

# Service Compositions: Curse or Blessing for Security?

Achim D. Brucker  
 achim.brucker@sap.com  
 http://www.brucker.ch/

SAP AG, Products & Innovation, Products and Innovations, Product Security Research  
 Vincenz-Priessnitz-Str. 1, 76131 Karlsruhe, Germany

2nd International Workshop on Behavioural Types  
 September 23-24 2013, Madrid, Spain

## Abstract

Building large systems by composing reusable services is not a new idea, it is at least 25 years old. Still, only recently the scenario of dynamic interchangeable services that are consumed via public networks is becoming reality. Following the *Software as a Service* (SaaS) paradigm, an increasing number of complex applications is offered as a service that themselves can be used composed for building even larger and more complex applications. This will lead to situations in which users are likely to unknowingly consume services in a dynamic and ad hoc manner. Leaving the rather static (and mostly on-premise) service composition scenarios of the past 25 years behind us, dynamic service compositions, have not only the potential to transform the software industry from a business perspective, they also requires new approaches for addressing the security, trustworthiness needs of users. The EU FP7 project Aniketos develops new technology, methods, tools and security services that support the design-time creation and run-time dynamic behaviour of dynamic service compositions, addressing service developers, service providers and service end users. In this talk, we will motivate several security and trustworthiness requirements that occur in dynamic service compositions and discuss the solutions developed within the project Aniketos. Based on our experiences, we will discuss open research challenges and potential opportunities for potential opportunities for applying type systems.

## About Me

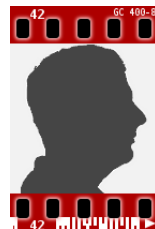
### My Employer: SAP AG



- Vendor of enterprise software systems
- World's third largest software vendor
- More than 25 industries
- 63% of the world's transaction revenue touches an SAP system
- 64 422 employees worldwide

### Personal Background:

- Senior Researcher: Security, Formal Methods, Software Engineering
- Security Expert: supporting all phases of a SDLC



## The Aniketos Project

Enable composite services to establish and maintain security and trustworthiness

### Goals of the Aniketos platform:

- Design-time discovery, composition and evaluation, threat awareness
- Runtime adaptation or change in service configuration
- Runtime monitoring, detection, notification

### Aniketos Fact-Sheet:

- EU Integrated Project (IP), FP7 Call 5
- Budget: € 13.9 Mio (€ 9.6 Mio funding)
- 42 month (Aug. 2010 – Feb. 2014)
- Coordinator: Sintef (Norway)

### Two related dimensions:

- **Trustworthiness:** Reputation, perception, centralised vs. distributed
- **Security properties:** Behaviour, contracts, interfaces, formal verification

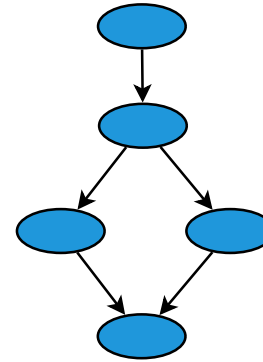


# Outline

- 1 (Secure) Service Composition: Past, Present, and Future
- 2 The Aniketos Approach: Overview
- 3 The Aniketos Approach: Exemplary Deep Dive
- 4 Service Compositions: A Curse or Blessing for Security?

# The Past: Service Compositions

“ **Service:** a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface ... with constraints and policies as specified by the service description.  
 OASIS Reference Model for Service Oriented Architecture



- At least 20 years old:
  - RPCs introduced in 1980s
  - CORBA published in 1991
- Motivated by
  - re-useability
  - reliability
- Used within organisations
- Frameworks considered to be *heavy-weight*

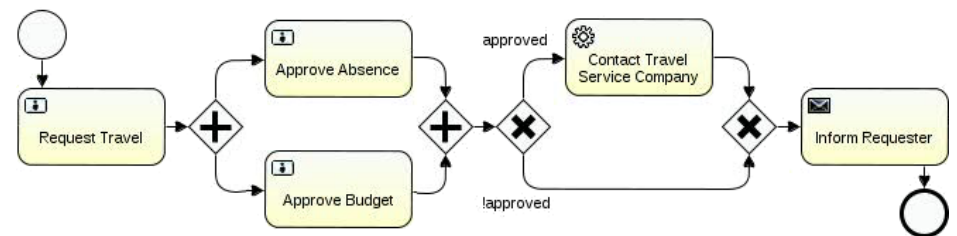
# The Past: Secure Service Compositions



Photo: Holger Weinandt

- Recall the past:
  - networks were expensive (and slow)
  - only a few people had access to networked systems
- Security model:
  - non-technical trust
    - small numbers of users allowing a personal relationship
    - system operators trust their users
  - security perimeter
    - limited access
    - firewalls
    - controlled system access

# The Present: Service Composition



- Motivated by business needs:
  - cost-savings
  - flexibility
- Used across organisations
- Environment
  - fast networks
  - many users
- relatively static compositions

# The Present: Secure Service Composition

Access Control



Photo: Syohei Arai

## Goal:

- Control access to services, resources (data), ...

## The core:

- Usually: users, roles, access rights, ...
- In special cases: data labelling or information flow

## On top:

- Separation of Duty
- Binding of Duty
- Delegation

# The Present: Secure Service Composition

Protecting Data (and Physical Goods)



Photo: Bundesarchiv, Bild 183-R0117-0003 / CC-BY-SA

## Goal:

- Ensure
  - confidentiality
  - integrity (safety) of data (and goods)

## The core:

- Need-to-Know
- Fingerprints
- Encryption
- Sensors

# The Present: Secure Service Composition

Compliance and Additional Requirements

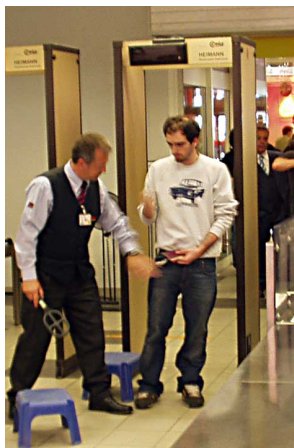


Photo: Ralf Roletschek

Many regulated markets

- Basel II/III, SoX, PCI
- HIPAA

Many customer-specific regulations

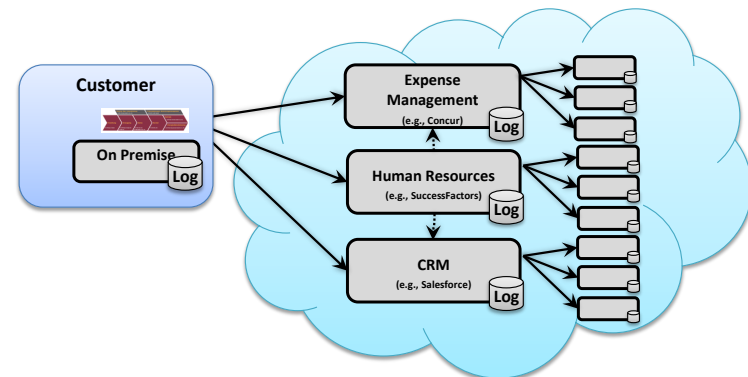
- Own governance to mitigate risks
- Own business code of conduct
- Fraud detection/prevention
- Non-observability

Customers are individually audited

- No "one certificate fits all" solution

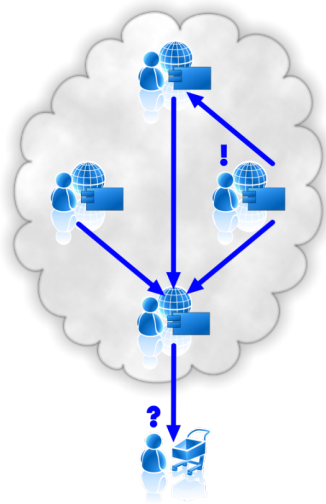
**Security should not hinder business**

# The Future: Service Composition



- Software as a service
  - complex components
  - updates controlled by provider
- Many external services
- No central orchestrator
- Complex data flows

# The Future: Secure Service Composition



We need to ensure the already discussed requirements.

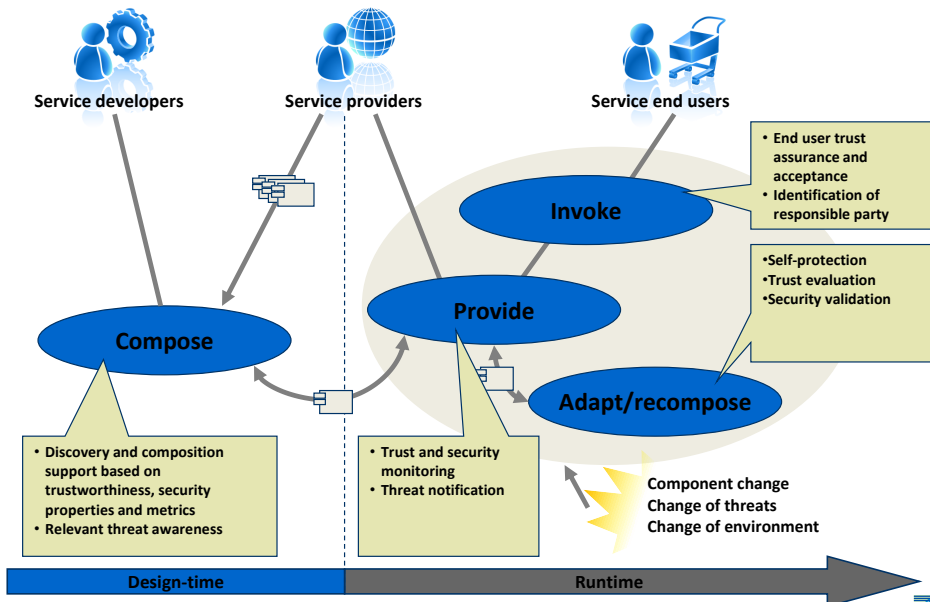
Many additional challenges, e.g.,

- Customer
  - ensure compliance in a changing environment
- Developer
  - provide secure, scalable services
- Provider
  - provide secure offerings
  - protect own infrastructure
  - protect data of customers

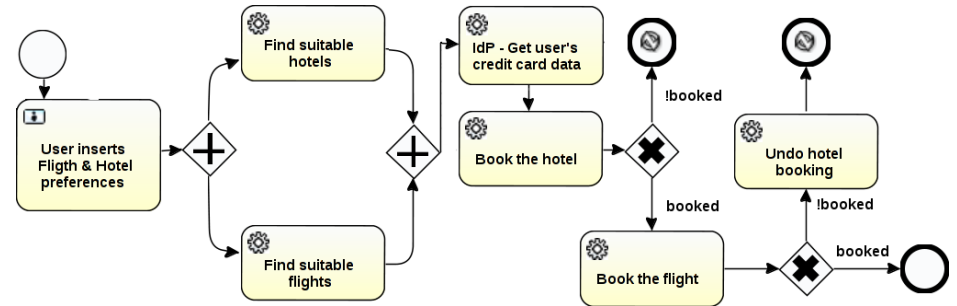
# Outline

- 1 (Secure) Service Composition: Past, Present, and Future
- 2 The Aniketos Approach: Overview
- 3 The Aniketos Approach: Exemplary Deep Dive
- 4 Service Compositions: A Curse or Blessing for Security?

# The Aniketos Process



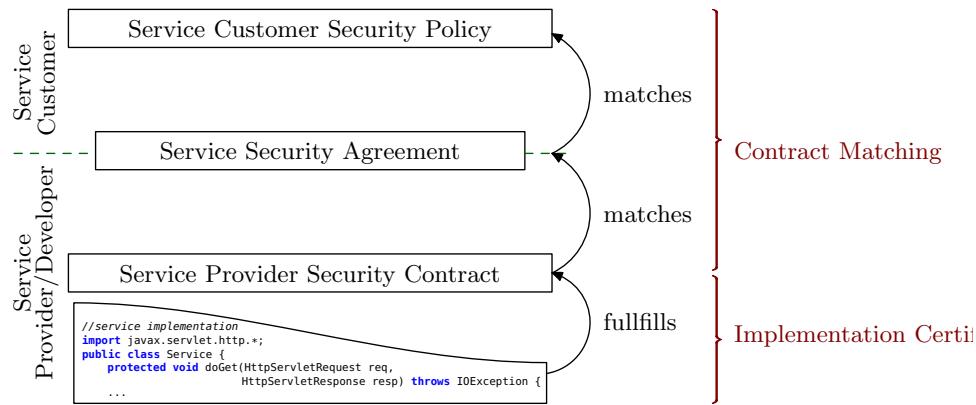
# Modeling Composition Plans using BPMN



- Human-centric tasks
- Automated tasks (services)
- Orchestration of services
- Start/end states
- Logical control flow (if/and/or)
- Error states

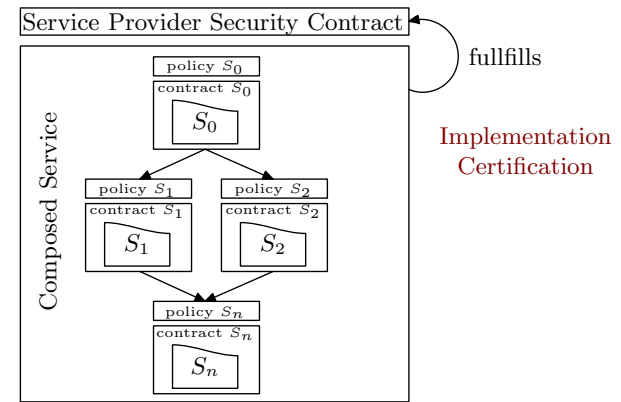
# Security by Contract

Atomic Services



# Security by Contract

Composed Services

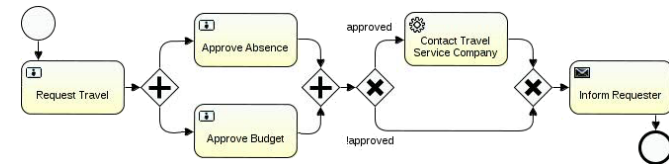


## Outline

- 1 (Secure) Service Composition: Past, Present, and Future
- 2 The Aniketos Approach: Overview
- 3 The Aniketos Approach: Exemplary Deep Dive
- 4 Service Compositions: A Curse or Blessing for Security?

## Secure Implementation of Atomic Services

Check the compliance of atomic service implementations.



- **Human tasks:**
  - Define the user interface (e.g., HTML, Java)
  - Create, read, or update of process variables
- **Service tasks:**
  - Define the business logic (e.g., Java, Web service specific configuration)
  - Create, read, or update of process variables



# Implementing "Contact Travel Service Company"

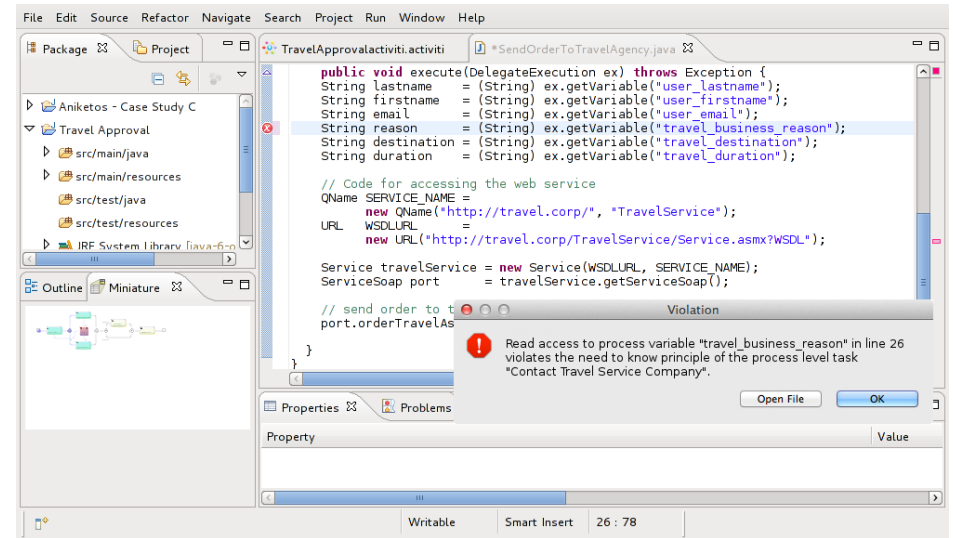
```

package corp.acme;
public class SendOrderToTravelAgency implements JavaDelegate {
    @Override
    public void execute(DelegateExecution ex) throws Exception {
        String lastname = (String) ex.getVariable("user_lastname");
        String firstname = (String) ex.getVariable("user_firstname");
        String email = (String) ex.getVariable("user_email");
        String reason = (String) ex.getVariable("travel_business_reason");
        String destination = (String) ex.getVariable("travel_destination");
        String duration = (String) ex.getVariable("travel_duration");

        // Code for accessing the web service
        QName SERVICE_NAME = new QName("http://travel.corp/", "TravelService");
        URL WSDLURL = new URL("http://travel.corp/TravelService/Service.asmx?WSDL");
        Service travelService = new Service(WSDLURL, SERVICE_NAME);
        ServiceSoap port = travelService.getServiceSoap();

        // send order to travel service
        port.orderTravelAssistance(firstname, lastname, email, reason,
            destination, duration);
    }
}
    
```

# Implementation Level Reasoning



# The Problem: RBAC with Separation of Duty

Role-based access control (RBAC)

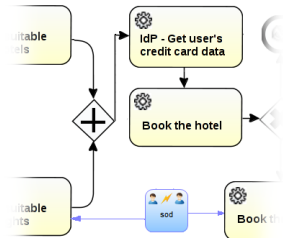
- Subjects are assigned to roles
- Permissions assign roles to tasks (resources)

Separation of duty (SoD)

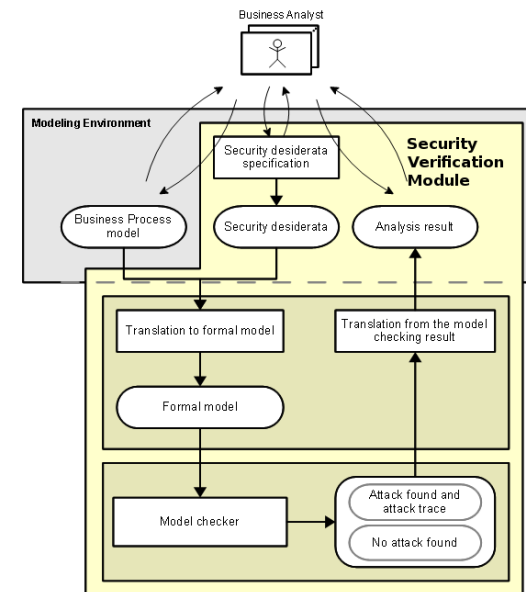
- restrict subjects in executing tasks

We analyse:

- Does the RBAC configuration comply to the SoD requirements?
  - yes: static SoD
  - no: dynamic SoD
- In case of a compliance violation:
  - change RBAC configuration
  - ensure dynamic enforcement of SoD



# Security Verification Module (RBAC/SoD Check)



## Analysing (Dynamic | Static) Separation of Duty

Does the access control enforce a separation of duty constraint

- Translate the composition plan to ASLan

```

hc rbac_ac(Subject, Role, Task) := CanDoAction(Subject, Role, Task)
    :- user_to_role(Subject, Role), poto(Role, Task)
hc poto_T6 := poto(Staff, Request Travel)
hc poto_T6 := poto(Manager, Approve Absence)
hc poto_T7 := poto(Manager, Approve Budget)
    
```

- Specify the test goal

```

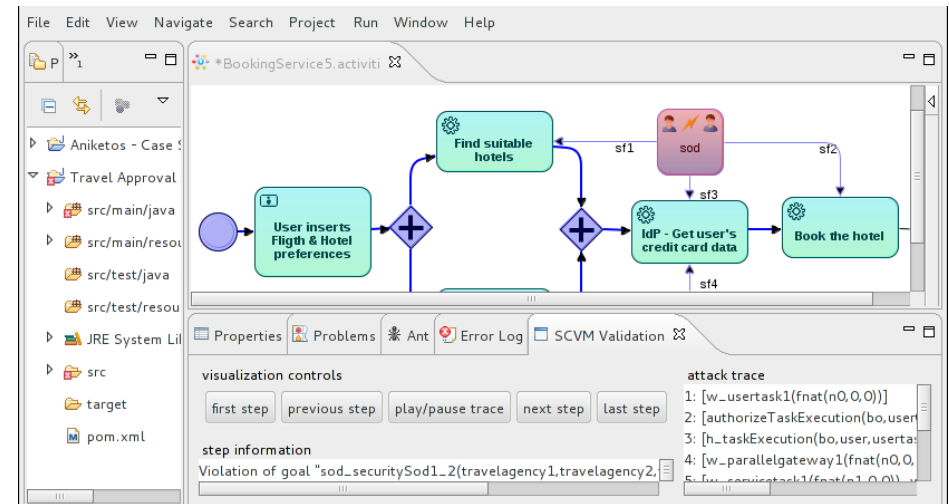
attack_state sod_securitySod1_1(Subject0,Subject1,Instance1,Instance2)
:= executed(Subject0,task(Request Travel,Instance1)).
   executed(Subject1,task(Approve Budget,Instance2)).
   executed(Subject3,task(Approve Absence,Instance3))
   &not(equal(Subject0,Subject1))
   &not(equal(Subject1,Subject2))
   &not(equal(Subject2,Subject3))
    
```

- Run the model checker
- Translate the analysis result back to BPMN (visualisation)

## Outline

- (Secure) Service Composition: Past, Present, and Future
- The Aniketos Approach: Overview
- The Aniketos Approach: Exemplary Deep Dive
- Service Compositions: A Curse or Blessing for Security?

## User Interface for the Service Designer



## Why Are Enterprises Moving Services to the Cloud

The shift to the cloud (SaaS) is driven by economical considerations

- total cost of ownership
- need for adaptability (flexibility)
- ...

Core assumption: specialised cloud providers

- operate systems cheaper (licenses, upgrades, etc.)
- achieve higher reliability
- provide more flexibility (elasticity, features, etc.)
- ...

And security?

# Security Chances and Risks of Service Compositions

## Chances

- regular updates
  - XSS, SQL injection, etc.
- system administration
  - misconfiguration
- secured data centres
  - specialises systems
- ...

## Challenges

- how to trust the provider
  - data disclosure
- new attacks
  - tenants as attackers
  - providers as attackers
- how to control delegation
  - subcontracting
- ...

# Challenges for Type-based Approaches (in Industry)

- Real systems are not build from scratch
  - existing frameworks
  - legacy systems
  - ...
- Developers hate to write type annotations

```
BufferedReader in = new BufferedReader(converter);
```

we need to advertise

- type inference,
  - a concise syntax for typed languages, and
  - helpful error messages
- Weakly/dynamically typed languages are gaining popularity
    - light-weight typed based analysis approaches
    - type systems for
      - well-defined subsets that
      - can interact with the whole language (libraries, etc.)

## Conclusion



The interesting challenges are still ahead of us!





- Real systems are large and complex:
  - many programming languages or frameworks
  - many security technologies
  - highly distributed
- There is a trend towards weakly typed languages
  - can we provide type-based analysis for such systems
  - can we provide (strongly) typed alternatives that
    - provide similar flexibility
    - can integrate existing frameworks
- Security is more than CIA
  - needs to be ensured on all levels
    - implementation level vulnerabilities
    - configuration errors
    - security frameworks/implementation (authentication, crypto)
  - business-level compliance
  - legal frameworks

# Thank you for your attention!

Any questions or remarks?



## References

-  Achim D. Brucker and Isabelle Hang.  
**Secure and compliant implementation of business process-driven systems.**  
In Marcello La Rosa and Pnina Soffer, editors, *Joint Workshop on Security in Business Processes (SBP)*, volume 132 of *Lecture Notes in Business Information Processing (LNBIP)*, pages 662–674. Springer-Verlag, 2012.
-  Achim D. Brucker, Isabelle Hang, Gero Lückemeyer, and Raj Ruparel.  
**SecureBPMN: Modeling and enforcing access control requirements in business processes.**  
In *ACM symposium on access control models and technologies (SACMAT)*, pages 123–126. ACM Press, 2012.
-  Achim D. Brucker, Francesco Malmignati, Madjid Merabti, Qi Shi, and Bo Zhou.  
**A framework for secure service composition.**  
In *ASE/IEEE International Conference on Information Privacy, Security, Risk and Trust (PASSAT)*. IEEE Computer Society, 2013.
-  Luca Compagna, Pierre Guillemot, and Achim D. Brucker.  
**Business process compliance via security validation as a service.**  
In Manuel Oriol and John Penix, editors, *IEEE International Conference on Software Testing, Verification and Validation (ICST)*, pages 455–462. IEEE Computer Society, 2013.