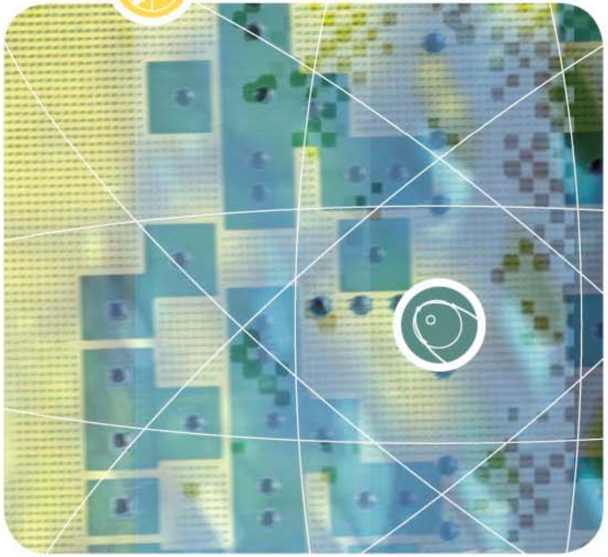
FFI Forsvarets forskningsinstitutt

Web Services and XML Security in an Operational Experiment

Paper 19
14th ICCRTS
June 15th, 2009

Trude Hafsøe
trude.hafsoe@ffi.no



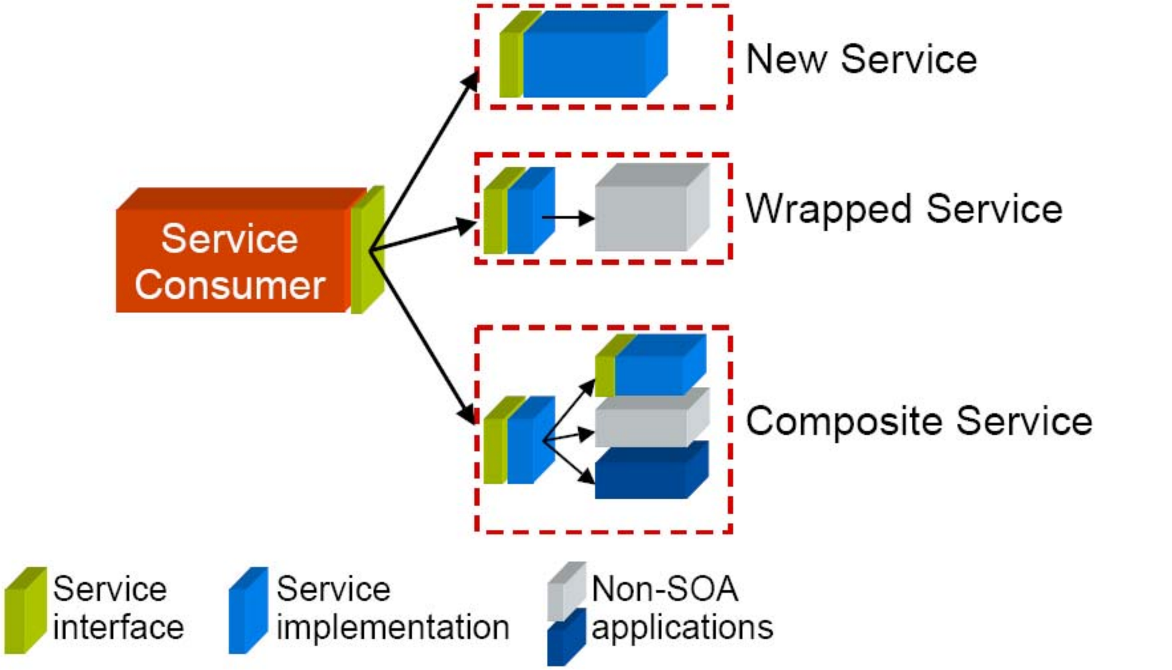
Outline

- SOA and Web services in two minutes
- Experiment goals
- Cooperative ESM Operation
- Experiment execution
- Experiment results and lessons learned

Service Oriented Architecture

"A service is a mechanism to enable access to resources, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description."

(OASIS: Reference Model for Service Oriented Architecture 1.0).



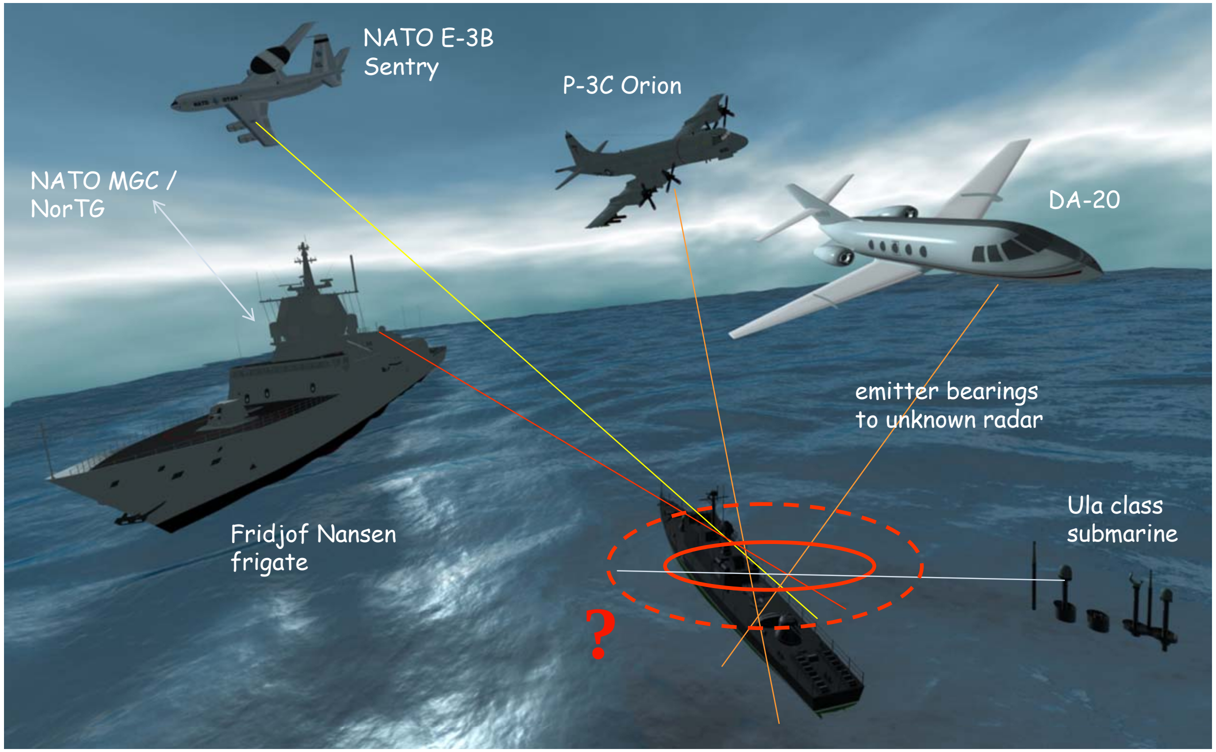
Gartner Research "Service-Oriented Architecture Under the Magnifying Glass" by Yefim Natis, Application Integration & Web Service, Summit 2005, April 18-20, 2005

Experiment Goal and Setting

- Large national experiment (late 2008)
  - interconnections between all the military services, some tried before and some new
  - large number of trials

- Using Web services in an operational setting
  - Proof-of-concept/feasibility test
  - Demonstrate how
    - Web services can function as an integrator,
    - use of subscriptions and automatic service discovery reduce the need for manual configuration,
  - Investigate the amount of overhead XML security standards introduce

Cooperative ESM Operations (CESMO)

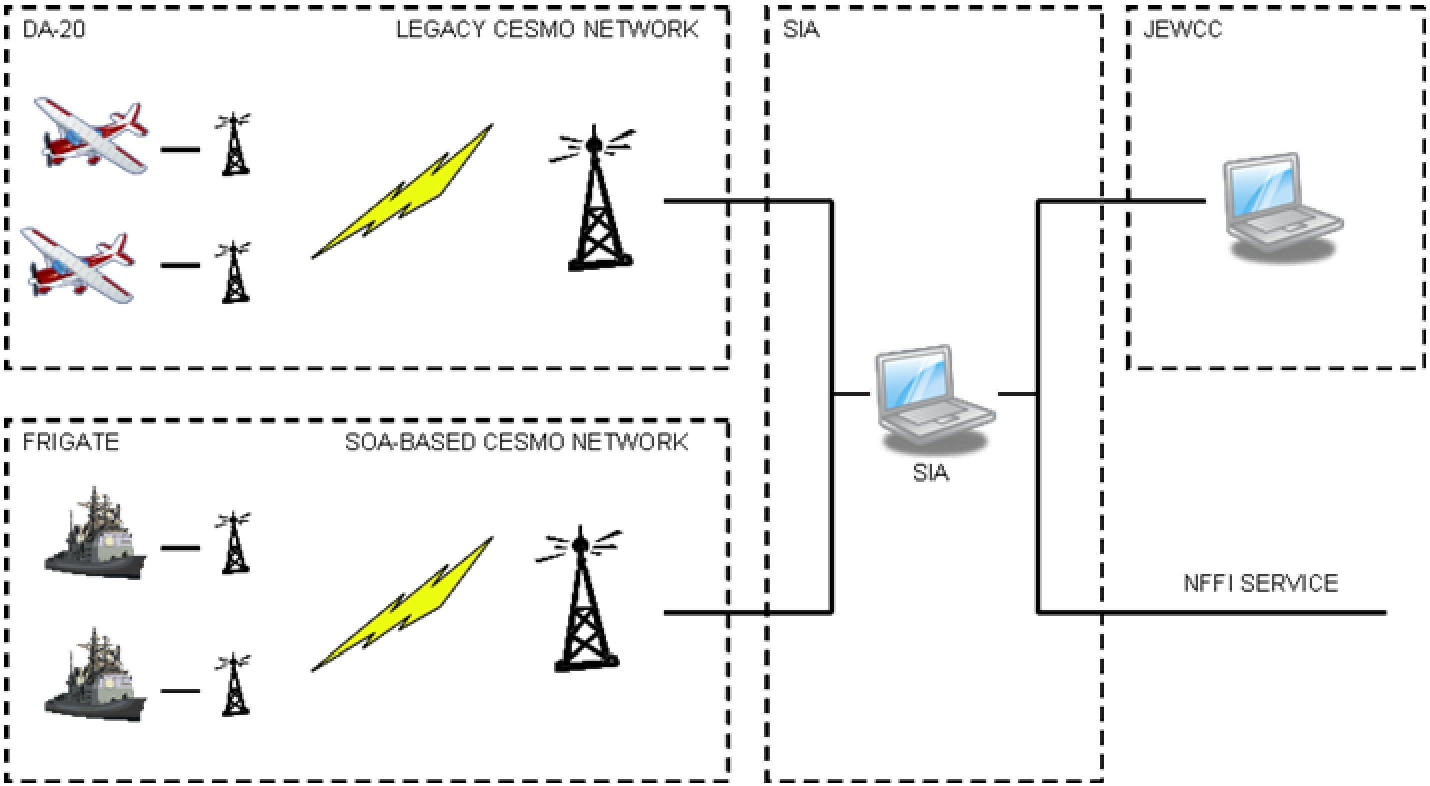


- An ESM sensor platform can have two roles
  - ordinary sensor platform
  - SIA, which coordinates observations and calculates the geolocations of observed emitters

The CESMO experiment

- Experiment participants
  - two air force sensor platforms
  - two navy sensor platforms
  - the SIA
  - a coordination cell
    - understands the ESM data format
    - wanted access to the ESM bearing as reported by the sensors
  - a C2 system
    - does not understand the ESM data format
    - wanted geolocations of observed emitters in NFFI format
- SOA-enabled CESMO platforms
  - through a self made web service front-end
  - uses an existing experimental middleware for publish/subscribe

Planned network setup



SOA as an integrator

- We used Web services to integrate systems that would otherwise not be able to share information by
  - Wrapping the legacy CESMO software
    - In the navy network we used our software to wrap the software on each platform
    - The air force nodes could not be wrapped individually, the solution was to wrap the entire network
  - Making information from both these networks available to other systems through new services
    - previously separate CESMO systems were able to share information
    - outside systems could benefit from the information by receiving geolocations of emitters

Publish/subscribe

- Interested parties subscribe to the information they are interested in
  - more fine-grained control of information flow
  - nodes only receive information that they have expressed an interest in
  - information is only sent onto the network if someone is interested in it
  - network traffic is only generated when new information is available, without the need for polling (less network traffic)
  - messages can be multicasted to interested parties, thus saving further on network resource usage

Notification Message Example

```
<?xml version="1.0" encoding="utf-8"?><S:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:wa48="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2002/12/policy"
xmlns:S="http://schemas.xmlsoap.org/soap/envelope/"><S:Header><wa48:To>http://193.156.33.75:18888</wa48:To><wa
48:MessageID>test</wa48:MessageID><wa48:Action>test</wa48:Action></S:Header><S:Body><extreme:StringNotificatio
n
xmlns:extreme="http://extreme.indiana.edu">PHdzbWdjbGllbnQgdG9waWM9J1BsYXRmb3JtVHlwZSc+dGVzdDwvd3NtZ2
NsaWVudD4=</extreme:StringNotification></S:Body></S:Envelope>
```

- Notification message without security
  - Envelope is left untouched
  - Body is compressed
  - Payload is shown in red
  - Message size depends on size of payload

- Subscription messages are fixed in size

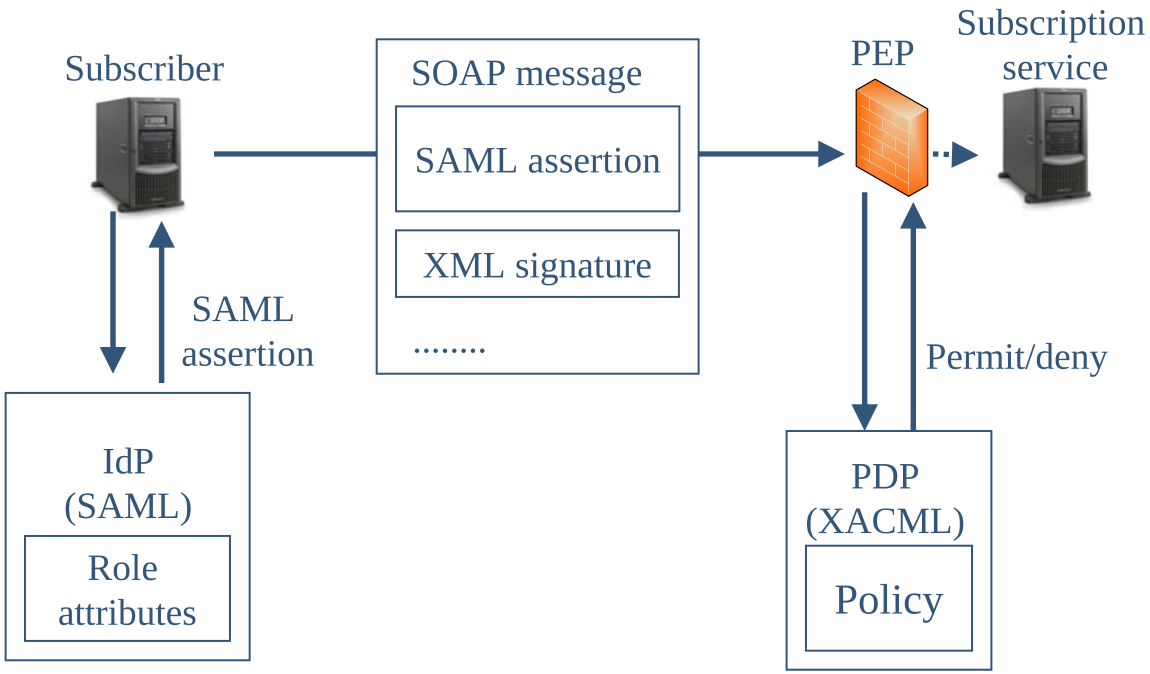
Automatic Service Discovery

- The platforms discover each other without manual configuration
- The list of available services is presented to the CESMO operators, allowing them to chose which services to subscribe to

- UdpDiscovery, a custom Java library
  - Optimized for disadvantaged grids
  - Compression of the objects was used to ensure compactness
  - Gives all nodes updated information about the network
  - Network usage
    - Exchange of information between nodes in the network at regular intervals
    - Each such update is approximately 500 bytes
    - One message pr node pr minute

XML Security

- Subscription requests were subject to role based access control
    - SAML
- Messages were subject to end-to-end integrity protection using XML signature
    - XACML



XML Security Overhead

	Subscription Message	Notification Message Envelope	Example Notification Message (compressed body)
Size	985 bytes	584 bytes	652 bytes
Size w/security	5074 bytes	2509 bytes	2577 bytes

Network usage summarized

- Communication based on UDP multicast
  - replaces the standard HTTP/TCP binding for Web services
  - more resistant to long communication delays
  - less communication overhead
- Service discovery sends small messages at regular intervals
- Subscription messages are fixed in size
- The size of a notification is dependent on the payload
- The use of XML security standards increase the size of the message significantly
  - compression and potentially removal of optional information is needed to allow the use of these standards in bandwidth constrained networks

Summary

- Illustrates the added value of SOA and shows that it can be applied in an operational network
  - allow legacy software to share information, and offer several new services based on this information
  - less manual configuration, more fine-grained control of information flow
- Compressed XML messages
  - data exchange of the SOA system was comparable to that of the standard CESMO
  - we got a lot of added value without introducing any significant overhead
- An opportunity to show the flexibility of SOA
  - the need for ad hoc reconfiguration of the network did not prevent our SOA software from functioning
  - adapt to changing condition at runtime by changing the dissemination of information between nodes (establishing and terminating subscriptions when needed)
- Testing Web services with real data under real workloads was a benefit; previously we had verified its functionality in our lab environment. During the experiment we saw that the software could handle the data and usage patterns of an operational system