OLYSYS
Technologies Integrator

Sea-Freight Automation Framework (SAF)

Alex Goussiatiner, P.Eng.

SAF Open Source Initiative

SAF is an open source initiative which started in March, 2018 . It aims to create standard automation framework and API for maritime logistics. The initiative offers a broader cross industry spectrum shifting away from today's current inefficient practices and human control to *efficient machine control processes*.

When adopted by the maritime industry, it should not only drastically increase the level of automation and reduce IT cost but also provide cross-platform interaction to all factions involved in cargo handling operations and logistics.

www. saf-project.org

Current Supply Chain Information Technology

- Human Control Supply Chain Processes, Manual Task Execution and Data Re-Entry
- Large Monolithic Systems
- Process Information "Silos“
- Incommensurable User Interfaces
- No standard definition for business roles, engagements and transactions
- No Standard Application Programming Interface (APIs)

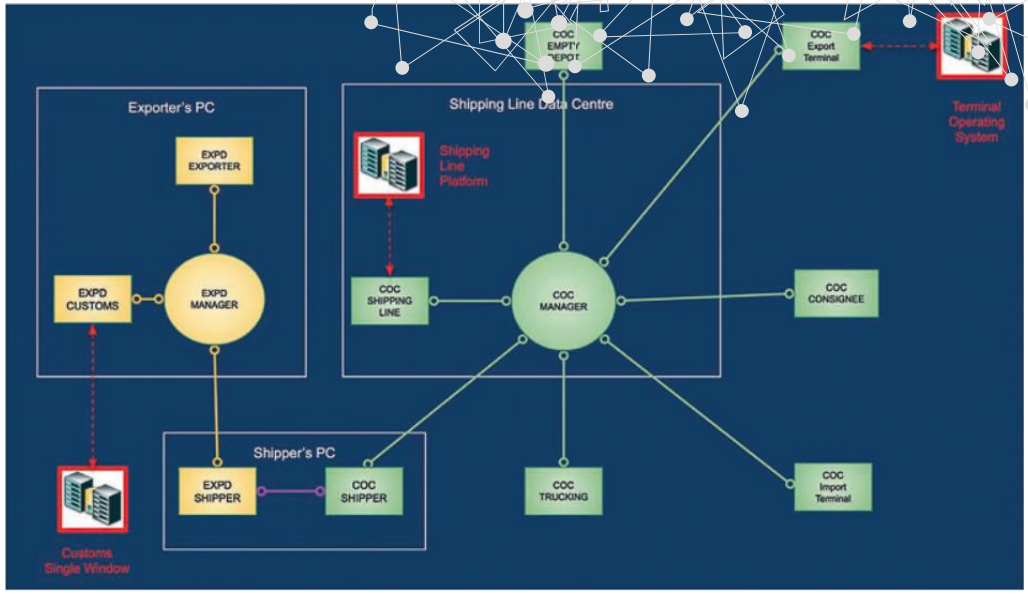
These factors, when combined, result in:

- A low level of process automation;
- High implementation and maintenance cost for the IT Platforms;
- Additional cost for stakeholders for usage of commercial integration platforms.

SAF

SAF decomposes logistics processes into parts - engagement processes and describes them. It introduces standard roles, engagement processes, messages and transactions as well as standard APIs for role services.

The role services (R-Service) will automate repetitive tasks performed now by humans. The services will also break the information silos - by passing information from one process to another. The services will be implemented as independent micro services or will be created by adding API gateway to the existing platforms.



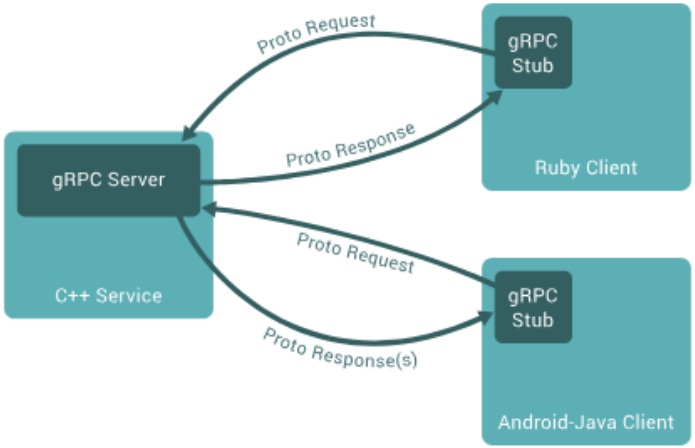
SAF Objects and APIs

Standard roles, engagements, messages, transactions are SAF Objects. Formal definitions of the objects (FD) are written using Proto 3 Data Definition Language (https://developers.google.com/protocol-buffers/docs/proto3)and published in the GitHub (https://github.com/saf-project/api).

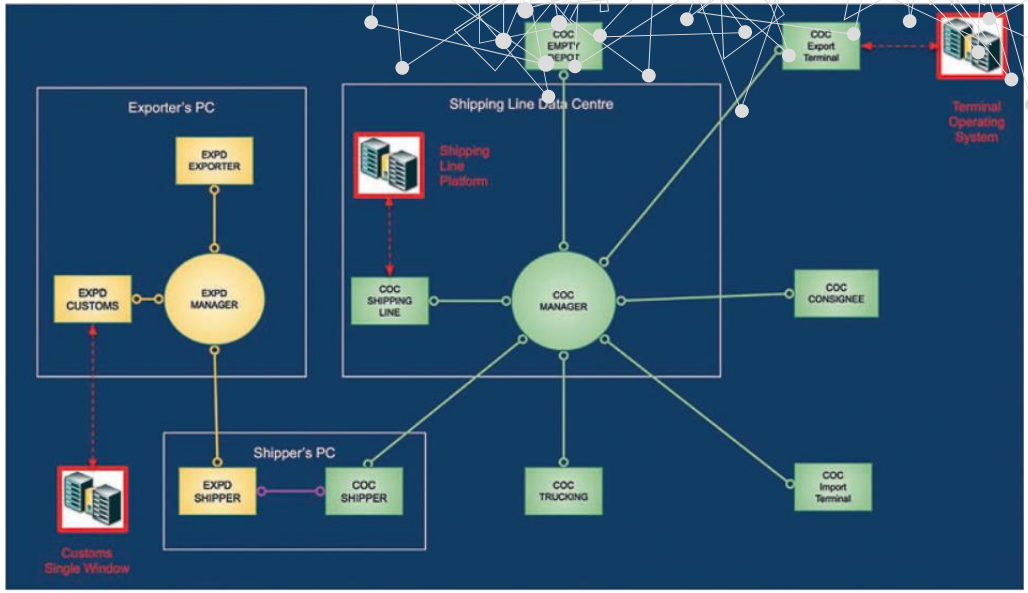
The FD of those objects are used by the role services for creation, validation and storage of the data. SAF also uses FD of the objects to describe the role of participants, transactions and messages used in the engagement processes.

Role service APIs describe remote methods exposed by the roles services and messages exchanged among services. The APIs are also written in Proto 3 and published in the GitHub.

For the communication role services use gRPC protocol (https://grpc.io/). SAF APIs are used as input to automatically generate idiomatic gRPC client and server stubs in a variety of programming languages including Java, GO, C++,Node.js, Ruby.



SAF Engagements



Engagement is a formal or informal agreement between SAF participants aiming to achieve certain well defined business objective such as:

- Transportation of a container from a port of loading to a port of discharge and passing it over to consignee;
- Port Vessel Call;
- Engagement Process (E-Process) is a business process with the aim to execute an Engagement.

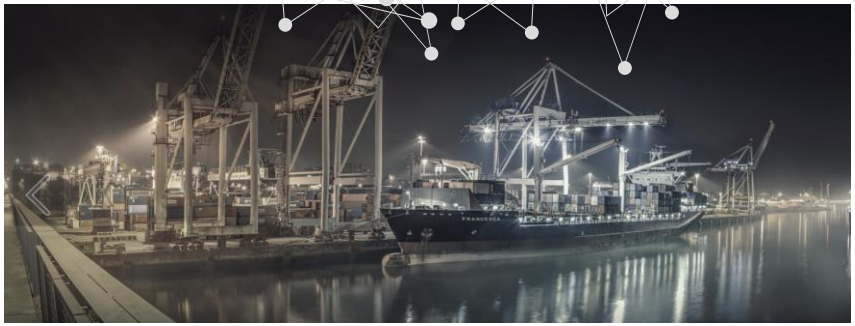
Most of the engagement processes end with the payment for the services performed.
SAF defines types of the standard Engagements: Export Customs Declaration Engagement, Empty Container Release Engagement, etc.
For each engagement type SAF defines participant roles and transactions to be executed by participants.

Engagement Definition Sample

```
syntax = "proto3";
package saf.shipping.engagements;
option go_package = "github.com/gosaf/saf/shipping/engagements";
import "saf/base/Engagement.proto";
import "saf/shipping/participants/exporter.proto";
import "saf/shipping/participants/ShippingLine.proto";
import "saf/shipping/participants/shipper.proto";
import "saf/shipping/participants/consignee.proto";
import "saf/shipping/transactions/ShipperLetterInstructionTxn.proto";
import "saf/shipping/transactions/GetCarriageRateTxn.proto";
import "saf/shipping/transactions/CarriageRateTxn.proto";
import "saf/shipping/transactions/ExportBookingTxn.proto";
import "saf/shipping/transactions/BookingAcknowledgementTxn.proto";
import "saf/shipping/transactions/ShippingInstructionTxn.proto";
import "saf/shipping/transactions/EmptyReleaseTxn.proto";
import "saf/shipping/transactions/VGMTxn.proto";
import "saf/shipping/transactions/OriginalBillLadingTxn.proto";
import "saf/shipping/transactions/FreightPaymentTxn.proto";
import "saf/shipping/transactions/ArrivalNoticeTxn.proto";
import "saf/shipping/transactions/EmptyReturnTxn.proto";

message ContractCarriagePortToPortStraightEng{
    saf.base.Engagement engagement=1;
    //accounts
    saf.shipping.participants.Exporter exporter=2;
    saf.shipping.participants.Shipper shipper=3;
    saf.shipping.participants.ShippingLine shippingLine=4;
    saf.shipping.participants.Consignee consignee=5;
    // transactions
    repeated  saf.shipping.transactions.ShipperLetterInstructionTxn  shipperLetterInstructionTxn=6;
    repeated  saf.shipping.transactions.GetCarriageRateTxn  getCarriageRateTxn=7;
    repeated  saf.shipping.transactions.CarriageRateTxn  carriageRateTxn=8;
    repeated  saf.shipping.transactions.ExportBookingTxn  exportBookingTxn=9;
    repeated  saf.shipping.transactions.BookingAcknowledgementTxn  bookingAcknowledgementTxn=10;
    repeated  saf.shipping.transactions.ShippingInstructionTxn  shippingInstructionTxn=11;
    repeated  saf.shipping.transactions.EmptyReleaseTxn  emptyReleaseTxn=12;
    repeated  saf.shipping.transactions.VGMTxn  VGMTxn=13;
    repeated  saf.shipping.transactions.OriginalBillLadingTxn  originalBillLadingTxn=14;
    repeated  saf.shipping.transactions.FreightPaymentTxn  freightPaymentTxn=15;
    repeated  saf.shipping.transactions.ArrivalNoticeTxn  arrivalNoticeTxn=16;
    repeated  saf.shipping.transactions.EmptyReturnTxn  emptyReturnTxn=17;
}
```

API Definition Sample



```
syntax = "proto3";
package saf.shipping.services;

import "saf/shipping/messages/EDShipperConfirmation.proto";
import "saf/shipping/messages/ShipperLetterInstruction.proto";
import "saf/shipping/messages/CarriageRate.proto";
import "saf/shipping/messages/BookingAcknowledgement.proto";
import "saf/shipping/messages/EmptyRelease.proto";
import "saf/shipping/messages/OriginalBillLading.proto";
import "saf/util/ack.proto";

option go_package = "github.com/gosaf/saf/shipping/services";

service ShipperService{
        rpc EDShipperConfirmationTxn(saf.shipping.messages.EDShipperConfirmation) returns  (saf.util.Ack);
        rpc ShipperLetterInstructionTxn(saf.shipping.messages.ShipperLetterInstruction) returns  (saf.util.Ack);
        rpc CarriageRateTxn(saf.shipping.messages.CarriageRate) returns  (saf.util.Ack);
        rpc BookingAcknowledgementTxn(saf.shipping.messages.BookingAcknowledgement)   returns  (saf.util.Ack);
        rpc EmptyReleaseTxn(saf.shipping.messages.EmptyRelease)     returns  (saf.util.Ack);
        rpc OriginalBillLadingTxn(saf.shipping.messages.OriginalBillLading) returns  (saf.util.Ack);

}
```


Paradigm Shift

Current State	SAF
Human Control Supply Chain Processes, Manual Task Execution and Data Re-Entry.	Machine Control Supply Chain with process flow automation spanning over all supply chain participants with less dependency on human availability and efficiency. No data re-entry.
Large Monolithic Platforms	Well defined granular functional services (applications), integrated when required with existing platforms.
Process "Silos".	Full integration between processes.
Incommensurable User Interfaces for online services and multiple sign-on.	Unified user interfaces for all online services, single sign-on.
No standard definitions for participant roles, engagement processes , messages and transactions.	Standard definitions for the participant roles, engagement processes , messages and transactions written in Proto 3 data definition language.
No Standard Application Programming Interface (APIs).	Standard APIs written in Proto 3, which describe remote methods exposed by the role services.
Use of Multiple Internet Communication Protocols.	gRPC Protocol - cross-platform, multi-language, open source protocol (initially developed at Google).
Not all participants have persistent online services on the Internet.	Online presence of all supply chain participants.

Hamad Port

DOHA, QATAR

The first of the three container terminals is currently operational and has an optimum capacity of 2 million TEUs per year, eventually increasing to over 7.5 million. This will then be supported by transhipment links by rail, sea and road to the region.

Hamad Port covers 28.5 square kilometres and will have the yearly capacity for 1.7 million tones of general freight and 1 million tones of grain, with a specialist terminal supporting the 500,000 vehicles per year.

Master Systems Integration Consultant

- **Tender Specifications**
- **Conceptual Design**
- **Design, Testing and Commissioning Management**

Hamad Port

مينـاء حمـد
HAMAD PORT

PORT (TERMINALS) OPERATIONS	PORT SECURITY	FACILITIES AND INFRASTRUCTURE
• Port Management Information System • Port Community System • Automated Gate Management System • Terminal Operating System • ERP • Enterprise Asset Management System • Enterprise Service Bus • Vehicle Booking System • Geographical Information System	• Incident and Resource Management System • Maritime Intrusion Detection	• Integrated Building Management System • Data Centre • Operations Control Rooms • Wireless Network • Outside Plant Cabling System • Local Area Network • Cyber Security Systems

