BPM & SOA

SOA

- Service-Oriented Architecture (SOA):
  - Paradigm for structuring information & sw systems based on capabilities that part of a system offers to other parts
  - Services have higher-level of abstraction & underlying philosophy than components (delivery)
- OASIS Definition:
  - SOA is a paradigm for organising & utilizing distributed capabilities that may be under the control of different ownership domains
  - Capability wrt. business as well as specific application systems
  - Service-orientation relevant at both the business & technical level
  - SOA provides common abstractions & principles to structure systems uniformly from the IT & business perspective

SOA

- Service is a business concept but turned also as IT concept:
  - Close gap between business and IT & achieve higher degree of business-IT alignment
- Ownership:
  - Services must be delivered to exist
    - Resource encapsulated by a service should exist at particular location & must be maintained and managed by service provider so as to deliver a capability to a service consumer
  - Service providers & consumers operate independently & can exist in different ownership domains
    - Perfect fit for SOA and BPM
    - BP can span different functional domains
    - SOA aims at structuring system such that eases communication & handover between these domains

WHAT ARE WEB SERVICES?

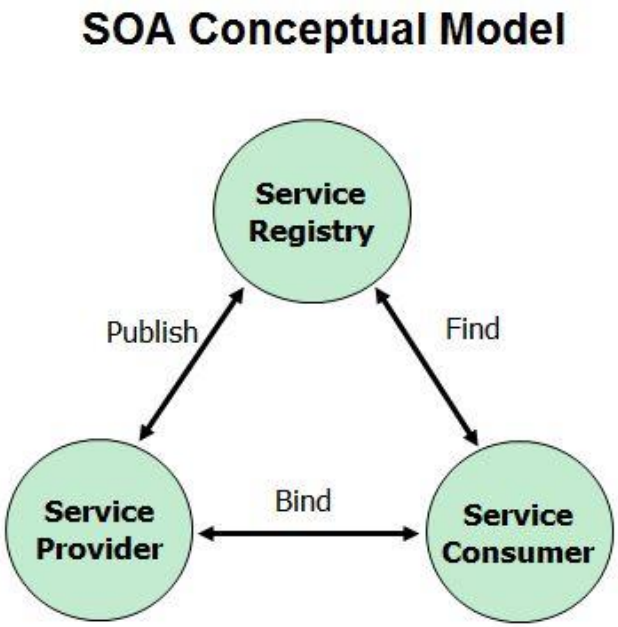
- A web service is any **piece of software** that makes itself **available over the internet** and uses a standardized **XML** messaging system. XML is used to encode all communications to a web service. For example, a client invokes a web service by sending an XML message, then waits for a corresponding XML response. As all communication is in XML, web services are not tied to any one operating system or programming language--Java can talk with Perl; Windows applications can talk with Unix applications.
- Web services are **self-contained, modular, distributed, dynamic applications** that can be described, published, located, or invoked over the network to create products, processes, and supply chains. These applications can be local, distributed, or web-based. Web services are built on top of open standards such as **TCP/IP, HTTP, Java, HTML, and XML.**

WHAT ARE WEB SERVICES?

- The basic web services platform is XML + HTTP. All the standard web services work using the following components
- **SOAP** (Simple Object Access Protocol)
- **UDDI** (Universal Description, Discovery and Integration)
- **WSDL** (Web Services Description Language)
- **REST** (REpresentational State Transfer)
- A web service enables communication among various applications by using open standards such as HTML, XML, WSDL, SOAP and REST. A web service takes the help of:
  - XML to tag the data
  - SOAP to transfer a message
  - WSDL to describe the availability of service.
  - REST is an *architectural style* and a *design* for network-based software architectures.

SOA ARCHITECTURE PRINCIPLES

- W3C Definition:
  - Service is an abstract resource that represents a capability
- Capability is offered by a service provider by performing a set of actions on behalf of service consumer at some time & place and interacting with consumer via a particular channel
- Service bus is a medium connecting service provider & consumer and comprises various technical infrastructure elements
- Service repository facilitates discovery of services & provides additional service information (constraints & service levels)



SOA ARCHITECTURE PRINCIPLES

- OASIS indicates that specific SOA aspects must be considered when analyzing and designing services for interaction, including visibility & interaction
  - Service provider & consumer must interact independently of whether they are humans or automated programs
  - Service consumer needs to know the service I/O & actions that can be performed against the service, as part of service description, for successful interaction
- Five principles apply for the identification of services:
  - Contract orientation
  - Cohesiveness
  - Coupling
  - Reusability
  - Autonomy

SOA ARCHITECTURE PRINCIPLES

- Contract orientation:
  - Service must share a formal contract (Service Level Agreement –SLA) defining terms of information exchange & commitments made by both parties (provider & consumer) to define a (business) relationship
  - Contract encompasses the description of:
    - Functional & non-functional characteristics
      - Includes description of exposed operations to be invoked
    - Remedies when violation of commitments occur
    - Ways to monitor the commitments by which party
  - Trust between parties is increased
    - Participation in contract of third parties for handling subtle points

SOA ARCHITECTURE PRINCIPLES

- Cohesiveness:
  - Refers to the concept of grouping operations when they are functionally related to the performance of a task
  - Analysis of underlying business objects is indicator of cohesiveness
    - High relation of operations on a business object indicates high cohesiveness
    - If operations of two services are highly-related, services can be merged
- Reusability:
  - Service should be useful in different cases / circumstances / scenarios & be exploited by different consumers

SOA ARCHITECTURE PRINCIPLES

- Coupling:
  - Describes strength of interdependency between multiple services & service compositions
  - Independent services are more reusable & maintainable
  - Coupling between services must be as loose as possible
    - Otherwise services must be merged
  - Levels of dependency can be minimized by minimizing the interactions between services
  - Balance between cohesion & coupling must be discovered
  - Coarse-grained interaction preferable than fine-grained one
    - Bigger size of data exchanged leads to less interactions

SOA ARCHITECTURE PRINCIPLES

- Autonomy:
  - Level of independence of a service
  - A purely autonomous service has full control of its environment
    - Increased reliability & predictability as external unpredictable influences are minimized
  - Data normalization techniques can be exploited to design operations in a non-redundant manner

TYPES OF SERVICES

- Business service:
  - is an outcome of set of operations of an organisation
  - can represent operations at different levels
  - can be aligned along the organisation hierarchical structure or be based on actual business capabilities & domains
  - may or may not leverage existing IT infrastructure
    - Distinguishable from a software service

TYPES OF SERVICES

- Software service:
  - is part of an application system that can be consumed separately by different entities
  - may enable a business service or can provide a capability that contributes to a business service delivery
  - It can also have a non-business but technical scope
  - Can be distinguished into:
    - Business-related services identified & specified based on business requirements
      - Requirements may refer to BPs, tasks or business entities
    - Technical services that are business-logic agnostic & include utility services that offer generic functions to other software services

TYPES OF SERVICES

Service-Type	Business-related service		Technical-related service	
Granularity	Business Process	Task	Entity	Utility
Composition	Composite Service		Elementary Service	
Interaction	Synchronous		Asynchronous	
Exchange Patterns	Request/Response	Notification (one-way)	Notification (one-way)	Conversational
State	Stateful		Stateless	
Accessibility	Intra-organisational		Inter-organisational	

TYPES OF SERVICES

#	SOAP	REST
1	A XML-based message protocol	An architectural style protocol
2	Uses WSDL for communication between consumer and provider	Uses XML or JSON to send and receive data
3	Invokes services by calling RPC method	Simply calls services via URL path
4	Does not return human readable result	Result is readable which is just plain XML or JSON
5	Transfer is over HTTP. Also uses other protocols such as SMTP, FTP, etc.	Transfer is over HTTP only
6	JavaScript can call SOAP, but it is difficult to implement	Easy to call from JavaScript
7	Performance is not great compared to REST	Performance is much better compared to SOAP - less CPU intensive, leaner code etc.

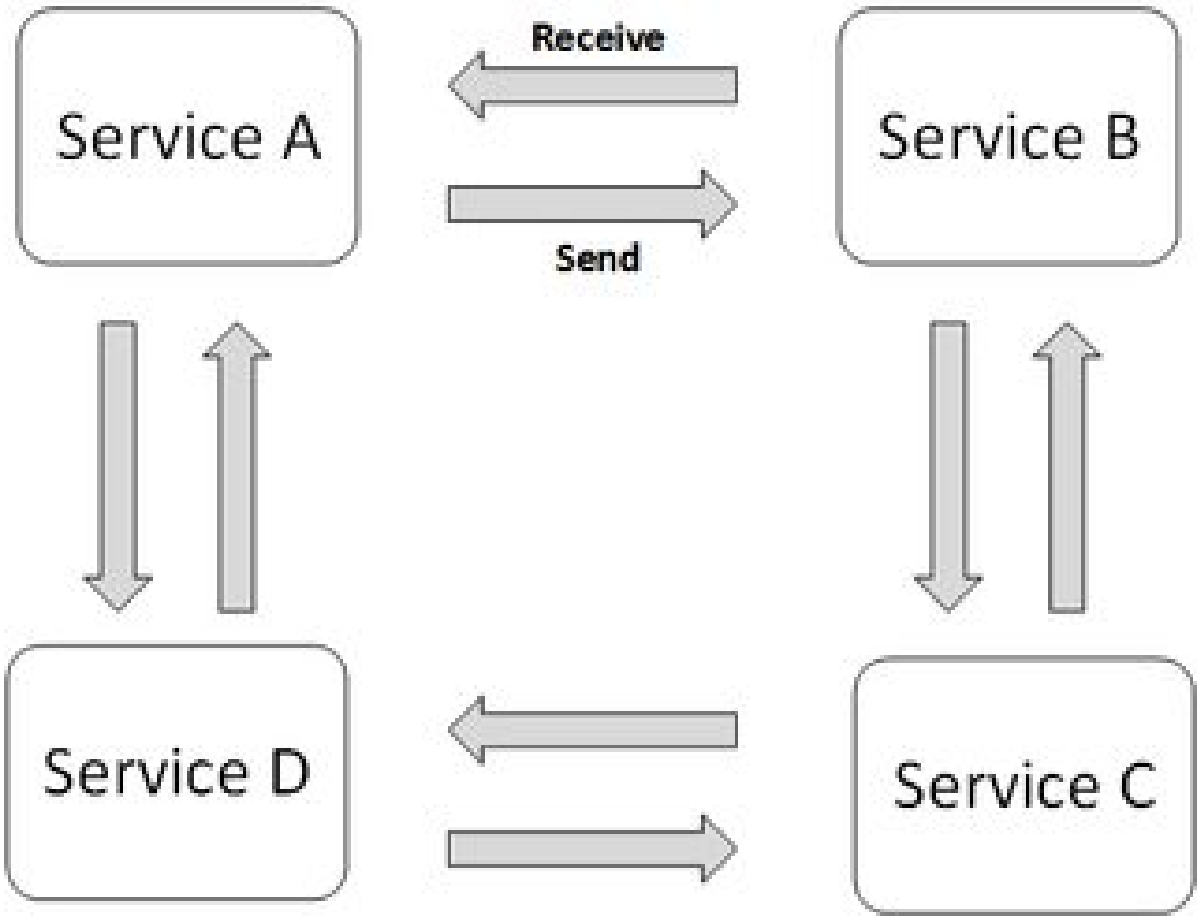
TYPES OF SERVICES

- Service can be elementary or composite
  - Elementary services can be classified into task (logic-driven), entity (data-driven) & utility services
  - Composite services can be classified into data-aggregation services & process-driven composite services
- Services can be differentiated according to interaction style, information exchange patterns, state information management & intended customer types (external, internal or both)
- Utility services:
  - Business-logic agnostic
    - Provide re-usable cross-cutting functionalities related to processing data within legacy application environments

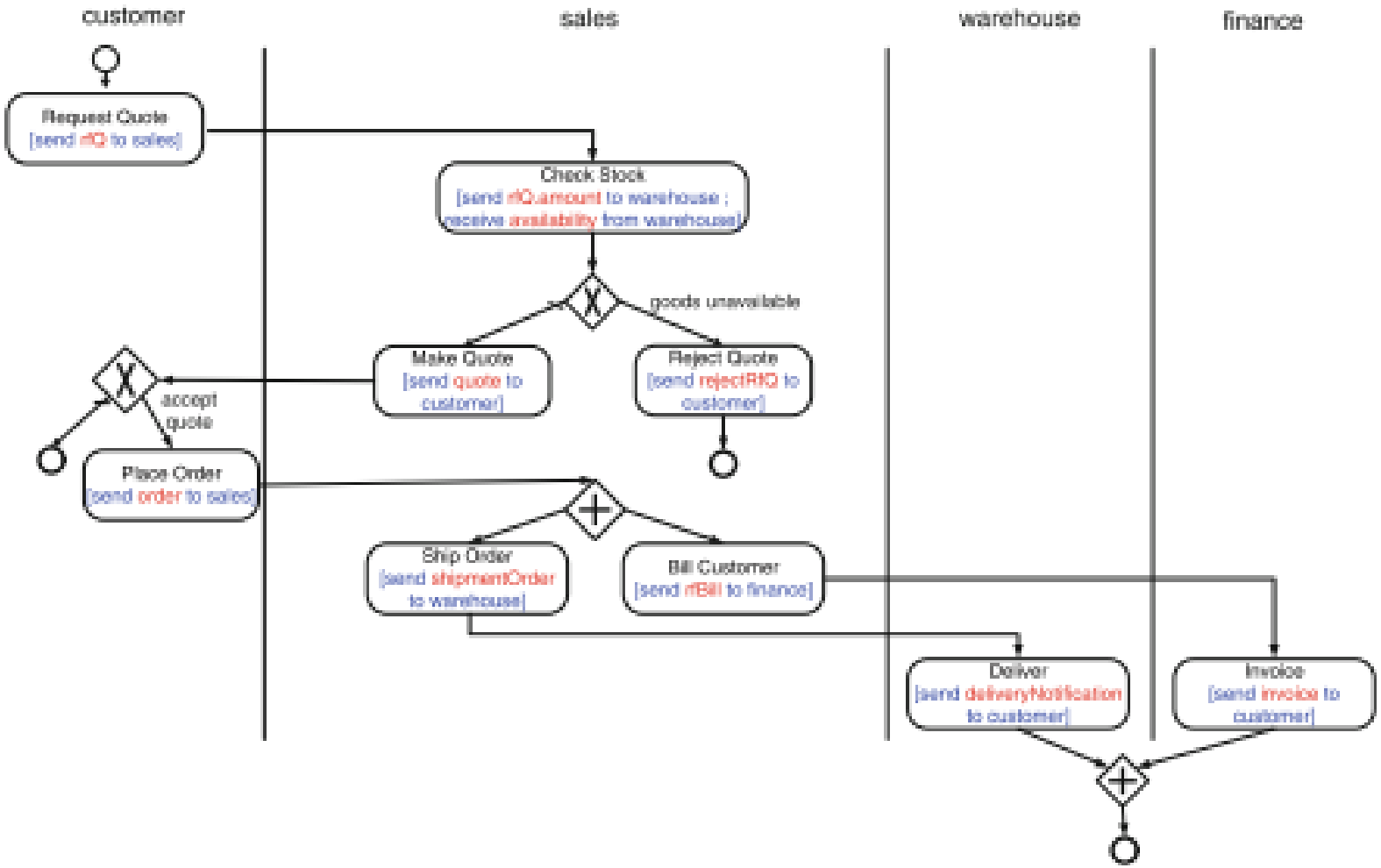
SERVICE-ENABLED PROCESS MODELS

- Service-enabled processes can be modelled either via service orchestrations or choreographies
- Service choreography:
  - Global model of interactions that can occur between a set of services in the context of a service-enabled process
  - Not only interactions but also dependencies between them are captured, including control-flow, data-flow, timing & QoS dependencies
  - High-level view of service-enabled process
    - Does not capture internal actions involved in a service whose effect is not externally visible
    - Global perspective is provided: view of a observer and not a process participant
    - Services are abstract, may not correspond to an actual service deployed on particular end-point

SERVICE CHOREOGRAPHY



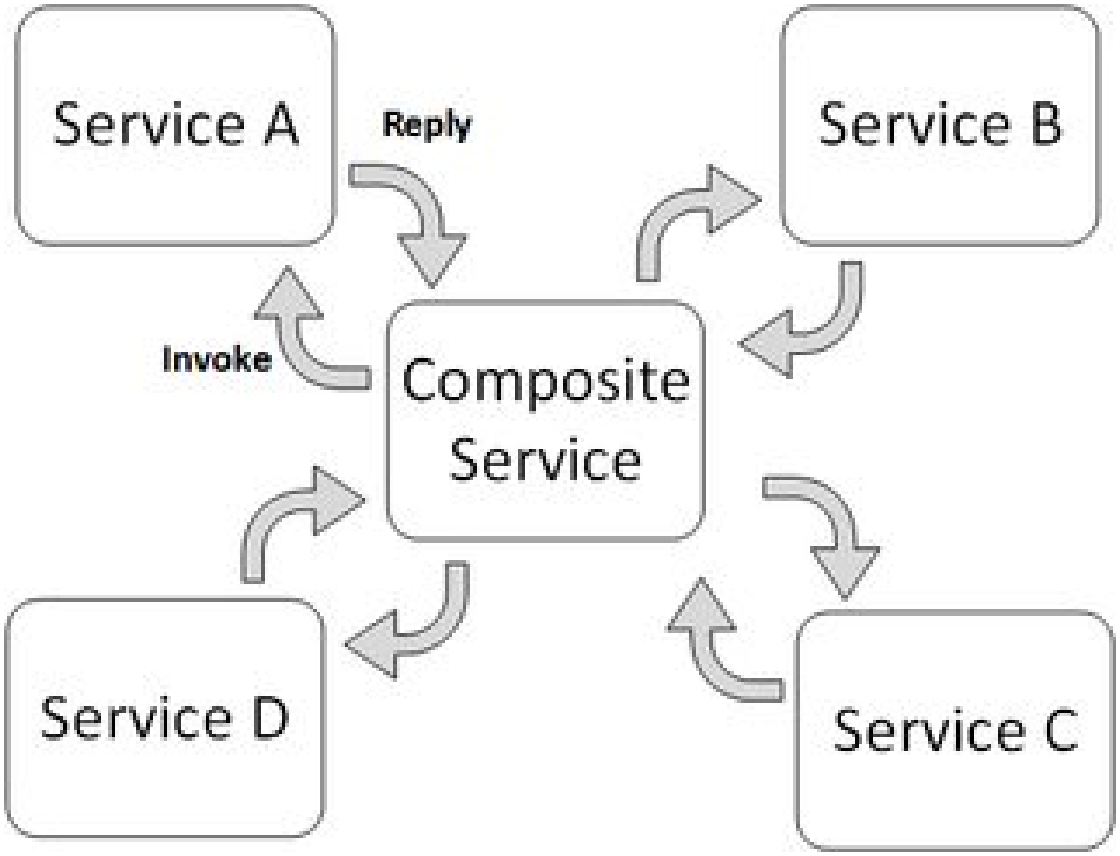
CHOREOGRAPHY EXAMPLE



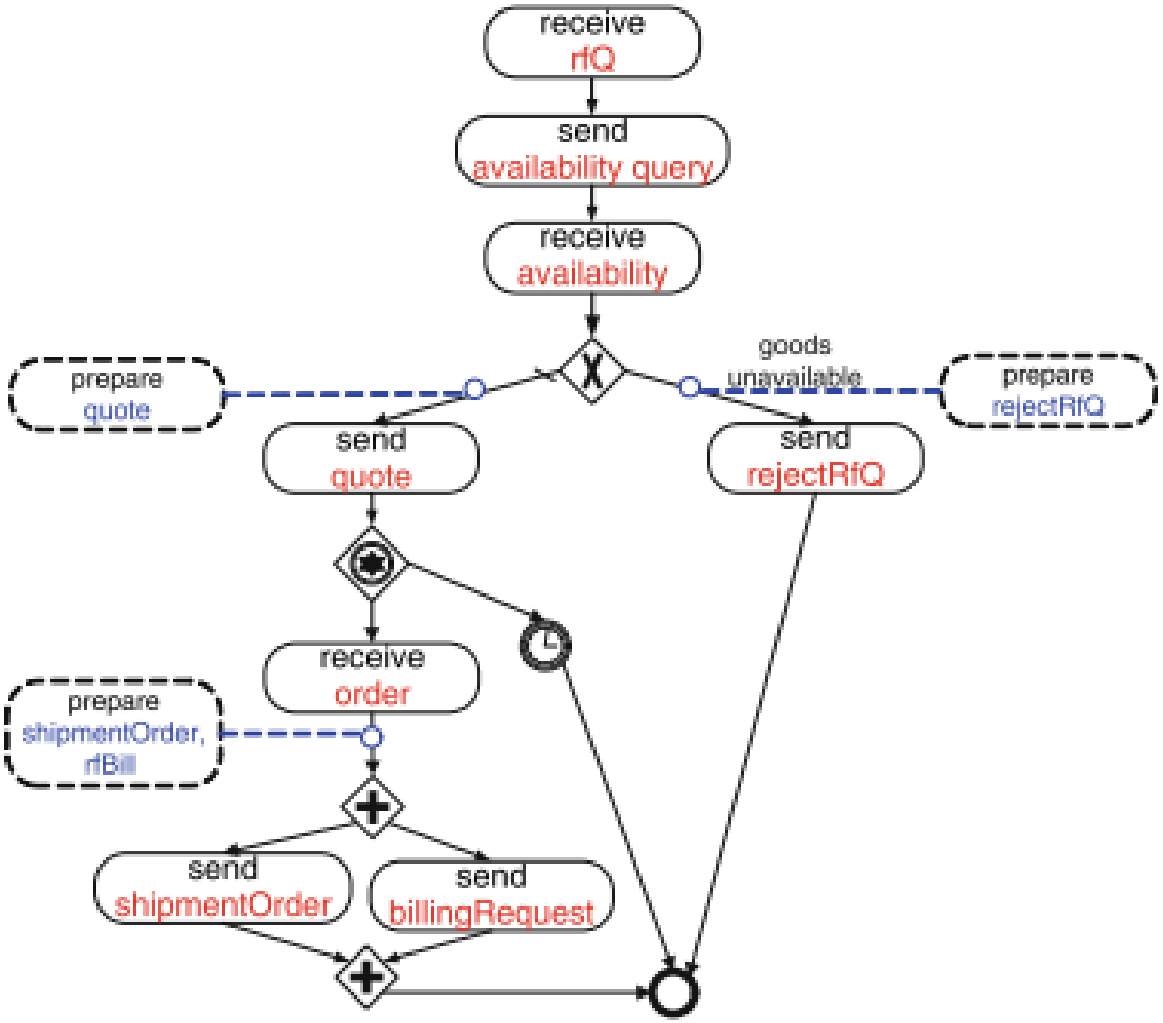
SERVICE ORCHESTRATION

- Refinement of a behavioural interface
- Includes the interactions plus internal actions that a service must perform
- Lower-level and focused-view of a service-enabled process
- Can be further refined into an executable BP e.g. in BPEL

SERVICE ORCHESTRATION



SERVICE ORCHESTRATION EXAMPLE



RECOMMENDED READING

- J. Vom Brocke and M. Rosemann. Handbook on Business Process Management 1 – Introduction, Methods, and Information Systems. Springer-Verlag, 2010.
- C. Rolland, M. Kirsch-Pinheiro and C. Souveyet. An Intentional Approach to Service Engineering. IEEE Transactions on Service Computing 3(4), 292-305, 2010
- K. Kritikos, S. Kubicki and E. Dubois. Goal-based business service composition. Service Oriented Computing and Applications (SOCA) 7(4), 231-157, 2013.
- https://www.youtube.com/watch?v=L1tM0tMJdzY
- https://www.youtube.com/watch?v=ukU6TyXOMv0&t=36s
- https://www.youtube.com/watch?v=bPNfu0IZhoE