Welcome to

12. Repetition and Summary

KEA System Integration F2020 10 ECTS

Henrik Kramselund Jereminsen hkj@zencurity.com @kramse

Slides are available as PDF, kramse@Github

13-repetition-and-summary-system-integration.tex in the repo security-courses

Goals for today

Todays goals:

- Repeat stuff, relate it, get an overview og system integration
- Talk about exam, online exam, subjects

Photo by Thomas Galler on Unsplash

Time schedule

- 08:15 - 09:00 and
- 09:15 - 10:00 2x sessions with 15min break
  Repeat stuff, relate to other parts

- 10:15 - 11:30
  Try to tie everything together – get an overview
- 12:15 - 13:45 Talk about exam, online exam, subjects
  Questions, discussions and finishing up

Plan for today

- Repeat stuff, relate it, get an overview og system integration
- What is system integration
- How does XML, JSON, APIs technologies relate to System Integration
- What are patterns and SOA
- Can we use the patterns
- Which patterns do we feel confident using
- How would we use them

  Exercises

-

Exam prep

Lets talk about it later today

For now, will repeat stuff, relate to other parts
Try to tie everything together – get an overview

Feel free to ask questions today

Definition

System integration is defined in engineering as the process of bringing together the component sub-systems into one system (an aggregation of subsystems cooperating so that the system is able to deliver the overarching functionality) and ensuring that the subsystems function together as a system,[1] and in information technology[2] as the process of linking together different computing systems and software applications physically or functionally,[3] to act as a coordinated whole.

The system integrator integrates discrete systems utilizing a variety of techniques such as computer networking, enterprise application integration, business process management or manual programming.[4]

Source:
`https://en.wikipedia.org/wiki/System_integration`

Intended Learning Outcomes

To get acquainted with the challenges of developing business applications

To understand the difference between

- tightly coupled and loosely coupled system
- synchronous and asynchronous integration

To get an overview of existing technologies and solutions in system integration

To get programming practice in developing P2P integration using networking protocols

Course Description

From: STUDIEORDNING

Knowledge

The objective is to give the student knowledge of

- business considerations associated with system integration
- standards and standardization organizations
- storage, transformation and integration of data resources
- techniques used in data conversion and migration
- the service concept and understanding of its connection with service-oriented architecture
- technologies that can be used to implement a service-oriented architecture
- integration tools

Skills

The objective is that the students acquire the ability to

- use object-oriented system in service-oriented architecture
- design a system for easy integration with other systems and using existing services
- transform or expand a system, so that it can work in a service-oriented architecture
- apply patterns that support system integration
- develop supplementary modules for generic systems
- integrate generic and other systems
- choose from different methods of integration
- translate elements of a business strategy into concrete requirements for system integration

Proficiencies

Proficiencies

The objective is that the students acquired proficiency in

- choosing from different integration techniques
- acquiring knowledge about development in standards for integration
- adapting IT architecture so that future integration of systems is taken into account
- converting elements in a business strategy to specific requirements for systems integration
- adapting a system development method, so that it supports system integration

Primary literature

Primary literature:

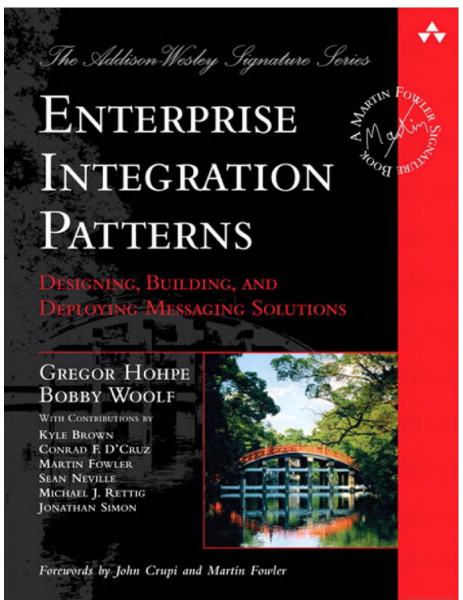
Free graphics by Lumen Design Studio

- *Enterprise Integration Patterns*, Gregor Hohpe and Bobby Woolf, 2004
  ISBN: 978-0-321-20068-6 EIP for short
- *Camel in action*, Claus Ibsen and Jonathan Anstey, 2018
  ISBN: 978-1-61729-293-4
- *Service-Oriented Architecture: Analysis and Design for Services and Microservices*,
  Thomas Erl, 2017 ISBN: 978-0-13-385858-7

Supporting literature:

- Various internet resources, to be decided

Book: Enterprise Integration Patterns



Enterprise Integration Patterns, Gregor Hohpe and Bobby Woolf, 2004
ISBN: 978-0-321-20068-6 EIP for short

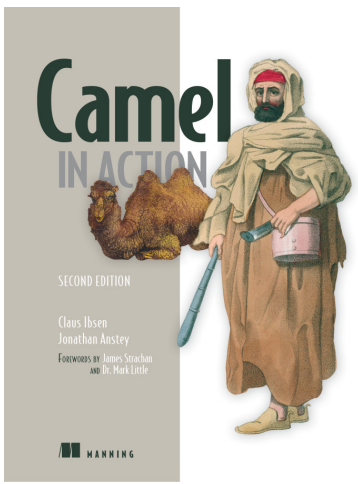
Companion Web Site

"That's why Bobby Woolf and I documented a pattern language consisting of 65 integration patterns to establish a technology-independent vocabulary and a visual notation to design and document integration solutions. Each pattern not only presents a proven solution to a recurring problem, but also documents common "gotchas"and design considerations.

The patterns are brought to life with examples implemented in messaging technologies, such as JMS, SOAP, MSMQ, .NET, and other EAI Tools. The solutions are relevant for a wide range of integration tools and platforms, such as IBM WebSphere MQ, TIBCO, Vitria, WebMethods (Software AG), or Microsoft BizTalk, messaging systems, such as JMS, WCF, Rabbit MQ, or MSMQ, ESB's such as Apache Camel, Mule, WSO2, Oracle Service Bus, Open ESB, SonicMQ, Fiorano or Fuse ServiceMix."

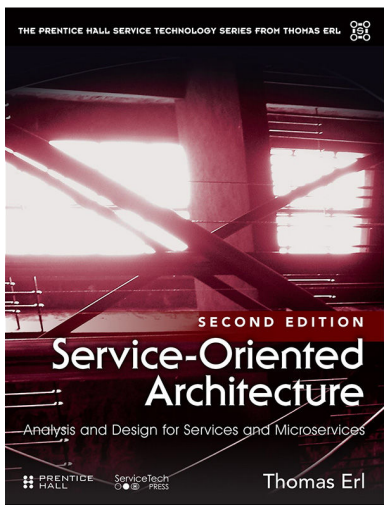
Source:
`https://www.enterpriseintegrationpatterns.com/`

Book: Camel in Action



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

Book: Service-Oriented Architecture



Service-Oriented Architecture: Analysis and Design for Services and Microservices, Thomas Erl, 2017 ISBN: 978-0-13-385858-7

Technologies used in this course

The following tools and environments are examples that were introduced in this course:

- Enterprise Integration Patterns (EIP) and Service-oriented Architecture (SOA)
- Programming languages and frameworks Java, Spring, Python
- Systems for running integration: Tomcat, Jetty, Docker, Camel
- Networking and network protocols: TCP/IP, HTTP, DNS, FTP, SMTP
- Formats XML, XSLT, YAML, JSON, WSDL, GRPC, msgpack, protobuf, apache thrift
- Web technologies and services: REST, API, SOAP
- Databases: JDBC, Postgresql, ACID
- Tools like Maven, Linux, APT, Ansible, Nginx, cURL, Git and Github
- Message queueing systems: Apache ActiveMQ, RabbitMQ and Redis
- Aggregated example platforms: Elastic stack, Logstash, Elasticsearch, Kibana
- Cloud and virtualisation Docker, Kubernetes, Azure, AWS, microservices

Exam subjects

1. Enterprise Integration Patterns (EIP)
2. Service-oriented Architecture (SOA)
3. Systems for running integration: **Camel**, Tomcat, Jetty, Docker, Kubernetes
4. Networking and network protocols: TCP/IP, HTTP, DNS, FTP, SMTP
5. Integration formats XML, XSLT, YAML, JSON, WSDL, in relation to system-integration
6. Web technologies and services: REST, API, SOAP
7. Databases: JDBC, Postgresql, ACID, persistence, resilience
8. Toolboxes Maven, Linux, APT, Ansible, Nginx, cURL, Git and Github
9. Message queueing systems: JMS, Apache ActiveMQ, RabbitMQ and Redis
10. Aggregated example platforms: Elastic stack, Elasticsearch, Logstash, Kibana

Always relate this to system integration, what part do they play in system integration

1. Enterprise Integration Patterns (EIP)

Companion Web Site

"That's why Bobby Woolf and I documented a pattern language consisting of 65 integration patterns to establish a technology-independent vocabulary and a visual notation to design and document integration solutions. Each pattern not only presents a proven solution to a recurring problem, but also documents common "gotchas"and design considerations.

The patterns are brought to life with examples implemented in messaging technologies, such as JMS, SOAP, MSMQ, .NET, and other EAI Tools. The solutions are relevant for a wide range of integration tools and platforms, such as IBM WebSphere MQ, TIBCO, Vitria, WebMethods (Software AG), or Microsoft BizTalk, messaging systems, such as JMS, WCF, Rabbit MQ, or MSMQ, ESB's such as Apache Camel, Mule, WSO2, Oracle Service Bus, Open ESB, SonicMQ, Fiorano or Fuse ServiceMix."

Source:
`https://www.enterpriseintegrationpatterns.com/`

Why Enterprise Integration Patterns?

Enterprise integration is too complex to be solved with a simple 'cookbook' approach. Instead, patterns can provide guidance by documenting the kind of experience that usually lives only in architects' heads: they are accepted solutions to recurring problems within a given context. Patterns are abstract enough to apply to most integration technologies, but specific enough to provide hands-on guidance to designers and architects. Patterns also provide a vocabulary for developers to efficiently describe their solution.

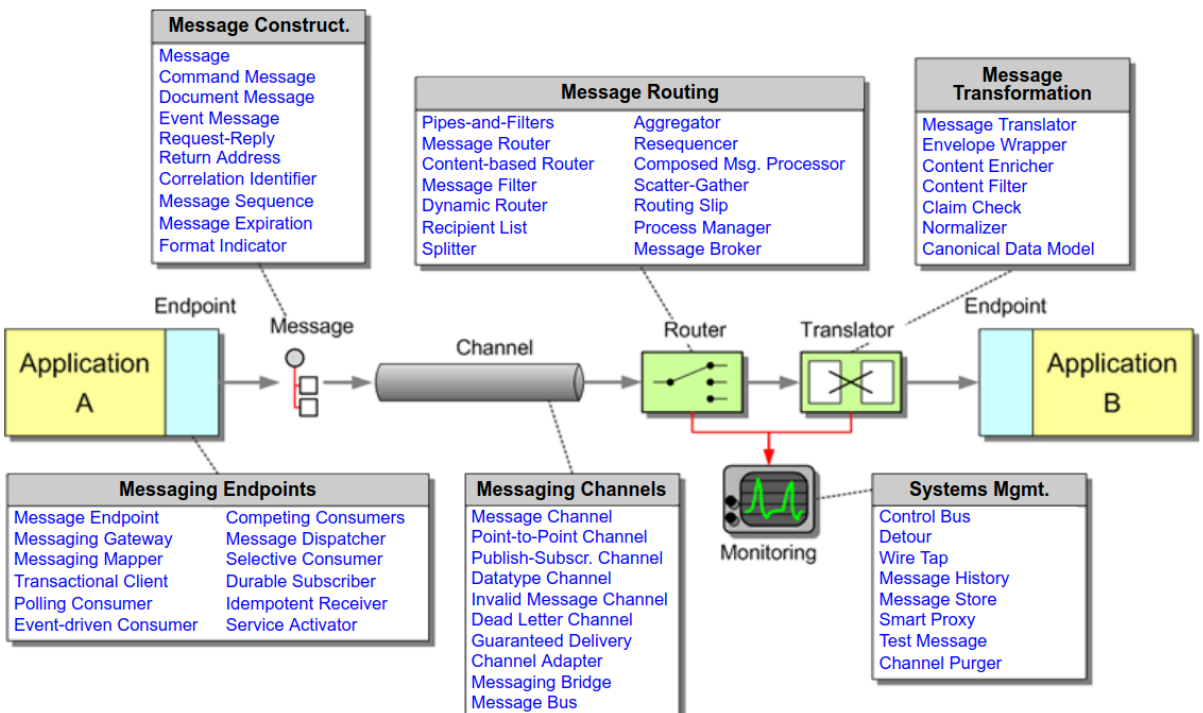
Patterns are not 'invented'; they are harvested from repeated use in practice. If you have built integration solutions, it is likely that you have used some of these patterns, maybe in slight variations and maybe calling them by a different name. The purpose of this site is not to "invent"new approaches, but to present a coherent collection of relevant and proven patterns, which in total form an integration pattern language.

Source:
`https://www.enterpriseintegrationpatterns.com/`

EIP Patterns

Messaging Patterns

We have documented 65 messaging patterns, organized as follows:



Challenges

- Networks are unreliable. The internet is always broken, somewhere a link is down, a system being booted etc.
- Networks are slow. Sending data across networks are slowers than making a local call
- Any two applications are different. Different programming languages, operating systems, and data formats
- Change is inevitable. Applications change over time
- Added: everything is linked, everything uses networking

Helpful patterns

- File Transfer(43)
- Shared database (47)
- Remote Procedure Invocation (50) - typically using Remote Procecure Call (RPC)
- Messaging (53) one application publishes a message to a common message channel, other applications read from the channel

Source: EIP book

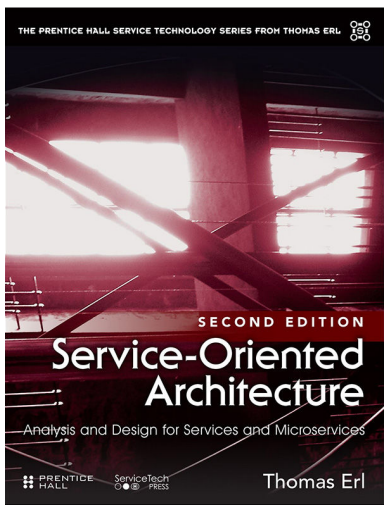
Camel Components

> Components are the primary extension point in Camel. Over the years since Camel's inception, the list of components has grown. As of version 2.20.1, Camel ships with more than 280 components, and dozens more are available separately from other com- munity sites.

Source: *Camel in action*, Claus Ibsen and Jonathan Anstey, 2018

2. Service-oriented Architecture (SOA)

Book: Service-Oriented Architecture



Service-Oriented Architecture: Analysis and Design for Services and Microservices, Thomas Erl, 2017 ISBN: 978-0-13-385858-7

Service-oriented architecture (SOA)

Service-oriented architecture (SOA) is a style of software design where services are provided to the other components by application components, through a communication protocol over a network. A SOA service is a discrete unit of functionality that can be accessed remotely and acted upon and updated independently, such as retrieving a credit card statement online. **SOA is also intended to be independent of vendors, products and technologies.[1]**

A service has four properties according to one of many definitions of SOA:[2]

- It logically represents a business activity with a specified outcome.
- It is self-contained.
- It is a black box for its consumers, meaning the consumer does not have to be aware of the service's inner workings.
- It may consist of other underlying services.[3]

Source:

`https://en.wikipedia.org/wiki/Service-oriented_architecture`

The SOA Manifesto

Through our work we have come to prioritize:

- **Business value** over technical strategy
- **Strategic goals** over project-specific benefits
- **Intrinsic interoperability** over custom integration
- **Shared services** over specific-purpose implementations
- **Flexibility** over optimization
- **Evolutionary refinement** over pursuit of initial perfection

Book references, in Appendix D the *SOA Manifesto* `www.soa-manifesto.org`.

I recommend reading the explanation in *The SOA Manifesto Explored*

Common technologies

We already mentioned some of the technologies used for this:

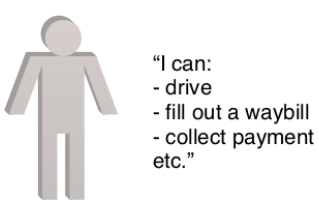
- Extensible Markup Language (XML) used in Web service (WS) with Web Services Description Language (WSDL) and Simple Object Access Protocol (SOAP)
- Allowing us to do Remote Method Invocation (RMI) aka Remote Procedure Call (RPC)
- Sometimes incorporating XML schema (XSD), Extensible Stylesheet Language Transformations (XSLT) and even producing HyperText Markup Language (HTML) documents or perhaps JavaScript Object Notation (JSON)
- Too many acronyms? Use Wikipedia!
- See the patterns and recognize common use-cases
- Or let Camel and Python convert data ☺

Services Are Collections of Capabilities

Services Are Collections of Capabilities

When discussing services, it is important to remember that a single service can offer an API that provides a collection of capabilities. They are grouped together because they relate to a functional context established by the service. The functional context of the service illustrated in Figure 3.5, for example, is that of "shipment." This particular service provides a set of capabilities associated with the processing of shipments.

Figure 3.5
Much like a human, an automated service can provide multiple capabilities.



Source: *Service-Oriented Architecture: Analysis and Design for Services and Microservices*, Thomas Erl, 2017

Service-Orientation is a Design Paradigm

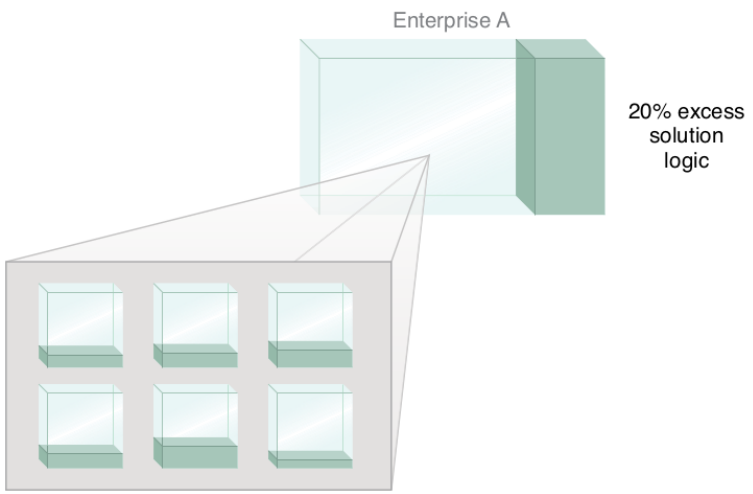
> A design paradigm is an approach to designing solution logic. When building distributed solution logic, design approaches revolve around a software engineering theory known as the "separation of concerns." In a nutshell, this theory states that a larger problem is more effectively solved when decomposed into a set of smaller problems or concerns. This gives us the option of partitioning solution logic into capabilities, each designed to solve an individual concern. Related capabilities can be grouped into units of solution logic.

- A *service composition* is a coordinated aggregate of services
- A *service inventory* is an independently standardized and governed collection of complementary services within a boundary that represents an enterprise or a meaningful segment of an enterprise.

Source: *Service-Oriented Architecture: Analysis and Design for Services and Microservices*, Thomas Erl, 2017

Problems Solved by Service-Orientation

Figure 3.14
This simple diagram portrays an enterprise environment containing applications with redundant functionality. The net effect is a larger enterprise.



- Redundant functionality is costly in the long run
- Integration Becomes a Constant Challenge

Source: *Service-Oriented Architecture: Analysis and Design for Services and Microservices*, Thomas Erl, 2017

The Need for Service-Orientation

The consistent application of the eight design principles we listed earlier results in the widespread proliferation of the corresponding design characteristics:

- increased consistency in how functionality and data is represented
- reduced dependencies between units of solution logic
- reduced awareness of underlying solution logic design and implementation details
- increased opportunities to use a piece of solution logic for multiple purposes
- increased opportunities to combine units of solution logic into different confi gurations
- increased behavioral predictability
- increased availability and scalability
- increased awareness of available solution logic

Source: *Service-Oriented Architecture: Analysis and Design for Services and Microservices*, Thomas Erl, 2017

Reusable Solution Logic

Increased Amounts of Reusable Solution Logic

Within a service-oriented solution, units of logic (services) encapsulate functionality not specific to any one application or business process (Figure 3.17). These services are therefore classified as reusable (and agnostic) IT assets.

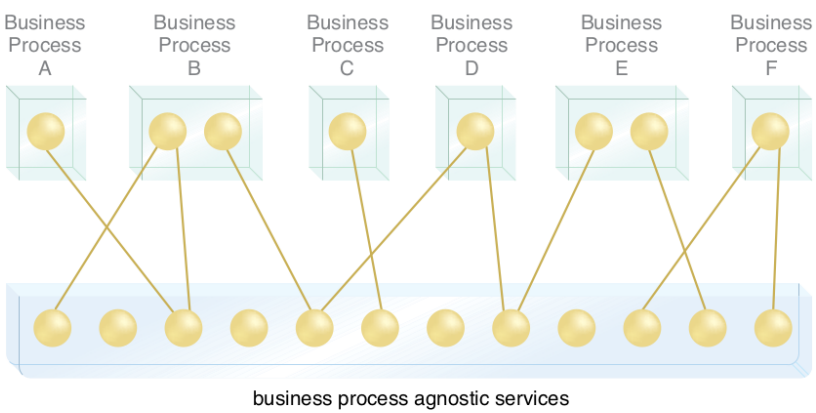


Figure 3.17

Business processes are automated by a series of business process–specific services (top layer) that share a pool of business process–agnostic services (bottom layer). These layers correspond to service models described in Chapter 5.

Source: *Service-Oriented Architecture: Analysis and Design for Services and Microservices*, Thomas Erl, 2017

3.5 Four Pillars of Service-Orientation

The four pillars of service-orientation are

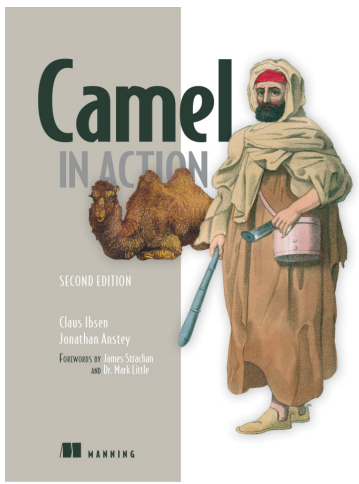
- Teamwork – Cross-project teams and cooperation are required.
- Education – Team members must communicate and cooperate based on common knowledge and understanding.
- Discipline – Team members must apply their common knowledge consistently.
- Balanced Scope – The extent to which the required levels of Teamwork, Education, and Discipline need to be realized is represented by a meaningful yet manageable scope.

Source: *Service-Oriented Architecture: Analysis and Design for Services and Microservices*, Thomas Erl, 2017

3. Systems for running integration

Camel Tomcat, Jetty, Docker, Kubernetes

Book: Camel in Action



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

3.1 Data transformation overview

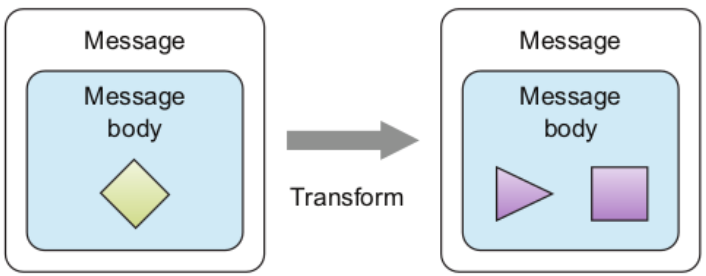


Figure 3.1 Camel offers many features for transforming data from one form to another.

- Data format transformation – The data format of the message body is transformed from one form to another. For example, a CSV record is formatted as XML.
- Data type transformation – The data type of the message body is transformed from one type to another. For example, java.lang.String is transformed into javax. jms.TextMessage .

Six ways data transformation typically takes place in Camel

- Data transformation using EIPs and Java
- Data transformation using components
- Data transformation using data formats
- Data transformation using templates
- Data type transformation using Camel's type-converter mechanism
- Message transformation in component adapters

3.3 Transforming XML

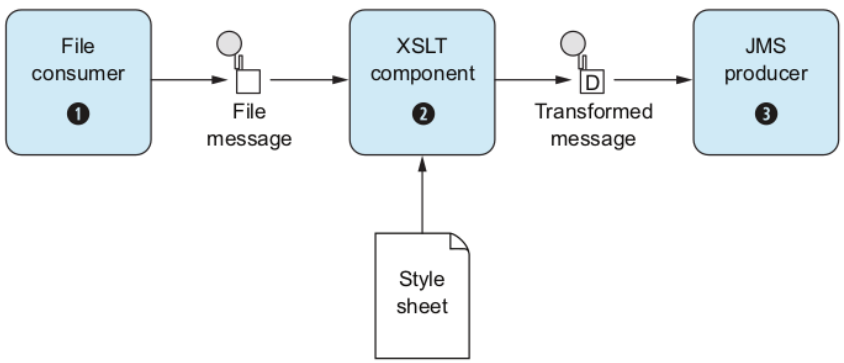


Figure 3.5 A Camel route using an XSLT component to transform an XML document before it's sent to a JMS queue

- Transforming XML with XSLT
- Transforming XML with object marshaling

Camel CSV

```
from("file://rider/csvfiles")
  .unmarshal().csv()
  .split(body()).to("jms:queue:csv.record");
```

- Camel has built-in support for many formats
- CSV is a very basic method of moving data

Chapter 1: file-copy example

```
hlk@debian-lab:~/projects/system-integration/camelinaction2/chapter1/file-copy$ find data/
data/
data/outbox
data/outbox/message1.xml
data/inbox
data/inbox/message1.xml
```

- We want to run the command for Maven to download tools, and *do stuff*
- `mvn compile exec:java`
- This might take some time!
- Note: this is a two step process, so split into `mvn compile` and `exec:java` if you have trouble running

Success Execute Java - new files

```
$ find data/
data/
data/outbox
data/outbox/message1.xml
data/outbox/message2.txt
data/inbox
data/inbox/message1.xml
data/inbox/message2.txt
```

- We added a new file using editor, and re-ran
  `echo "some data" > data/inbox/message2.txt`

Apache Tomcat

The Apache Tomcat® software is an open source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket technologies. The Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket specifications are developed under the Java Community Process.

- Allows the deployment of web applications J2EE
  `https://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition`
- Allows the use of Java security policies
- Contains the core functionality found in commercial packages
- `http://Tomcat.apache.org/`

Java Apps, Tomcat, XML config

We will download Apache Tomcat, and perform the following:

- Download the software use version 9.0.30
  I downloaded `apache-Tomcat-9.0.30.tar.gz`, Windows users take the `.zip`
- Unpack and Run the software, see it works
- Work with Tomcat manager
- Check the configuration - which is in XML
- Change the configuration - make the software listen on all IPs, specific IP

Tomcat Tasks

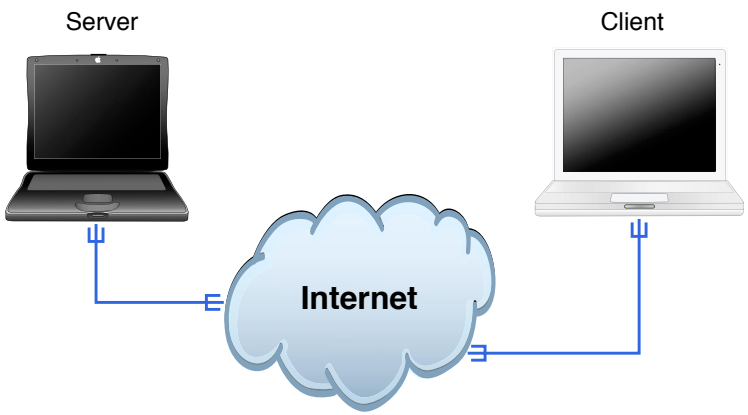
- Try going to `http://127.0.0.1:8080/manager/` - get 401 Unauthorized
  Read and fix the problem
- Look into the XML file `conf/server.xml`
  Change the port 8080 into something else, does port 80/tcp work?
- XML configuration files are very common in the Java and system integration space
- Tools often found in Java world, Ant, Maven, Tomcat, Spring
- Read `https://Tomcat.apache.org/Tomcat-9.0-doc/monitoring.html` about Java Management Extensions (JMX) `https://en.wikipedia.org/wiki/Java_Management_Extensions`
  Note: references to RMI, JMS, SNMP, HTTP etc.

4. Networking and network protocols

TCP/IP, HTTP, DNS, FTP, SMTP

After introducing these it would be natural to introduce Camel and how it can support these methods and protocols

Internet Today

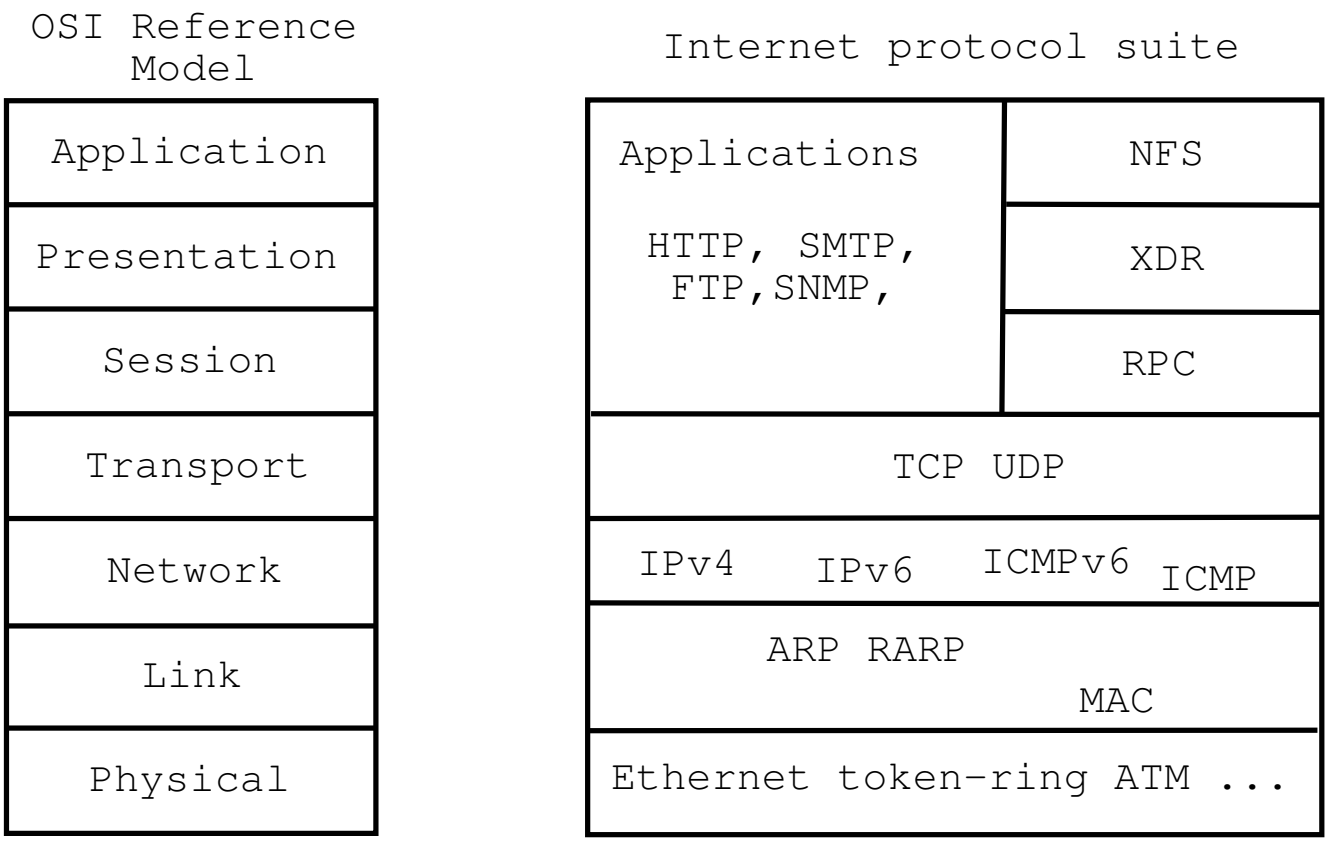


Clients and servers, roots in the academic world

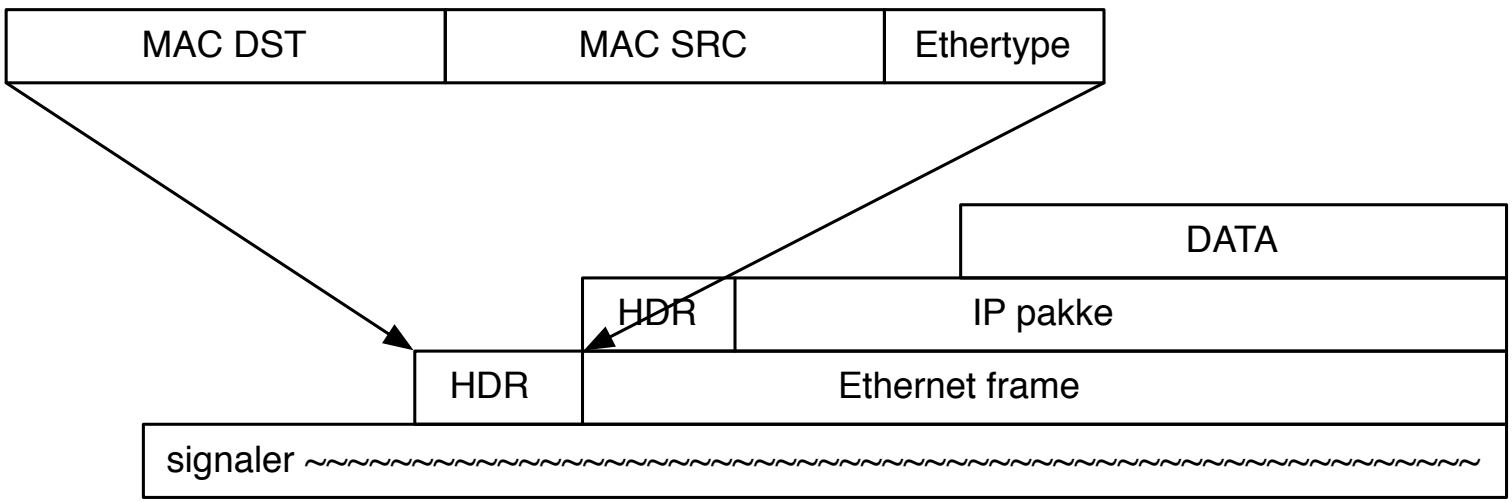
Protocols are old, some more than 20 years

Very little is encrypted, mostly HTTPS

OSI og Internet modellerne



Packets across the wire or wireless



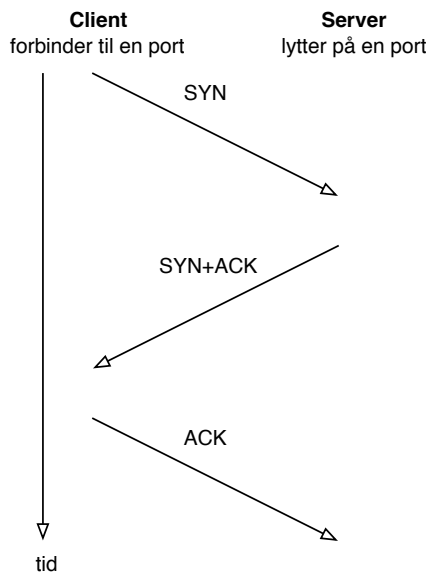
Looking at data as a stream the packets are a pattern laid on top

Network technology defines the start and end of a frame, example Ethernet

From a lower level we receive a packet, example 1500-bytes from Ethernet driver

Operating system masks a lot of complexity

TCP three way handshake



- Session setup is used in some protocols
- Other protocols like HTTP/2 can perform request in the first packet

Well-known port numbers

IANA maintains a list of magical numbers in TCP/IP

Lists of protocl numbers, port numers etc.

A few notable examples:

- Port 25/tcp Simple Mail Transfer Protocol (SMTP)
- Port 53/udp and 53/tcp Domain Name System (DNS)
- Port 80/tcp Hyper Text Transfer Protocol (HTTP)
- Port 443/tcp HTTP over TLS/SSL (HTTPS)

Source: `http://www.iana.org`

Communicate with HTTP

Try this - use netcat/ncat, available in Nmap package from `Nmap.org`:

```
$ netcat www.zencurity.com 80
GET / HTTP/1.0

HTTP/1.1 200 OK
Server: nginx
Date: Sat, 01 Feb 2020 20:30:06 GMT
Content-Type: text/html
Content-Length: 0
Last-Modified: Thu, 04 Jan 2018 15:03:08 GMT
Connection: close
ETag: "5a4e422c-0"
Referrer-Policy: no-referrer
Accept-Ranges: bytes
...
```

Nginx Load Balancer

```
http {
    upstream myapp1 {
        least_conn;
        server srv1.example.com;
        server srv2.example.com;
        server srv3.example.com;
    }

    server {
        listen 80;

        location / {
            proxy_pass http://myapp1;
        }
    }
}
```

Example from: `http://nginx.org/en/docs/http/load_balancing.html`

Basic test tools TCP - Telnet and OpenSSL

Telnet used for terminal connections over TCP, but is clear-text

Telnet can be used for testing connections to many older protocols which uses text commands

- `telnet mail.kramse.dk 25` create connection to port 25/tcp
- `telnet www.kramse.dk 80` create connection to port 80/tcp

Encrypted connections often use TLS and can be tested using OpenSSL command line tool `openssl`

- `openssl s_client -host www.kramse.dk -port 443`
  create connection to port 443/tcp with TLS
- `openssl s_client -host mail.kramse.dk -port 993`
  create connection to port 993/tcp with TLS

Using OpenSSL in client-mode allows the use of the same commands like Telnet after connection

Camel SMTP

Whenever you send an email, you're using the Simple Mail Transfer Protocol (SMTP) under the hood. In Camel, an SMTP URI looks like this: `[smtp|stmps]://[username@]host[:port][?options]`

Table 6.15 Common URI options used to configure the Mail component

Option	Default value	Description
`password`		The password of the user account corresponding to username in the URI.
`subject`		Sets the subject of the email being sent. You can override this value by setting a Subject message header.
`from`	`camel@localhost`	Sets what email address will be used for the From field of the email being sent.
`to`	`username@host`	The email address you're sending to. Multiple addresses must be separated by commas.
`cc`		The carbon copy (CC) email address you're sending to. Multiple addresses must be separated by commas.

SMTP should always be protected with password!

5. Integration formats XML, XSLT, JSON, WSDL

Make sure to relate to system-integration

Dont be afraid to talk about your opinion about these

Data overview XML data, JSON

Photo by Chris Lawton on Unsplash

XML data

> Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The World Wide Web Consortium's XML 1.0 Specification[2] of 1998[3] and several other related specifications[4]—all of them free open standards—define XML.[5]
>
> The design goals of XML emphasize simplicity, generality, and usability across the Internet.[6] It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures[7] such as those used in web services.

Source: `https://en.wikipedia.org/wiki/XML`

- We have seen XML used for configuration in Apache Tomcat and Camel
- Perfect for computers, less for humans

XML data example - Nmap output

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE nmaprun>
<?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl" type="text/xsl"?>
<!-- Nmap 7.70 scan initiated Sat Feb 22 23:35:53 2020 as: nmap -oA router -sP 10.0.42.1 -->
<nmaprun scanner="nmap" args="nmap -oA router -sP 10.0.42.1" start="1582410953"
 startstr="Sat Feb 22 23:35:53 2020" version="7.70" xmloutputversion="1.04">
<verbose level="0"/>
<debugging level="0"/>
<host><status state="up" reason="echo-reply" reason_ttl="62"/>
<address addr="10.0.42.1" addrtype="ipv4"/>
<hostnames>
</hostnames>
<times srtt="2235" rttvar="5000" to="100000"/>
</host>
<runstats><finished time="1582410953" timestr="Sat Feb 22 23:35:53 2020" elapsed="0.32"
 summary="Nmap done at Sat Feb 22 23:35:53 2020; 1 IP address (1 host up)
 scanned in 0.32 seconds" exit="success"/><hosts up="1" down="0" total="1"/>
</runstats>
</nmaprun>
```

XML data - documents

> Hundreds of document formats using XML syntax have been developed,[8] including RSS, Atom, SOAP, SVG, and XHTML. XML-based formats have become the default for many office-productivity tools, including Microsoft Office (Office Open XML), OpenOffice.org and LibreOffice (OpenDocument), and Apple's iWork[citation needed]. XML has also provided the base language for communication protocols such as XMPP. Applications for the Microsoft .NET Framework use XML files for configuration, and property lists are an implementation of configuration storage built on XML.[9]

Source: `https://en.wikipedia.org/wiki/XML`

- Document formats using XML may still be proprietary!
- Documents using XML can be validated, are they well-formed according to the Document Type Definition (DTD)

XML data - standards

> Many industry data standards, such as Health Level 7, OpenTravel Alliance, FpML, MISMO, and National Information Exchange Model are based on XML and the rich features of the XML schema specification. Many of these standards are quite complex and it is not uncommon for a specification to comprise several thousand pages.[citation needed] In publishing, Darwin Information Typing Architecture is an XML industry data standard. XML is used extensively to underpin various publishing formats.

Source: `https://en.wikipedia.org/wiki/XML`

- Dont worry too much about standards at this time
- The important standards will often be defined by the business area

XML data for Service-oriented architecture (SOA)

XML is widely used in a Service-oriented architecture (SOA). Disparate systems communicate with each other by exchanging XML messages. The message exchange format is standardised as an XML schema (XSD). This is also referred to as the canonical schema. XML has come into common use for the interchange of data over the Internet. IETF RFC:3023, now superseded by RFC:7303, gave rules for the construction of Internet Media Types for use when sending XML. It also defines the media types application/xml and text/xml, which say only that the data is in XML, and nothing about its semantics.

Source: `https://en.wikipedia.org/wiki/XML`

Transforming XML using XSLT

> XSLT (Extensible Stylesheet Language Transformations) is a language for transforming XML documents into other XML documents,[1] or other formats such as HTML for web pages, plain text or XSL Formatting Objects, which may subsequently be converted to other formats, such as PDF, PostScript and PNG.[2] XSLT 1.0 is widely supported in modern web browsers.[3]

Source: `https://en.wikipedia.org/wiki/XSLT`

- Can be seen as a mapping between formats, different XML schemas
- Also is Turing complete, is a programming language

XSLT example

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml" indent="yes"/>
  <xsl:template match="/persons">
    <root> <xsl:apply-templates select="person"/> </root>
  </xsl:template>
  <xsl:template match="person">
    <name username="{@username}"> <xsl:value-of select="name" /> </name>
  </xsl:template>
</xsl:stylesheet>
```

- XSLT uses XPath to identify subsets of the source document tree and perform calculations. XPath also provides a range of functions
- XSLT functionalities overlap with those of XQuery, which was initially conceived as a query language for large collections of XML documents
  Source: `https://en.wikipedia.org/wiki/XSLT`

xsltproc example using Nmap

```
$ su -
# apt install nmap xsltproc
# nmap -sP -oA /tmp/router 91.102.91.18
# exit
$ xsltproc /tmp/router.xml > /tmp/router.html
$ firefox /tmp/router.html
```

- We can use the command line tool `xlstproc` for transforming documents
- `apt install xsltproc`
- Its part of the package Libxslt `https://en.wikipedia.org/wiki/Libxslt`

- Try installing the tools Nmap and `xsltproc` and reproduce the above
- This is an easy tool to try, and very useful too

Data overview JSON

> JavaScript Object Notation (JSON, pronounced / d e sən/; also / d e s n/[note 1]) is an open-standard file format or data interchange format that uses **human-readable text** to transmit data objects consisting of attribute–value pairs and array data types (or any other serializable value). It is a very common data format, with a diverse range of applications, such as serving as replacement for XML in AJAX systems.[6]

Source: `https://en.wikipedia.org/wiki/JSON`

- I like JSON much better than XML
- Many web services can supply data in JSON format

JSON example

```
{
  "first name": "John",
  "last name": "Smith",
  "age": 25,
  "address": {
    "street address": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postal code": "10021"
  },
  "phone numbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
  ],
}
```

- This is a basic JSON document, new data attribute-value pairs can be added
  Source: `https://en.wikipedia.org/wiki/JSON`

6. Web technologies and services: REST, API, SOAP

Introduce W3C, HTTP, then move quickly to the subjects REST, API, SOAP

Web Services

The term Web service (WS) is either:

- a service offered by an electronic device to another electronic device, communicating with each other via the World Wide Web, or
- a server running on a computer device, listening for requests at a particular port over a network, serving web documents (HTML, JSON, XML, images), and creating web applications services, which serve in solving specific domain problems over the Web (WWW, Internet, HTTP)

Source: `https://en.wikipedia.org/wiki/Web_service`

- Today a generic name for services using the internet
- Web servers such as Apache HTTPD, Nginx etc. provide a service to thew internet allowing access using HTTP
- Source for some parts on this slide, `https://en.wikipedia.org/wiki/Web_service`

W3C Web Services

> A web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards.

Source – W3C, Web Services Glossary[3]

SOAP - Simple Object Access Protocol

> SOAP (abbreviation for Simple Object Access Protocol) is a messaging protocol specification for exchanging structured information in the implementation of web services in computer networks. Its purpose is to provide extensibility, neutrality, verbosity and independence. It uses XML Information Set for its message format, and relies on application layer protocols, most often Hypertext Transfer Protocol (HTTP), although some legacy systems communicate over Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission.

Source: `https://en.wikipedia.org/wiki/SOAP`

Utilizes UDDI (Universal Description, Discovery, and Integration)

Web Service Explained

The term "Web service"describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards over an Internet Protocol backbone. XML is the data format used to contain the data and provide metadata around it, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI lists what services are available.

Source:`https://en.wikipedia.org/wiki/Web_service`

WSDL - Web Services Description Language

The Web Services Description Language (WSDL / w z dəl/) is an XML-based interface description language that is used for describing the functionality offered by a web service. The acronym is also used for any specific WSDL description of a web service (also referred to as a WSDL file), which provides a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns. Therefore, its purpose is roughly similar to that of a type signature in a programming language.

The current version of WSDL is WSDL 2.0. The meaning of the acronym has changed from version 1.1 where the "D"stood for "Definition".

Source: `https://en.wikipedia.org/wiki/Web_Services_Description_Language`

WSDL XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<description xmlns="http://www.w3.org/ns/wsdl"
             xmlns:tns="http://www.tmsws.com/wsdl20sample"
             xmlns:whttp="http://schemas.xmlsoap.org/wsdl/http/"
             xmlns:wsoap="http://schemas.xmlsoap.org/wsdl/soap/"
             targetNamespace="http://www.tmsws.com/wsdl20sample">

<documentation>
    This is a sample WSDL 2.0 document.
</documentation>
```

Source: https://en.wikipedia.org/wiki/Web_Services_Description_Language

WSDL XML types

```
<!-- Abstract type -->
   <types>
      <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
                 xmlns="http://www.tmsws.com/wsdl20sample"
                 targetNamespace="http://www.example.com/wsdl20sample">

         <xs:element name="request"> ... </xs:element>
         <xs:element name="response"> ... </xs:element>
      </xs:schema>
   </types>
```

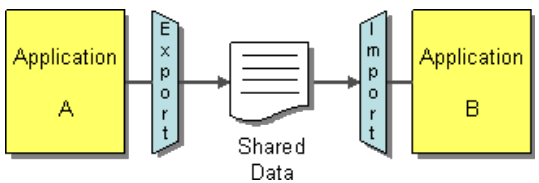
Source: `https://en.wikipedia.org/wiki/Web_Services_Description_Language`

Types Describes the data. The XML Schema language (also known as XSD) is used (inline or referenced) for this purpose.

7. Databases: JDBC, Postgresql

We have used JDBC and Postgresql as examples What is ACID, persistence, resilience

File Transfer



File Transfer — Have each application produce files of shared data for others to consume, and consume files that others have produced.

Common systems and technologies used:

- File Transfer Protocol (FTP) - old protocol, uses clear text password - should not be used, but still is
- SFTP/SCP - replaces FTP, Secure FTP/ Secure Copy is part of the Secure Shell (SSH) protocol - available since 1995
- Hyper Text Transfer Protocol / HTTP Secure (HTTP/HTTPS) - web based protocols

Using Persistence

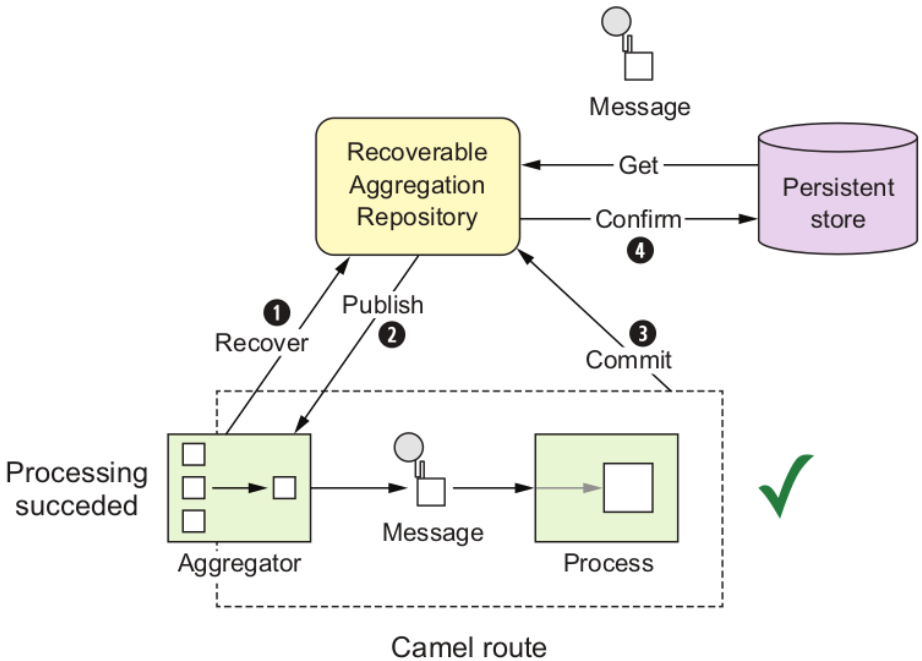


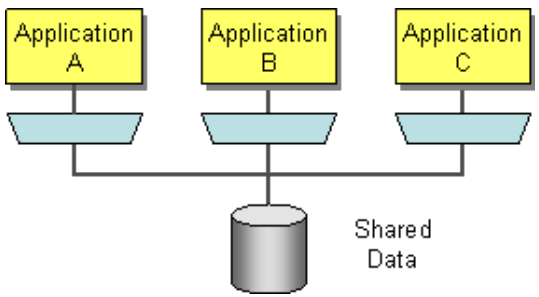
Figure 5.5 The Aggregator recovers failed messages ❶, which are published again ❷, and this time the messages completed ❸ successfully ❹.

Camel database support

In pretty much every enterprise-level application, you need to integrate with a database at some point, so it makes sense that Camel has first-class support for accessing databases. Camel has five components that let you access databases in various ways:

- *JDBC component* – Allows you to access JDBC APIs from a Camel route. item SQL component—Allows you to write SQL statements directly into the URI of the component for using simple queries. This component can also be used for call- ing stored procedures.
- *JPA component* – Persists Java objects to a relational database by using the Java Persistence Architecture.
- *Hibernate component* – Persists Java objects by using the Hibernate framework. This component isn't distributed with Apache Camel because of licensing incompatibilities. You can find it at the camel-extra project (https://github.com/camel-extra/camel-extra).
- *MyBatis component* – Allows you to map Java objects to relational databases.

Shared Database



Shared Database — Have the applications store the data they wish to share in a common database.

Common systems and technologies used:

- database management system (DBMS) using Structured Query Language (SQL), relational database examples:

- PostgresSQL, Oracle DM, Microsoft SQL, MySQL `https://en.wikipedia.org/wiki/SQL`
- NoSQL databases has been a new input with examples like: MongoDB, CouchDB, Redis, RIAK `https://en.wikipedia.org/wiki/NoSQL`

PostgreSQL

New to PostgreSQL?

PostgreSQL is a powerful, open source object-relational database system with over 30 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance.

There is a wealth of information to be found describing how to install and use PostgreSQL through the official documentation. The PostgreSQL community provides many helpful places to become familiar with the technology, discover how it works, and find career opportunities. Reach out to the community here.

Relational databases are used around the world for storing production data. When doing system integration projects we will often need to read or store data in databases, so a minimum of knowledge about these are needed.

- About relational database systems RDBMS `https://en.wikipedia.org/wiki/Relational_database`
- The home page of PostgreSQL `https://www.postgresql.org/`

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!

JMS to Database example

To demonstrate the SQL command-message concept, let's revisit the order router at Rider Auto Parts. In the accounting department, when an order comes in on a JMS queue, the accountant's business applications can't use this data. They can only import data from a database. That means any incoming orders need to be put into the corporate database. Using Camel, a possible solution is illustrated in figure 6.10.

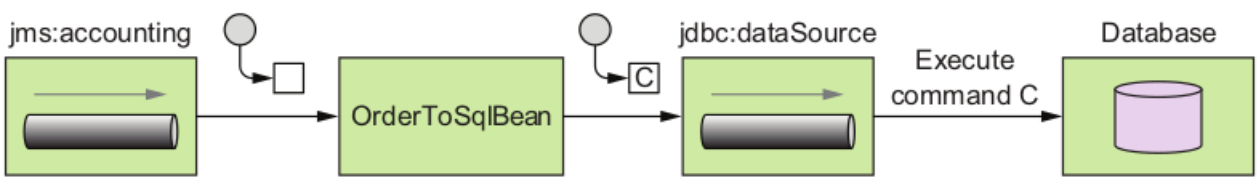


Figure 6.10 A message from the JMS accounting queue is transformed into an SQL command message by the `OrderToSqlBean` bean. The JDBC component then executes this command against its configured data source.

```
from("jms:accounting")
.to("bean:orderToSql")
.to("jdbc:dataSource?useHeadersAsParameters=true");
```

Uses a bean for mapping

Listing 6.3 A bean that converts an incoming order to a SQL statement:

```
public class OrderToSqlBean {
    public String toSql(@XPath("order/@name") String name,
                        @XPath("order/@amount") int amount,
                        @XPath("order/@customer") String customer,
                        @Headers Map<String, Object> outHeaders) {
        outHeaders.put("partName", name);
        outHeaders.put("quantity", amount);
        outHeaders.put("customer", customer);
        return "insert into incoming_orders"
            + "(part_name, quantity, customer) values"
            + " (:?partName, :?quantity, :?customer)";
    }
}
```

The last part is an SQL statement doing the *inserting* into the database

8. Toolboxes

What are things you need to run and control development within system integration, examples could be:

Maven, Linux, APT, Ansible, Nginx, cURL, Git and Github

Java tools Needed for Camel Maven

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

```
hlk@debian-lab:~$ mvn -v
Apache Maven 3.5.4 (1edded0938998edf8bf061f1ceb3cfdeccf443fe; 2018-06-17T20:33:14+02:00)
Maven home: /home/user/projects/system-integration/apache-maven-3.5.4
Java version: 11.0.6, vendor: Debian, runtime: /usr/lib/jvm/java-11-openjdk-amd64
```

- Maven - mvn command
- `https://en.wikipedia.org/wiki/Apache_Maven`

Chapter 1: file-copy example

```
hlk@debian-lab:~/projects/system-integration/camelinaction2/chapter1/file-copy$ find data/
data/
data/outbox
data/outbox/message1.xml
data/inbox
data/inbox/message1.xml
```

- We want to run the command for Maven to download tools, and *do stuff*
- `mvn compile exec:java`
- This might take some time!
- Note: this is a two step process, so split into `mvn compile` and `exec:java` if you have trouble running

Success Execute Java - shortened for slide

```
hlk@debian-lab:~/projects/system-integration/camelinaction2/chapter1/file-copy$ mvn exec:java
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------< com.camelinaction:chapter1-file-copy >----------------
[INFO] Building Camel in Action 2 :: Chapter 1 :: File Copy Example 2.0.0
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- exec-maven-plugin:1.2.1:java (default-cli) @ chapter1-file-copy ---
[ion.FileCopierWithCamel.main()] DefaultCamelContext  INFO  Apache Camel 2.24.3 (CamelContext: camel-1) is starting
[ion.FileCopierWithCamel.main()] FileEndpoint         INFO  Using default memory based idempotent repository with cache max size: 10
[ion.FileCopierWithCamel.main()] DefaultCamelContext  INFO  Route: route1 started and consuming from: file://data/inbox?noop=true
[ion.FileCopierWithCamel.main()] DefaultCamelContext  INFO  Total 1 routes, of which 1 are started
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 11.908 s
[INFO] Finished at: 2020-02-17T07:11:18+01:00
[INFO] ------------------------------------------------------------------------
```

Git intro

> Git (/ ɡɪt/)[7] is a distributed version-control system for tracking changes in source code during software development.[8] It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed,[9] data integrity,[10] and support for distributed, non-linear workflows.[11]

Source: `https://en.wikipedia.org/wiki/Git`

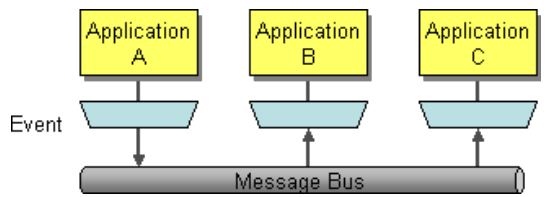
- We will introduce Git and Github - commercial Git hosting company `https://en.wikipedia.org/wiki/Git`
- Try creating a Git repository, remember to add a README
- Checkout the repository
- Edit a file
- add and commit it

Use the Github Hello World example: `https://guides.github.com/activities/hello-world/`

9. Message queueing systems

We have spent a little time talking about JMS, Apache ActiveMQ, RabbitMQ and Redis

Messaging



Messaging — Have each application connect to a common messaging system, and exchange data and invoke behavior using messages.

Common systems and technologies used:

- Java Message Service (JMS) API is a Java message-oriented middleware Application Programming Interface (API)
- Apache ActiveMQ, RabbitMQ, Oracle WebLogic
- See more at `https://en.wikipedia.org/wiki/Message_passing`

Enterprise Integration Patterns: Messaging Systems

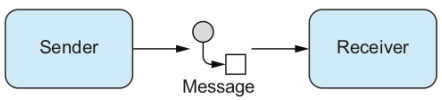


Figure 1.5 Messages are entities used to send data from one system to another.

Basic Messaging Concepts

- Channels Messaging applications transmit data through a *Message channel*, a virtual pipe that connects a sender to a receiver.
- Messages A *message* is an atomic packet of data that can be transmitted on a channel
- Pipes and filters can perform actions on the messages as they travel through the system, validation or transformed between formats
  We already saw that last time converting data with Camel
- Routing a message may have to go through several channels to reach its final destination
- Transformation, a *message translator* can convert from one format to another
- Endpoints enable applications to send and receive messages

3.1 Data transformation overview

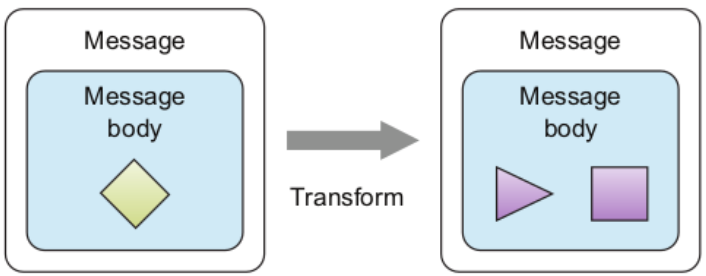


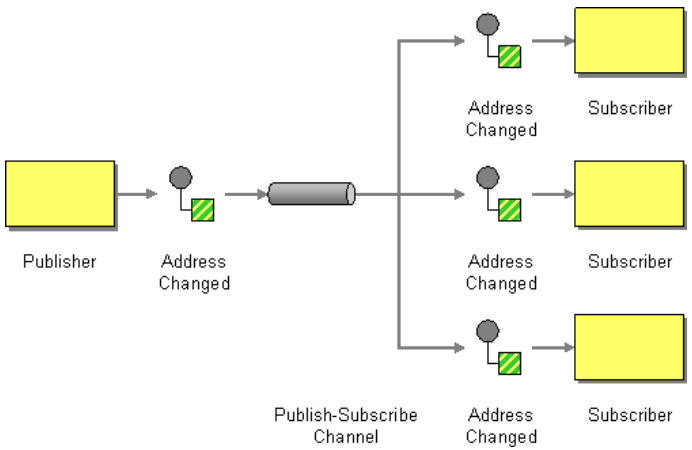
Figure 3.1 Camel offers many features for transforming data from one form to another.

- Messages consist of two parts: Header and Body
- See Internet email, picture formats, packet headers
- Data is envelopped many times during transmission!
- Data format transformation – The data format of the message body is transformed from one form to another. For example, a CSV record is formatted as XML.
- Data type transformation – The data type of the message body is transformed from one type to another. For example, java.lang.String is transformed into javax. jms.TextMessage .

SOAP Request

```
// HTTP here
POST / HTTP/1.0
Host: localhost:8080
User-agent: SOAPpy xxx (pywebsvcs.sf.net)
Content-type: text/xml; charset=UTF-8
Content-length: 340
SOAPAction: "hello"
// SOAP start here
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
  SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
>
<SOAP-ENV:Body>
<hello SOAP-ENC:root="1">
</hello>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Publish-Subscribe Channel

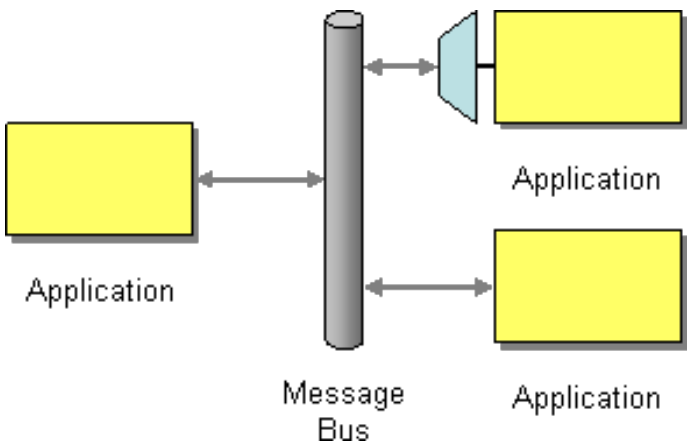


A Publish-Subscribe Channel works like this: It has one input channel that splits into multiple output channels, one for each subscriber. When an event is published into the channel, the Publish-Subscribe Channel delivers a copy of the message to each of the output channels. Each output channel has only one subscriber, which is only allowed to consume a message once. In this way, each subscriber only gets the message once and consumed copies disappear from their channels.

Source:

`https://www.enterpriseintegrationpatterns.com/patterns/messaging/PublishSubscribeChannel.html`

Message Bus



A Message Bus is a combination of a common data model, a common command set, and a messaging infrastructure to allow different systems to communicate through a shared set of interfaces. This is analogous to a communications bus in a computer system, which serves as the focal point for communication between the CPU, main memory, and peripherals. Just as in the hardware analogy, there are a number of pieces that come together to form the message bus:

Source: `https://www.enterpriseintegrationpatterns.com/patterns/messaging/MessageBus.html`

Java Message Service (JMS) Elements

- JMS provider An implementation of the JMS interface for message-oriented middleware (MOM). Providers are implemented as either a Java JMS implementation or an adapter to a non-Java MOM.
- JMS client An application or process that produces and/or receives messages. JMS producer/publisher
- A JMS client that creates and sends messages. JMS consumer/subscriber
- A JMS client that receives messages. JMS message An object that contains the data being transferred between JMS clients.
- JMS queue A staging area that contains messages that have been sent and are waiting to be read (by only one consumer). As the name queue suggests, the messages are delivered in the order sent. A JMS queue guarantees that each message is processed only once.
- JMS topic A distribution mechanism for publishing messages that are delivered to multiple subscribers.

Source: `https://en.wikipedia.org/wiki/Java_Message_Service`

Redis

> Redis (/ r d s/;[7][8] Remote Dictionary Server)[7] is an in-memory data structure project implementing a distributed, in-memory key-value database with optional durability.

- Often used as a queue, a kind of buffer between systems
- `https://en.wikipedia.org/wiki/Redis`
- `https://redislabs.com`

Apache ActiveMQ

Apache ActiveMQ™ is the most popular open source, multi-protocol, Java-based messaging server. It supports industry standard protocols so users get the benefits of client choices across a broad range of languages and platforms. Connectivity from C, C++, Python, .Net, and more is available. Integrate your multi-platform applications using the ubiquitous AMQP protocol. Exchange messages between your web applications using STOMP over websockets. Manage your IoT devices using MQTT. Support your existing JMS infrastructure and beyond. ActiveMQ offers the power and flexibility to support any messaging use-case.

`https://activemq.apache.org/`

RabbitMQ

RabbitMQ is an open-source message-broker software (sometimes called message-oriented middleware) that originally implemented the Advanced Message Queuing Protocol (AMQP) and has since been extended with a plug-in architecture to support Streaming Text Oriented Messaging Protocol (STOMP), Message Queuing Telemetry Transport (MQTT), and other protocols.[1]

The RabbitMQ server program is written in the Erlang programming language and is built on the Open Telecom Platform framework for clustering and failover. Client libraries to interface with the broker are available for all major programming languages.

Source: `https://en.wikipedia.org/wiki/RabbitMQ`

Home page: `https://www.rabbitmq.com/`

Example from RabbitMQ tutorial: Sending

Sending

The following code fragment establishes a connection, makes sure the recipient queue exists, then sends a message and finally closes the connection.

```
#!/usr/bin/env python
import pika
connection = pika.BlockingConnection(pika.ConnectionParameters(host='localhost'))
channel = connection.channel()
channel.queue_declare(queue='hello')
channel.basic_publish(exchange='', routing_key='hello', body='Hello World!')

print(" [x] Sent 'Hello World!'")

connection.close()
```

Example from: `https://www.rabbitmq.com/tutorials/tutorial-one-python.html`

Example from RabbitMQ tutorial: Receiving

Receiving

Similarly, the following program receives messages from the queue and prints them on the screen:

```
#!/usr/bin/env python
import pika
connection = pika.BlockingConnection(
    pika.ConnectionParameters(host='localhost'))
channel = connection.channel()
channel.queue_declare(queue='hello')
def callback(ch, method, properties, body):
    print(" [x] Received %r" % body)

#channel.basic_consume(queue='hello', on_message_callback=callback, auto_ack=True)
# With apt package python-pika 0.11.0-4, this works:
channel.basic_consume(callback, 'hello', no_ack=True)

print(' [*] Waiting for messages. To exit press CTRL+C')
channel.start_consuming()
```

10. Aggregated example platforms: Elastic stack

What is the benefit of using a large packet with a lot of components, Elasticsearch, Logstash, Kibana

Elasticsearch

> Elasticsearch is a search engine based on the Lucene library. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents. Elasticsearch is developed in Java.

Source: `https://en.wikipedia.org/wiki/Elasticsearch`

- Open core means parts of the software are licensed under various open-source licenses (mostly the Apache License)
- Various browser tools and plugins for ES exist, to make life easier
- We often use ES for storing Log Messages and Events from multiple systems, a SIEM Security information and event management.

Real Example using Elasticsearch

We will read about a real example:

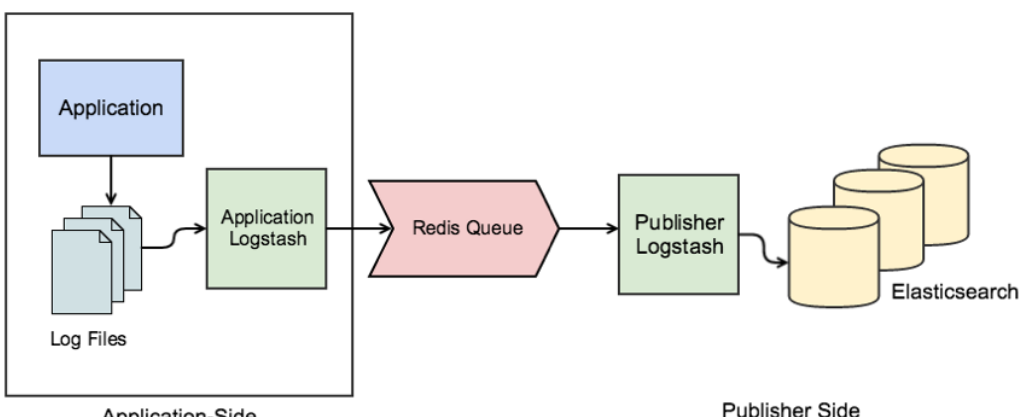
> Scaling the Elastic Stack in a Microservices Architecture @ Rightmove
> Rightmove is the UK's #1 Property Portal. In the process of helping people find the places they want to live, we serve 55 million requests a day and use Elasticsearch to power our searches and provide our teams with useful analytics to help support our applications. ... Fast forward to 2017 and we have over 50 microservices all sending their logs to our Elasticsearch cluster. In doing so, we needed a way to scale our configuration on both the hardware and application side of things. So how did we achieve this and what did we learn along the way?

Source:

`https://www.elastic.co/blog/scaling-the-elastic-stack-in-a-microservices-architecture-rightmove`

About Elasticsearch

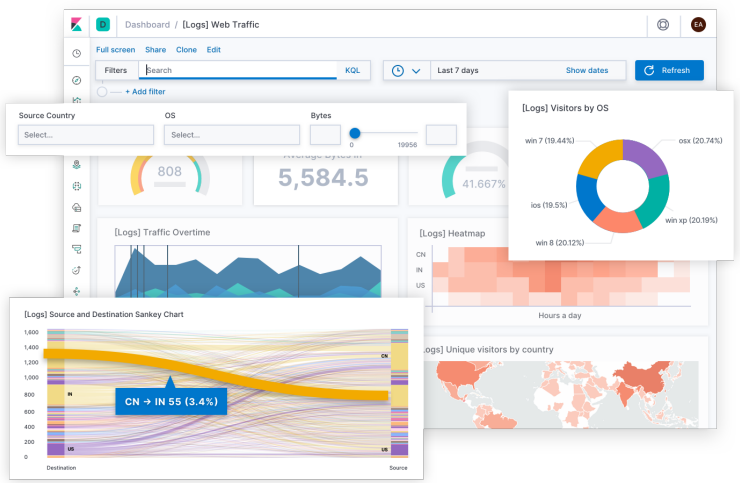
Look at the example Elasticsearch



Source:

`https://www.elastic.co/blog/scaling-the-elastic-stack-in-a-microservices-architecture-rightmove`

Elastic stack Kibana



Screenshot from `https://www.elastic.co/kibana`

Elasticsearch is a search engine and ocument store used in a lot of different systems, allowing cross application integration.

Getting started with Elastic Stack

Easy to get started using the tutorial *Getting started with the Elastic Stack* :

`https://www.elastic.co/guide/en/elastic-stack-get-started/current/get-started-elastic-stack.html`

Today Elastic Stack contains lots of different parts.

- Elasticsearch - the core engine
- Logstash - a tool for parsing logs and other data.
  `https://www.elastic.co/logstash`

  "Logstash dynamically ingests, transforms, and ships your data regardless of format or complexity. Derive structure from unstructured data with grok, decipher geo coordinates from IP addresses, anonymize or exclude sensitive fields, and ease overall processing."
- Kibana - a web application for accessing and working with data in Elasticsearch
  `https://www.elastic.co/kibana`

Part II

Questions about exam and exercises

Exam

- Exam
- Individual: Oral based on curriculum
- Graded (7 scale)
- Draw a question. Question covers a topic
- Discuss the topic, and use practical examples
- Exam is 40 minutes in total, including pulling the question and grading
- Prepare material (keywords, examples, exercises, wireshark captures) for different topics so that you can use it to help you at the exam

Above to the best of my knowledge, but may have changed!