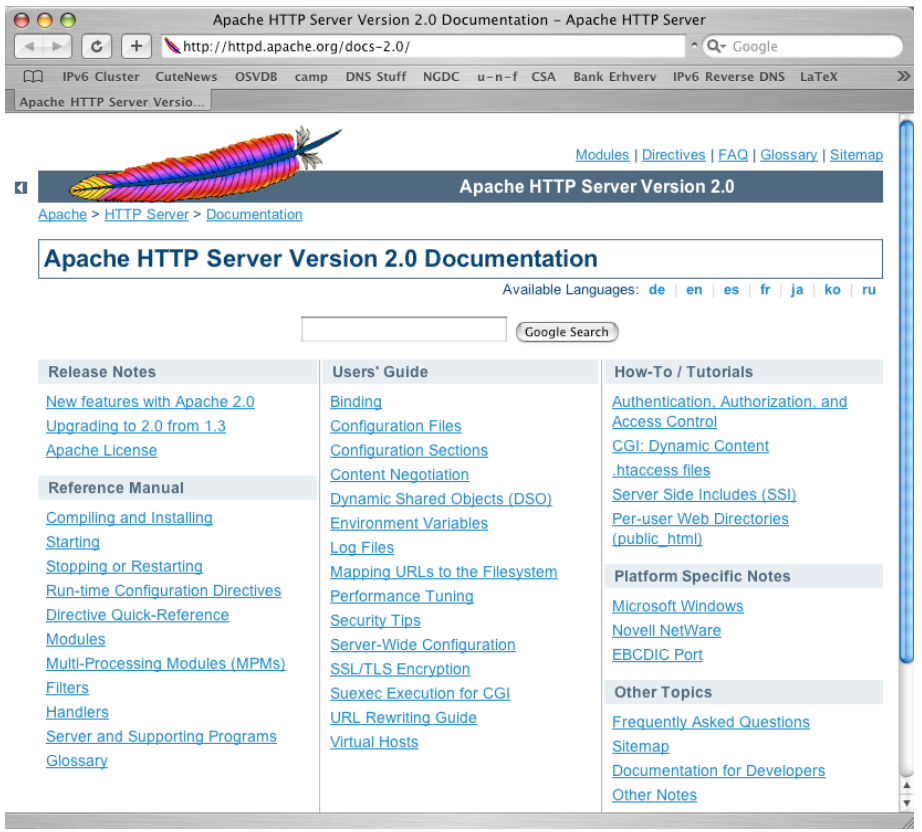
Ingrained culture – frozen in time

How do you get started improving security?

Documentation in Enterprises

What tools exist?

- Office tools like Microsoft Word and Excel
- Wikis
- Online documentation like Apache projects
- Specialized systems, collecting data automatically
- Example BornHack repositories
  `https://github.com/bornhack`



Center for Internet Security CIS Controls

> The CIS ControlsTM are a prioritized set of actions that collectively form a defense-in-depth set of best practices that mitigate the most common attacks against systems and networks. The CIS Controls are developed by a community of IT experts who apply their first-hand experience as cyber defenders to create these globally accepted security best practices. The experts who develop the CIS Controls come from a wide range of sectors including retail, manufacturing, healthcare, education, government, defense, and others.

Source: `https://www.cisecurity.org/` CIS-Controls-Version-7-1.pdf

Center for Internet Security CIS Controls 7.1

- The five critical tenets of an effective cyber defense system as reflected in the CIS Controls are:
- **Offense informs defense:** Use knowledge of actual attacks that have compromised systems to provide the foundation to continually learn from these events to build effective, practical defenses. Include only those controls that can be shown to stop known real-world attacks.
- **Prioritization:** Invest first in Controls that will provide the greatest risk reduction and protection against the most dangerous threat actors and that can be feasibly implemented in your computing environment. The CIS Implementation Groups discussed below are a great place for organizations to start identifying relevant Sub-Controls.
- **Measurements and Metrics:** Establish common metrics to provide a shared language for executives, IT specialists, auditors, and security officials to measure the effectiveness of security measures within an organization so that required adjustments can be identified and implemented quickly.
- **Continuous diagnostics and mitigation:** Carry out continuous measurement to test and validate the effectiveness of current security measures and to help drive the priority of next steps.
- **Automation:** Automate defenses so that organizations can achieve reliable, scalable, and continuous measurements of their adherence to the Controls and related metrics. Source: CIS-Controls-Version-7-1.pdf

Inventory and Control of Hardware Assets

CIS controls 1-6 are Basic, everyone must do them.

> CIS Control 1:
> Inventory and Control of Hardware Assets
> Actively manage (inventory, track, and correct) all hardware devices on the network so that only authorized devices are given access, and unauthorized and unmanaged devices are found and prevented from gaining access.

What is connected to our networks?

What firmware do we need to install on hardware?

Where IS the hardware we own?

What hardware is still supported by vendor?

Source: Center for Internet Security CIS Controls 7.1 CIS-Controls-Version-7-1.pdf

Inventory and Control of Software Assets

CIS Control 2:
Inventory and Control of Software Assets
Actively manage (inventory, track, and correct) all software on the network so that only authorized software is installed and can execute, and that all unauthorized and unmanaged software is found and prevented from installation or execution.

What licenses do we have? Paying too much?

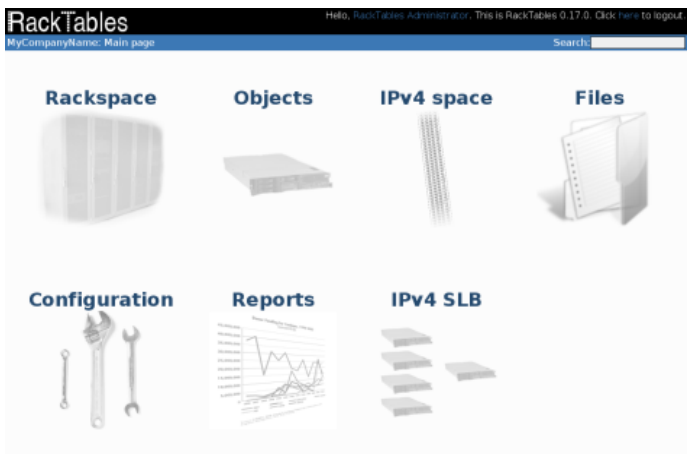
What versions of software do we depend on?

What software needs to be phased out, upgraded?

What software do our employees need to support?

Source: Center for Internet Security CIS Controls 7.1 CIS-Controls-Version-7-1.pdf

Hardware asset management

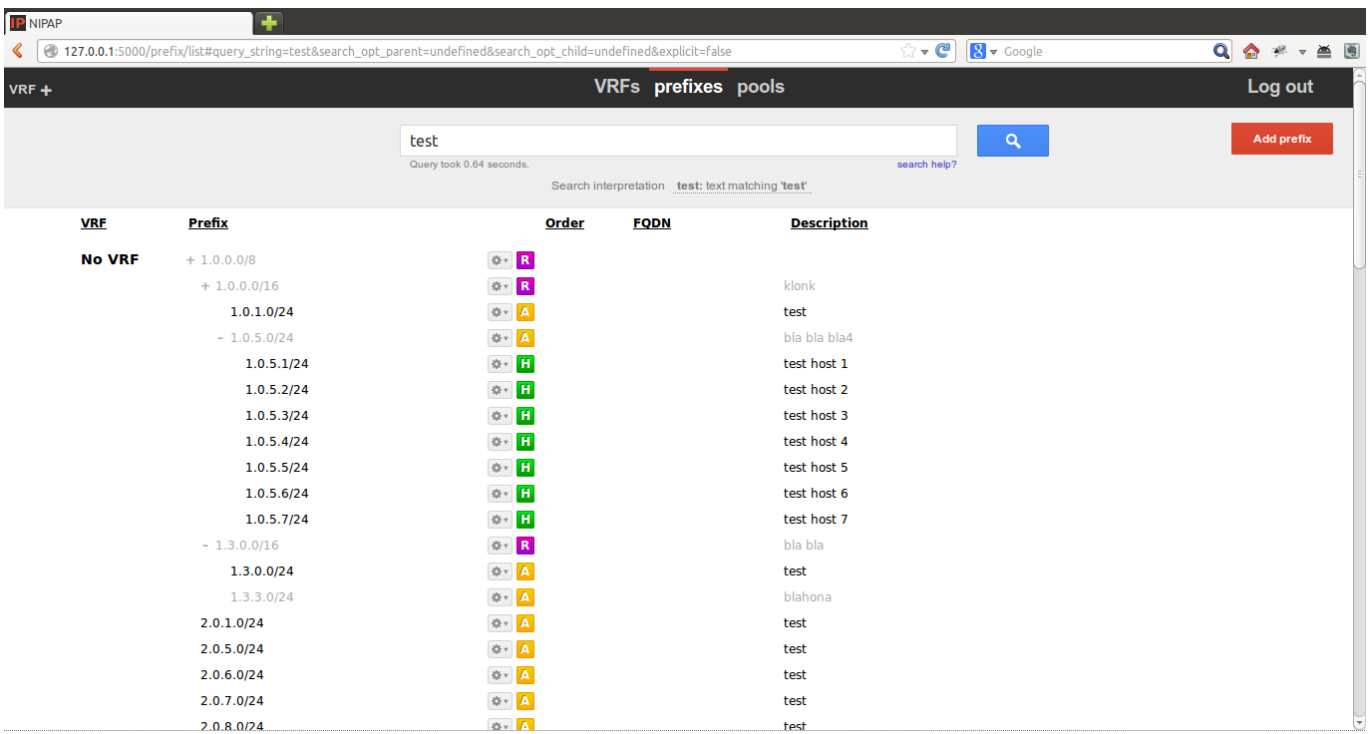


- Many systems exist
- Recommend systems designed for this task, like RackTables:
  Have a list of all devices you've got, Have a list of all racks and enclosures, Mount the devices into the racks, Maintain physical ports of the devices and links between them
- `https://www.racktables.org/`

Software asset management - virtual archives

- Software - licenses, procurement, use, upgrade prices
- Virtual servers - is a server an asset, or the data?
- IP addresses - current price per IPv4 public IP is about $30
- Data archives - GDPR, pictures is from Version2.dk listing data leaks

IP Address Management IPAM



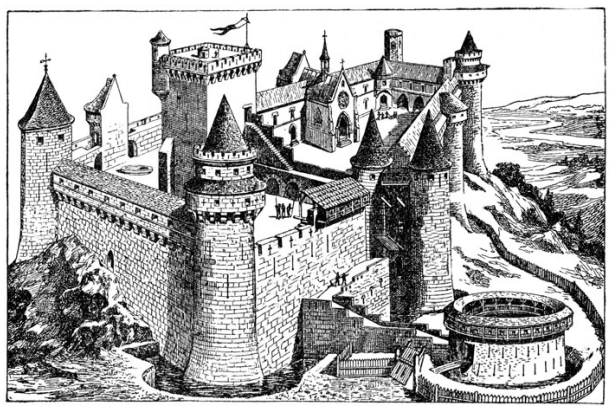
- Recommend Nipap `http://spritelink.github.io/NIPAP/`

What about dependencies

.

- Are you using some special software, or hardware
- Does your application depend on some tools, library that needs help

Building a Castle



- Open discussion how to select technologies
- - and how to manage it, starting with documentation
- Need to have, nice to have, alternatives
- Which technologies do you know
- Which technologies do you need?

Reading Summary

Camel ch 12-13

Chapter 12: Transactions and idempotency

Chapter 13: Parallel processing

Camel in action, Claus Ibsen and Jonathan Anstey, 2018, 2nd edition ISBN: 978-1-61729-293-4

Camel chapter 12: Transactions and idempotency

This chapter covers

- Understanding why you need transactions
- Using and configuring transactions
- Understanding the differences between local and global transactions
- Using transactions with messaging and databases
- Rolling back transactions
- Preventing duplicate messages by using idempotency
- Learning about the idempotent repository implementations shipped out of the box

Source:

Camel in action, Claus Ibsen and Jonathan Anstey, 2018, 2nd edition ISBN: 978-1-61729-293-4

A Transaction

1. Find the book *Camel in Action*, 2nd Edition.
2. Put the book into the basket.
3. Maybe continue shopping and look for other books.
4. Go to the checkout.
5. Enter shipping and credit card details.
6. Confirm the purchase.
7. Wait for the confirmation.
8. Leave the web store.

The ultimate resolution of this transaction is one of two states: either the purchase is accepted and confirmed, or the purchase is declined, leaving your credit card balance uncharged.

Source:

Camel in action, Claus Ibsen and Jonathan Anstey, 2018, 2nd edition ISBN: 978-1-61729-293-4

ACID

In computer science, **ACID (atomicity, consistency, isolation, durability)** is a set of properties of database transactions intended to guarantee validity even in the event of errors, power failures, etc. In the context of databases, a sequence of database operations that satisfies the ACID properties (and these can be perceived as a single logical operation on the data) is called a transaction. For example, a transfer of funds from one bank account to another, even involving multiple changes such as debiting one account and crediting another, is a single transaction.

Source:

`http://en.wikipedia.org/wiki/ACID`

Atomic Transactions

> That's why the series of events is described as atomic: either they all are completed or they all fail—it's all or nothing. In transactional terms, they either *commit* or *roll back*.

Source:

Camel in action, Claus Ibsen and Jonathan Anstey, 2018, 2nd edition ISBN: 978-1-61729-293-4

- Books uses Spring TransactionManager
- I recommend using available - and mature solutions like this – don't write your own if you can avoid it
- Which one is up to you though!

The Spring JmsTransactionManager

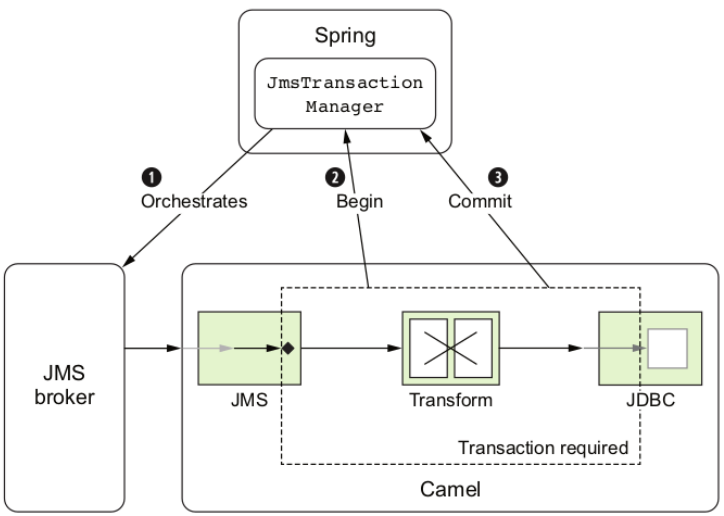
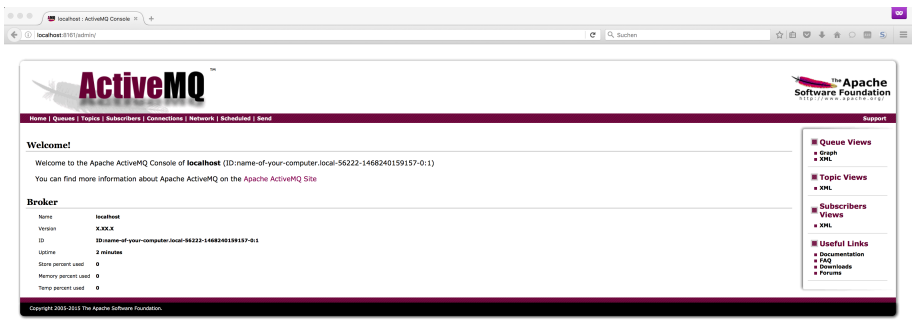


Figure 12.4 The Spring `JmsTransactionManager` orchestrates the transaction with the JMS broker. The Camel route completes successfully and signals the commit to the `JmsTransactionManager`.

- A database transaction is started, make changes and is then committed – written as one into the database
- May update multiple tables and rows

A Note about testing



For example, the book's source code uses Apache ActiveMQ and Derby as live resources. We picked these because they can be easily downloaded using Apache Maven and they're lightweight and embeddable, which makes them perfect for unit testing.

- Many developers write tests, for functions
- Many do not write tests for transactions like shown in the book!
- The book show it is possible, by including freely available tools with Maven

Global Transactions Jta- vs Jms-TransactionManager

This transaction manager is appropriate for handling distributed transactions, i.e. transactions that span multiple resources, and for controlling transactions on application server resources (e.g. JDBC DataSources available in JNDI) in general. For a single JDBC DataSource, DataSourceTransactionManager is perfectly sufficient

In Java, JTA is an implementation of the XA standard protocol, which is a global transaction protocol. To be able to use XA, the resource drivers must be XA- compliant, which some JDBC and most JMS drivers are. JTA is part of the Java EE spec- ification, which means that any Java EE–compliant application server must provide JTA support. This is one of the benefits of Java EE servers, which have JTA out of the box, unlike some lightweight alternatives, such as Apache Tomcat.

Source: Camel book and web site spring.io

`https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/jms/connection/JmsTransactionManager.html`

`https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/transaction/jta/JtaTransactionManager.html`

JTA software

JTA is also available in OSGi containers such as Apache Karaf, ServiceMix, or JBoss Fuse. Using JTA outside a Java EE server takes some work to set up because you have to find and use a JTA transaction manager, such as one of these:

- Atomikos (external third party) – `http://www.atomikos.com`
- Narayana (JBoss AS/WildFly) – `http://narayana.io`
- Apache Geronimo (Java EE) – `http://geronimo.apache.org`
- Apache Aries (OSGi platform) – `http://aries.apache.org`

For more information on JTA, see the Wikipedia page on the subject: `http://en.wikipedia.org/wiki/Java_Transaction_API` . XA is also briefly discussed here: `http://en.wikipedia.org/wiki/X/Open_XA` .

Idempotency

> The term idempotent is used in mathematics to describe a function that can be applied multiple times without changing the result beyond the initial result. In computing, idempotent is used to describe an operation that will produce the same results if exe- cuted once or multiple times. Idempotency is documented in the EIP book as the Idempotent Consumer pattern.

In Camel, the Idempotent Consumer EIP is used to ensure routing a message once and only once. To achieve this, Camel needs to be able to detect duplicate messages, which involves the following two procedures:

- Generating a unique key for each message
- Storing and retrieving previously seen keys

Source:

Camel in action, Claus Ibsen and Jonathan Anstey, 2018, 2nd edition ISBN: 978-1-61729-293-4

Camel chapter 13: Parallel processing

This chapter covers

- Camel's threading model
- Configuring thread pools and thread profiles
- Using concurrency with EIPs
- Handling scalability with Camel
- Writing asynchronous Camel components

Source:

Camel in action, Claus Ibsen and Jonathan Anstey, 2018, 2nd edition ISBN: 978-1-61729-293-4

Concurrency is another word for multitasking, and we multitask all the time in our daily lives.

Thread Pools

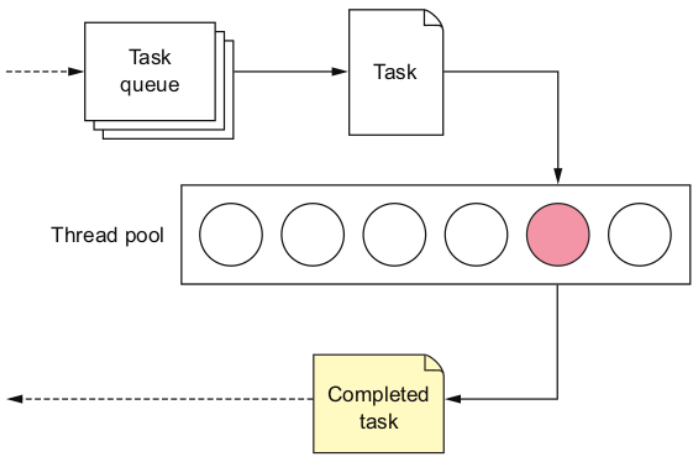


Figure 13.5 Tasks from the task queue wait to be executed by a thread from the thread pool.

- Similar is found in databases, and frameworks using databases

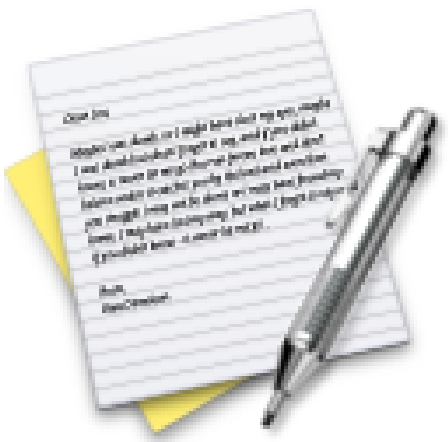
Best practices related to concurrency and scalability

- *Use concurrency if possible* – Concurrency can greatly speed up your applications. Note that using concurrency requires business logic that can be invoked in a concurrent manner.
- *Tweak thread pools judiciously* – Tweak thread pools only when you have a means of measuring the changes. It's often better to rely on the default settings.
- *Use asynchronous processing for high scalability* – If you require high scalability, try using the Camel components that support the asynchronous processing model.
- *Take care when implementing your own asynchronous component* – You're required to structure your component code according to numerous rules.
- *Reactive systems* Reactive streams and frameworks are gaining in popularity.

Source:

Camel in action, Claus Ibsen and Jonathan Anstey, 2018, 2nd edition ISBN: 978-1-61729-293-4

Run some of the examples from the book?



- If we have time, lets try running the examples from the book chapters 12 and 13

Hand-inn assignment hints and walk-through

Hand-in assignment hints and walk-through

- Read the hand-in assignment description
- Number the tasks per chapter, main chapters are listed
- Do each chapter before going to the next one

Help:
Think about what you would like to receive if you were responsible for buying hardware, selecting software and hiring people.

Hand-in assignment I

Hand-in assignment I: Describe the system environment for an organisation

Assignment:
Consider a system environment running the services we have presented in this course. The services are:

- Tomcat J2EE server, we ran this early in the course and this can be used for running Camel in production
- Camel we have run this multiple times during the course
- PostgreSQL Server

Report Contents

1. Company back story, create back story similar to the SOA book chapter 2 for your fictive company
2. Describe the software requirements for each of the above systems and create summary of the environment as a whole. Any synergies?
   Dont forget that these systems cannot float in free space, but requires one or more operating systems, which you must choose. Maybe include some monitoring and configuration management like Ansible.
3. Describe the hardware requirements for your initial deployment, taking into account creating a more production ready system. Do you need development, staging, testing, production systems, some redundancy?
4. Suggest an initial deployment overview - little detail, with a naming scheme to use for servers, physical and virtual
5. Create a list of skills requirements for running this environment. Consider job postings for similar jobs, and you may copy parts of that and adapt

Report Contents

The report should include the following sections at least:

- Title, Table of contents, formal report thanks
- Confidentiality agreement – Write "Confidential"on each page
- Appendices

Must be handed in as PDF and latest on May 7, 2020 at 23:59. Teams up to five are allowed. Make sure to list team members in the report.

Expect PDF as A4, portrait mode around 10 pages with illustrations as needed. No more than 15 pages if 3 members. Up to 25 pages if five members.

This should be a formal business report with page numbers, ToC etc.

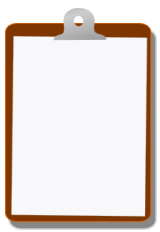
How Will the Report Look

Hand-in Assignment I: Zencurity System Environment

Author: Henrik Kramselund Jereminsen

Table of Contents

For Next Time



Think about the subjects from this time, write down questions

Check the plan for chapters to read in the books

Visit web sites and download papers if needed

Retry the exercises to get more confident using the tools