An Architecture for Developing Aspect-Oriented Web Services

Santokh Singh

Professor John Grundy

Professor John Hosking

Dr Jing Sun

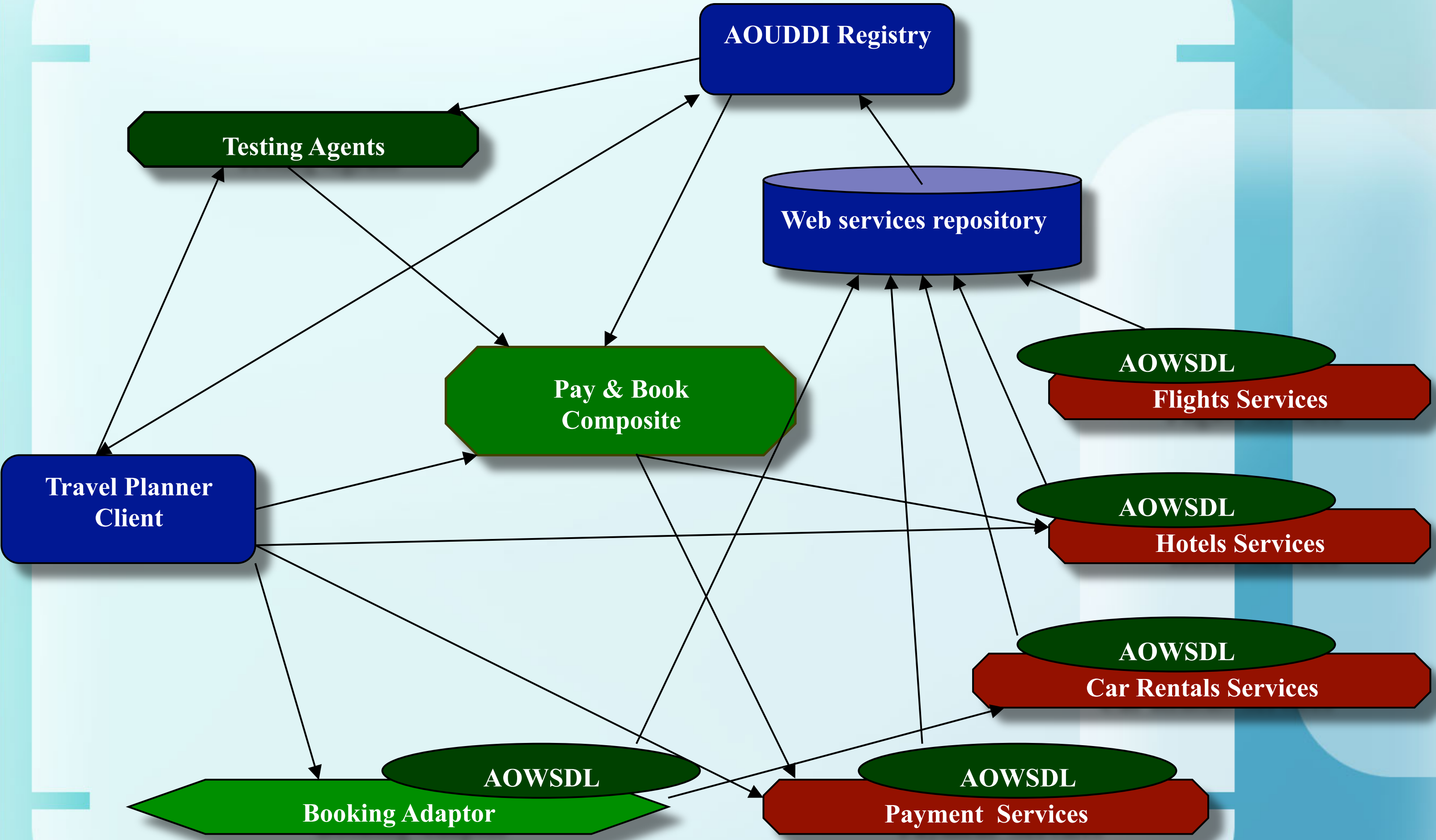
Computer Science Dept
University of Auckland
New Zealand

Outline

- Introduction & Motivation
- Background knowledge and our earlier works
  - Aspect-oriented Component Engineering
- Aspect-Oriented Web Services (AOWS) and AOConnector
- Specification of AOWS using Alloy
- AOWS Dynamic Behaviour
- Ongoing and Future Work
- Conclusions

Introduction

An Aspect-Oriented Web services (AOWS) system



Introduction

Why did we choose to use CBSE?

- **Improves software modularity, reuse, development efficiency.**

Why do we need better CBSE methodology?

- **Present component-based systems engineering focuses on low level software component interface design and implementation.**
- **Great setbacks –**
  - ➢ **Often results in development of components whose services are hard to understand and combine.**
  - ➢ **Makes too many assumptions about other components related to it.**
  - ➢ **Component documentation is too low level which is again hard to understand and use at higher levels.**

Motivation

1. **Web services promises dynamic application to application communication.**
2. **Current Web Services has limitations as regards:**
   - ➢ Description
   - ➢ Discovery
   - ➢ Integration and
   - ➢ Consumption.
3. **Factors that cause limitation to be clearly identified and urgently resolved**

Motivation *cont.*

We also need a better CBSE methodology

- **Low-level software component interface design and implementation - cumbersome and difficult to comprehend.**
  - **The larger the software system, the more prevalent and critical this problem becomes.**
- **Problem even in industries producing or refactoring the code for commercial software tools and systems.**
- **Leads to tremendous wastage in terms of time, effort and resources.**
- *A solution needs to be found.*
- **We propose Aspect-oriented Component Engineering (AOCE).**

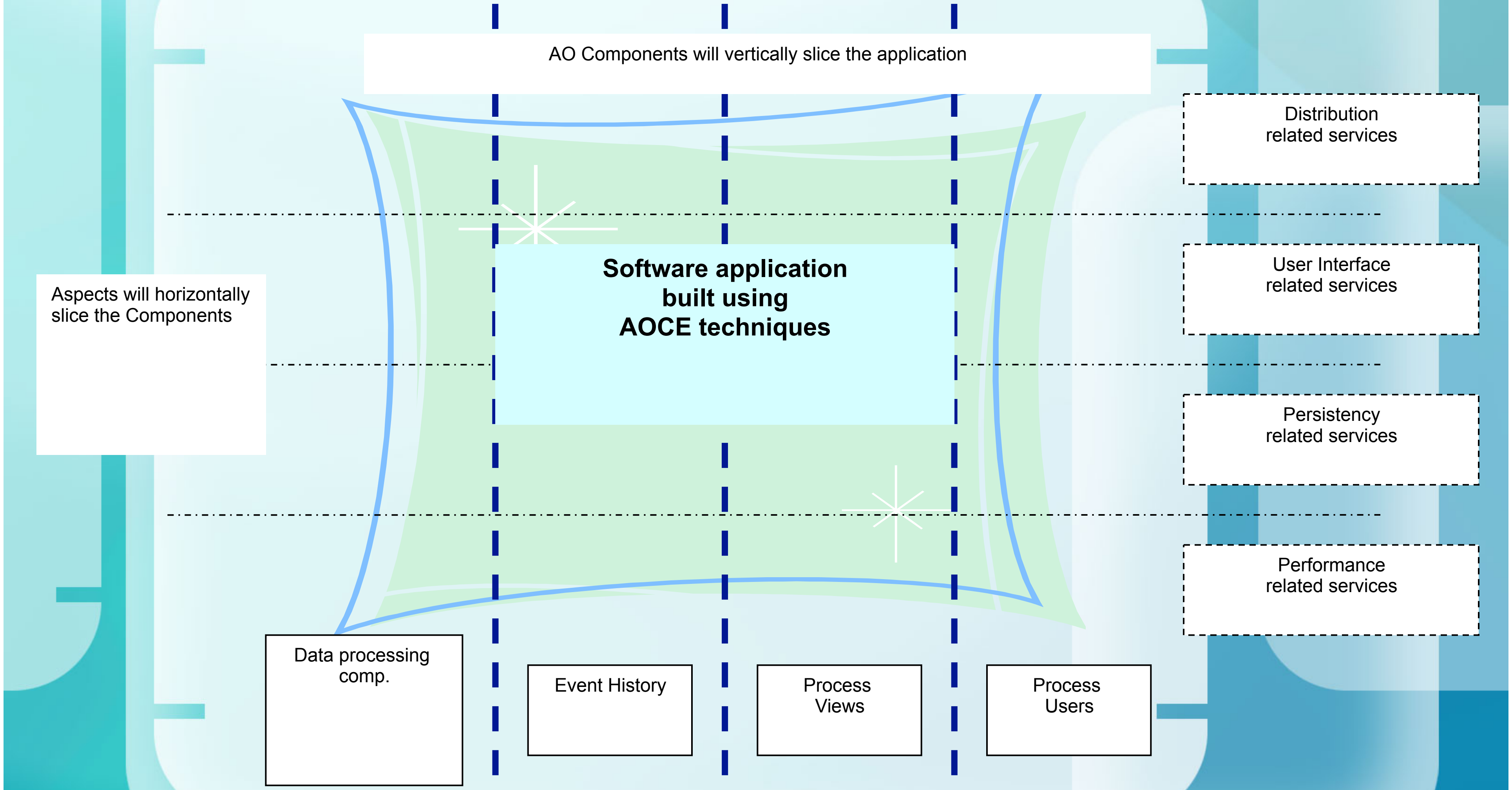
Aspect-oriented Component Engineering

- AOCE - a new component based software development methodology.
- Aims at enabling software engineers develop efficient, better and more reusable software components.
- AO-Components are better characterised and categorised.
- Aspect-oriented systems developed using AOCE are easier to maintain, and are more understandable and scalable.

AO-Components

- AO-Component main features:
  - It constitutes an independent and replaceable part of a aspect-oriented software system and has a clear function to fulfil.
  - It works within the context of well defined aspect-oriented software architecture.
  - It communicates with other ao-components through its interface.
  - It is highly categorised and characterised with aspectual information.

AOCE Software



AO Components will vertically slice the application

Aspects will horizontally slice the Components

Software application built using AOCE techniques

Distribution related services

User Interface related services

Persistency related services

Performance related services

Data processing comp.

Event History

Process Views

Process Users

Aspects in AOCE

- Aspects horizontally slice the overall software system that was vertically componentised.

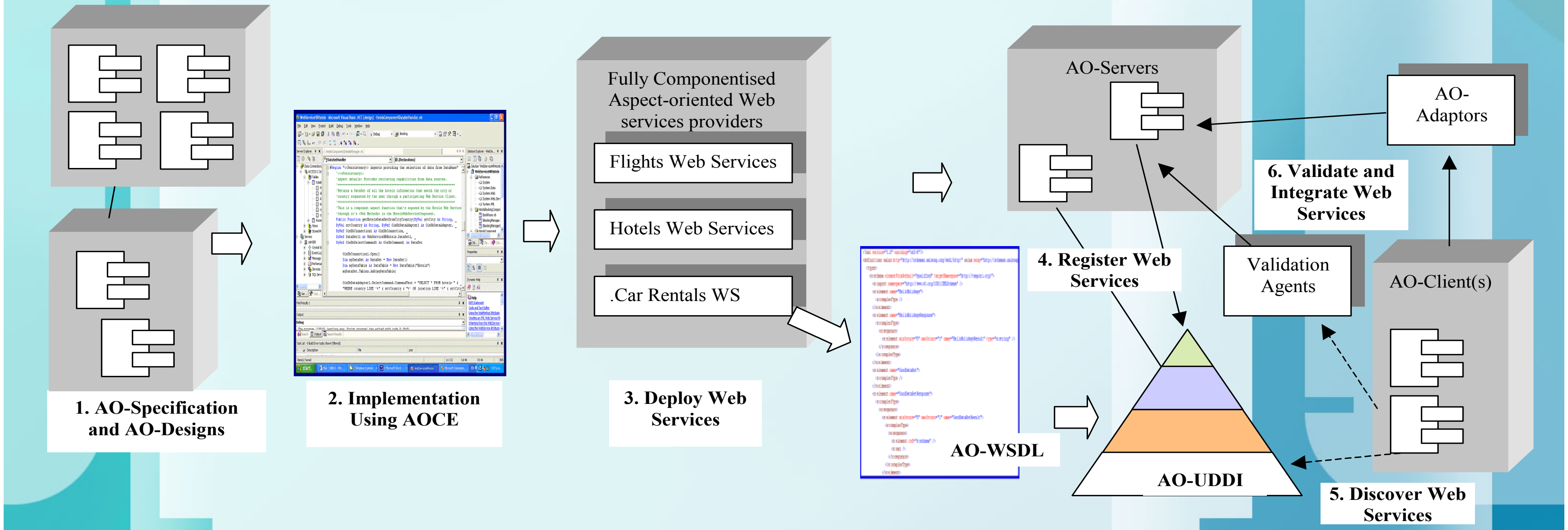
- Aspects characterize specific cross-cutting functional and non-functional properties of the components.

- Examples of Aspects:

  security, persistency, configuration, collaboration, transaction processing, distribution, user interface, performance and resource utilization.

AOCE

- Overview of using AOCE to develop web service-based software systems



1. AO-Specification and AO-Designs

2. Implementation Using AOCE

Fully Componentised Aspect-oriented Web services providers

Flights Web Services

Hotels Web Services

.Car Rentals WS

3. Deploy Web Services

AO-WSDL

AO-Servers

4. Register Web Services

AO-UDDI

AO-Adaptors

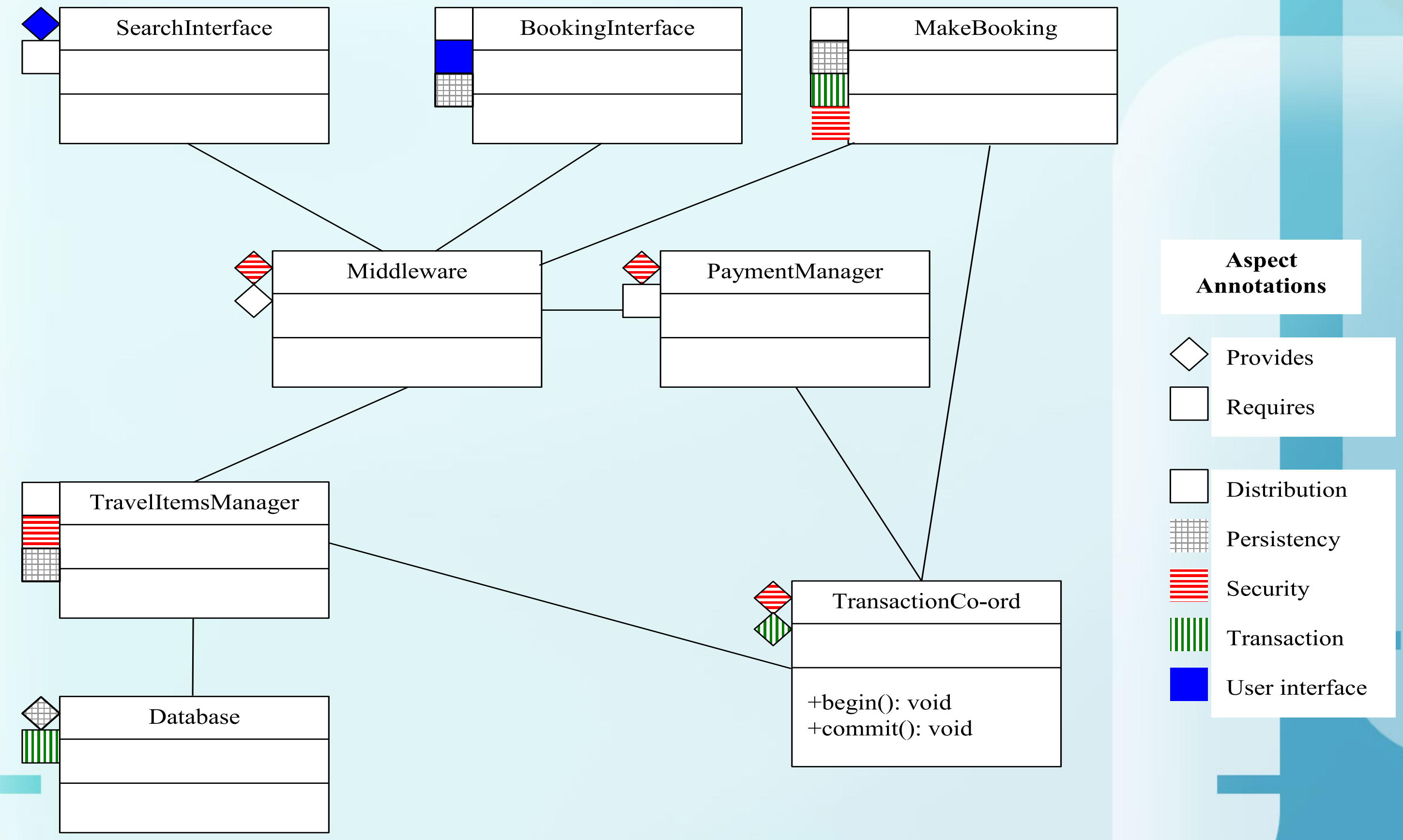
6. Validate and Integrate Web Services

Validation Agents

AO-Client(s)

5. Discover Web Services

AOCE design for travel planner components



SearchInterface

BookingInterface

MakeBooking

Middleware

PaymentManager

TravelItemsManager

TransactionCo-ord

+begin(): void
+commit(): void

Database

Aspect Annotations

◇ Provides

□ Requires

□ Distribution

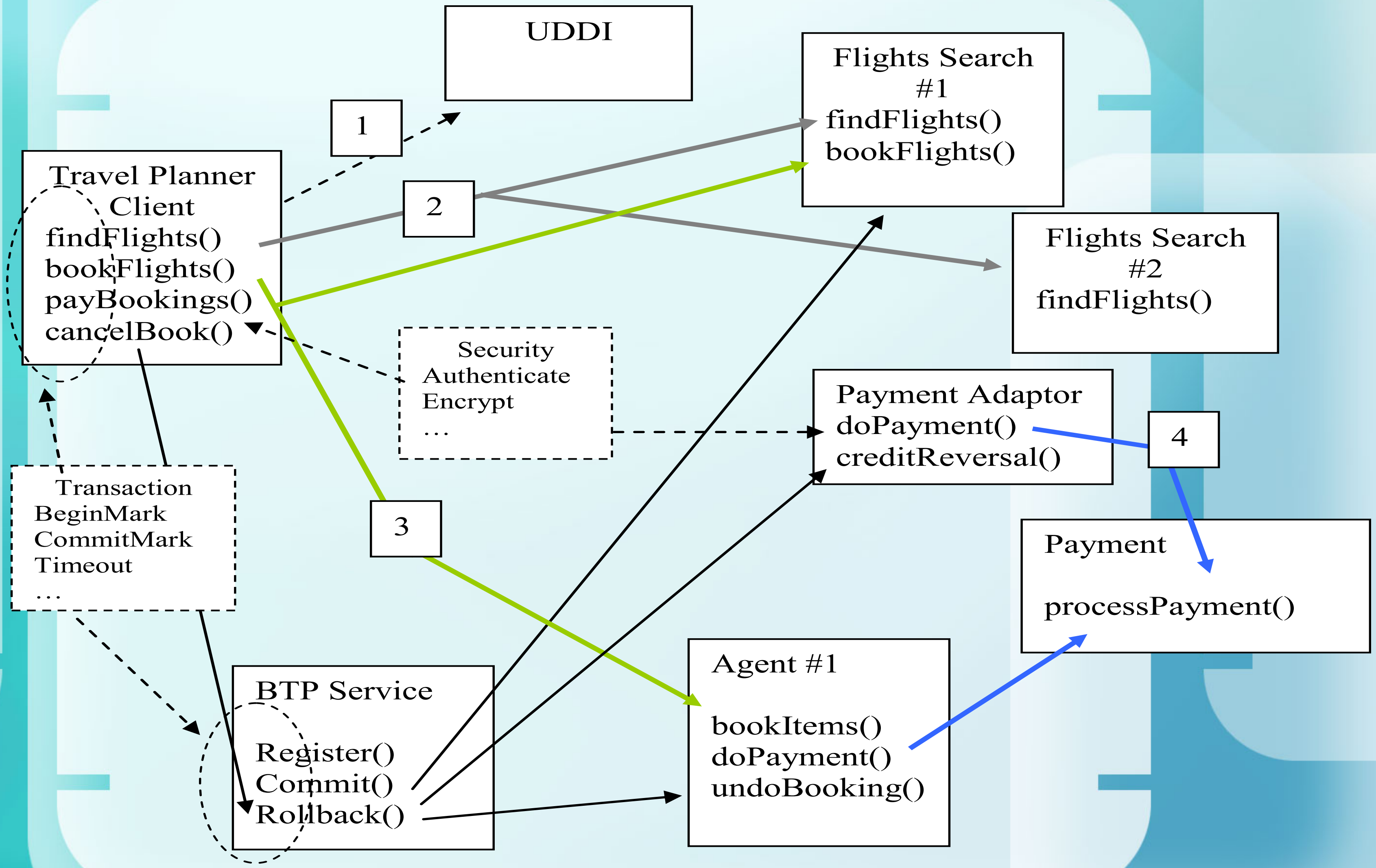
▦ Persistency

🟥 Security

🟩 Transaction

🟦 User interface

Examples of web service aspects



Santokh, Grundy, Hosking, Jing Sun

ECOWS 2005(C)

Alloy

1.) Formal modeling language

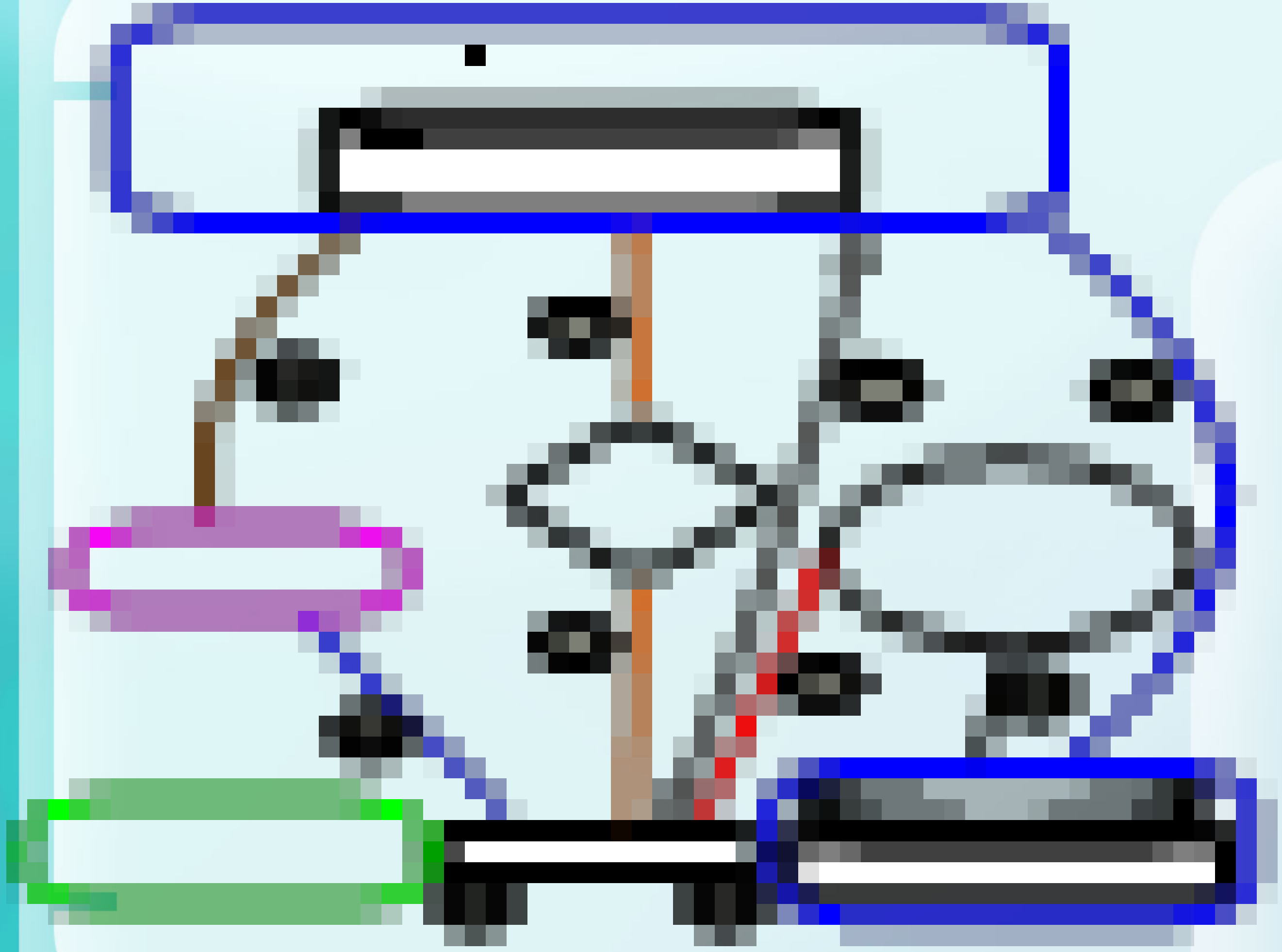
2.) Based on first-order logic for expressing complex structural constraints and behaviors.

3.) Essential constructs are:

- Signature (sig)
- Function (fun)
- Predicate (pred)
- Fact (fact)
- Assertion (assert)

4.) Alloy can be used to formally model, analyse and verify validity of models.

Overview of AOWS architecture



The relationships between subsystems are represented by numbers.

Alloy code deals with relationships between the different entities.

Alloy Specification of AOWS

```
sig AOConnector{
            aocomposite : lone AOComposite,
            directlyConnectedAOWS : set AOWSDL,
            newlyAdvertisedAOWSDL : lone AOWSDL,
            chosenAOWSDL : lone AOWSDL,
            oldAOWSDL : lone directlyConnectedAOWS
}
sig AOWebServiceRequester{
            aoconnector : AOConnector,
            newlyAdvertisedAOWSDL : lone AOWSDL
}
sig AOWebServiceProvider{
            aowsdl : set AOWSDL
}
sig AOWSDL{
            aoComponents : AOComponents
}
sig AOComponents{
            name : String,
            aoComponent : set AOComponent,
            aoDocumentation : AODocumentation,
            aoWSDescription : AOWSDescription
}
```

```
sig AOComponent{
            name : String,
            aoComponentDescription : AOComponentDescription,
            functionalAspect : set FunctionalAspect,
            nonFunctionalAspect :
             set NonFunctionalAspect
}
sig FunctionalAspect {
            type : String,
            aspectName : String,
            aoWSEntryPoint : Boolean,
            standalone : Boolean,
            aspectDetail : FunctionalAspectDetail,
            userOperation : String,
            returnType : String,
            parameter : Parameter
}
```

Alloy Specification of AOWS *cont.*

Facts and predicates, relating providers, requesters and aoconnectors:

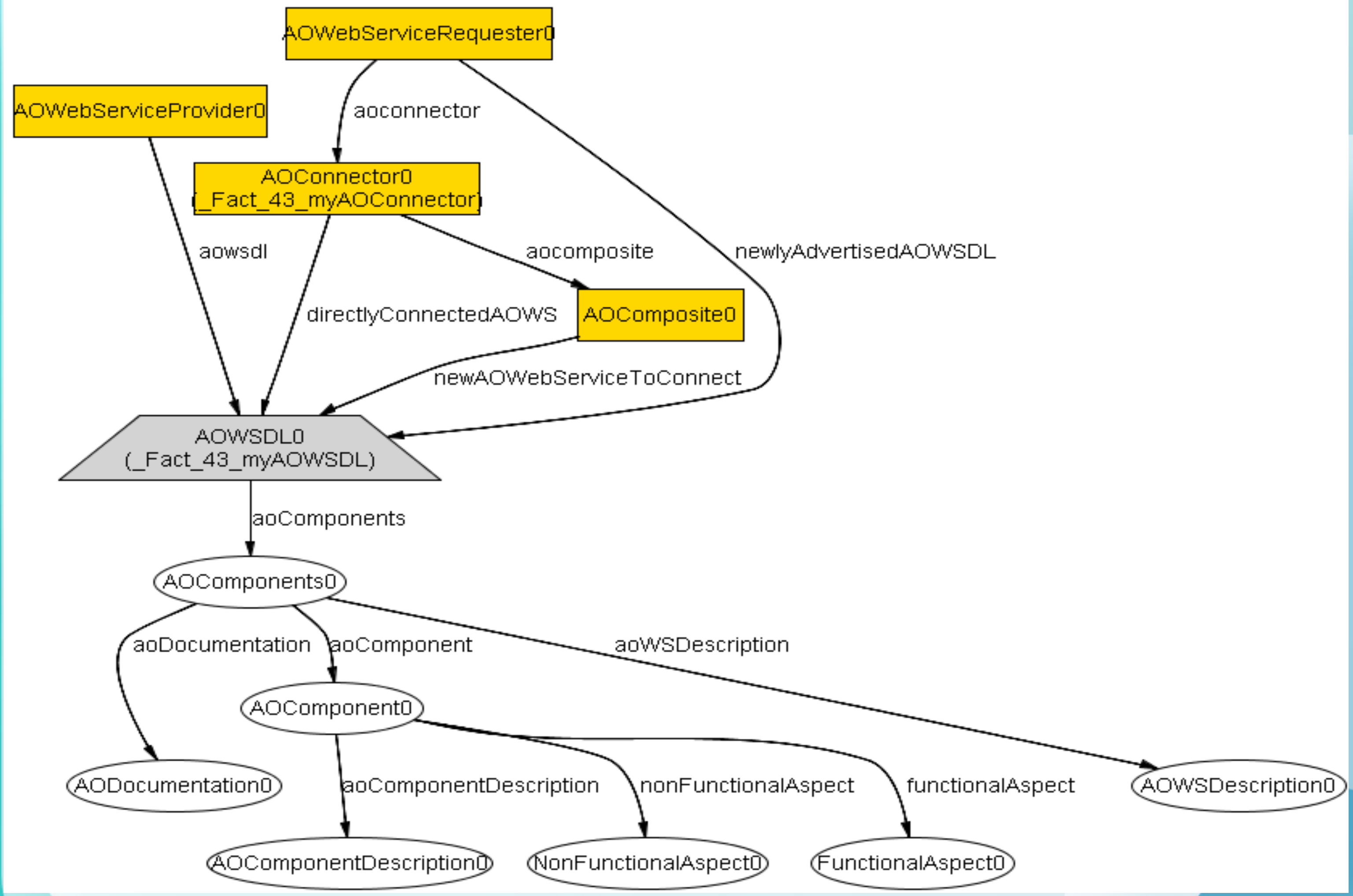
```
fact { no aowsProvider1,
               aowsProvider2 : AOWebServiceProvider |
               aowsProvider1.aowsdl = aowsProvider2.aowsdl }
fact { all myAOWSDL : AOWSDL |
 (one aowsProvider : AOWebServiceProvider |
                              myAOWSDL in aowsProvider.aowsdl) }
pred DirectConnectionToNewAOWS ( myAOConnector' : AOConnector,
myAOConnector : AOConnector ) {
               --precondition
               myAOConnector.newlyAdvertisedAOWSDL
     !in myAOConnector.aowsdl
               -- update the aoconnector
               myAOConnector'.aowsdl =
     myAOConnector.aowsdl +
     myAOConnector.newlyAdvertisedAOWSDL
}
```

Alloy Specification of AOWS *cont.*

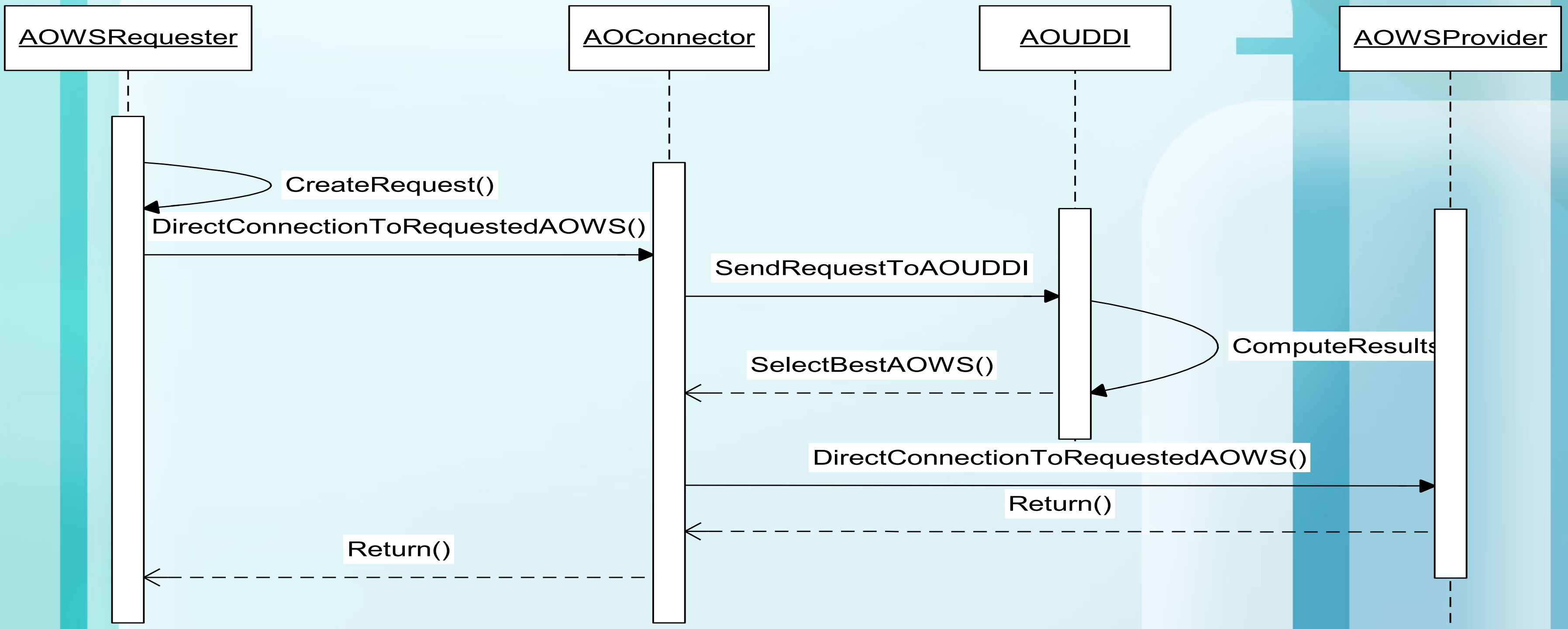
```
sig SearchForHotel {
            type : Persistency,
            aspectName : String,
            aoWSEntryPoint : Boolean,
            standalone : Boolean,
            aspectDetail : SearchForHotelDetail,
            userOperation : String,
            returnType : String,
            parameter : SearchForHotelParameter}
sig SearchForHotelRoom {
            type : Persistency,
            aspectName : String,
            aoWSEntryPoint : Boolean,
            standalone : Boolean,
            aspectDetail : SearchForHotelRoomDetail,
            userOperation : String,
            returnType : String,
            parameter :
SearchForHotelRoomParameter}
```

```
sig SearchForHotelDetail {
            type : SearchForHotelDataRetrieval,
            detail : SelectHotel,
            provided : Boolean}
sig SearchForHotelRoomDetail {
            type : SearchForHotelRoomDataRetrieval,
            detail : SelectHotelRoom,
            provided : Boolean}
fact { all searchHotel : SearchForHotel |
 (one searchHotelDetail :
 SearchForHotelDetail |
 searchHotelDetail in
   searchHotel.aspectDetail) }
fact { all searchHotelRoom :
 SearchForHotelRoom |
  (one searchHotelRoomDetail :
      SearchForHotelRoomDetail |
      searchHotelRoomDetail in
        searchHotelRoom.aspectDetail)}
```

Example of AOWS model generated using ALLOY



Sequence diagram depicting dynamic service discovery



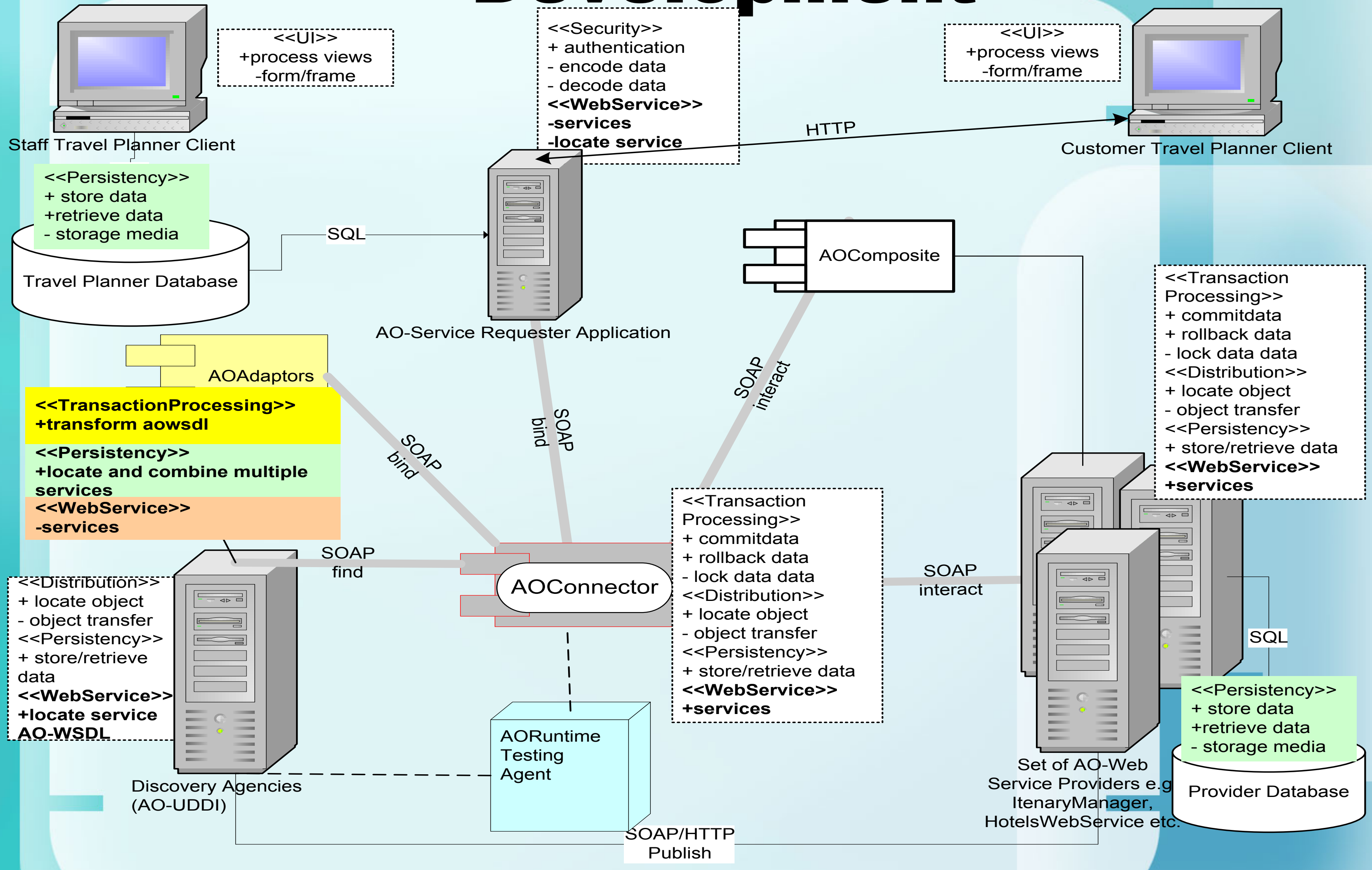
Dynamic discovery processes like this can be captured and analysed using the Alloy Analyser tool

Using Alloy to capture AOWS Dynamic Behaviour

```
assert TestDirectConnectionToRequestedAOWS {
                          all myRequest : Request,
     aowsRequester : AOWebServiceRequester,
                          myAOConnector : AOConnector,
     myAOUDDI : AOUDDI,
                          myResult : Result,
      myAOWSDL : AOWSDL,
                          myAOUDDI' : AOUDDI,
     myAOConnector' : AOConnector  |
                           {
                          CreateRequest ( myRequest,
        aowsRequester )
                          SendRequestToAOConnector
( aowsRequester,
         myAOConnector )
                          SendRequestToAOUDDI ( myAOUDDI',
         myAOConnector, myAOUDDI )
                          ComputeResultAndTransmit ( myResult,
         myAOUDDI, myAOConnector )
                          SelectBestAOWS ( myAOConnector,
         myAOWSDL )
                          DirectConnectionToRequestedAOWS(
        myAOConnector', myAOConnector )
                          }
} check
   TestDirectConnectionToRequestedAOWS for 2
```

Alloy assertion for dynamic service discovery via an AOConnector.

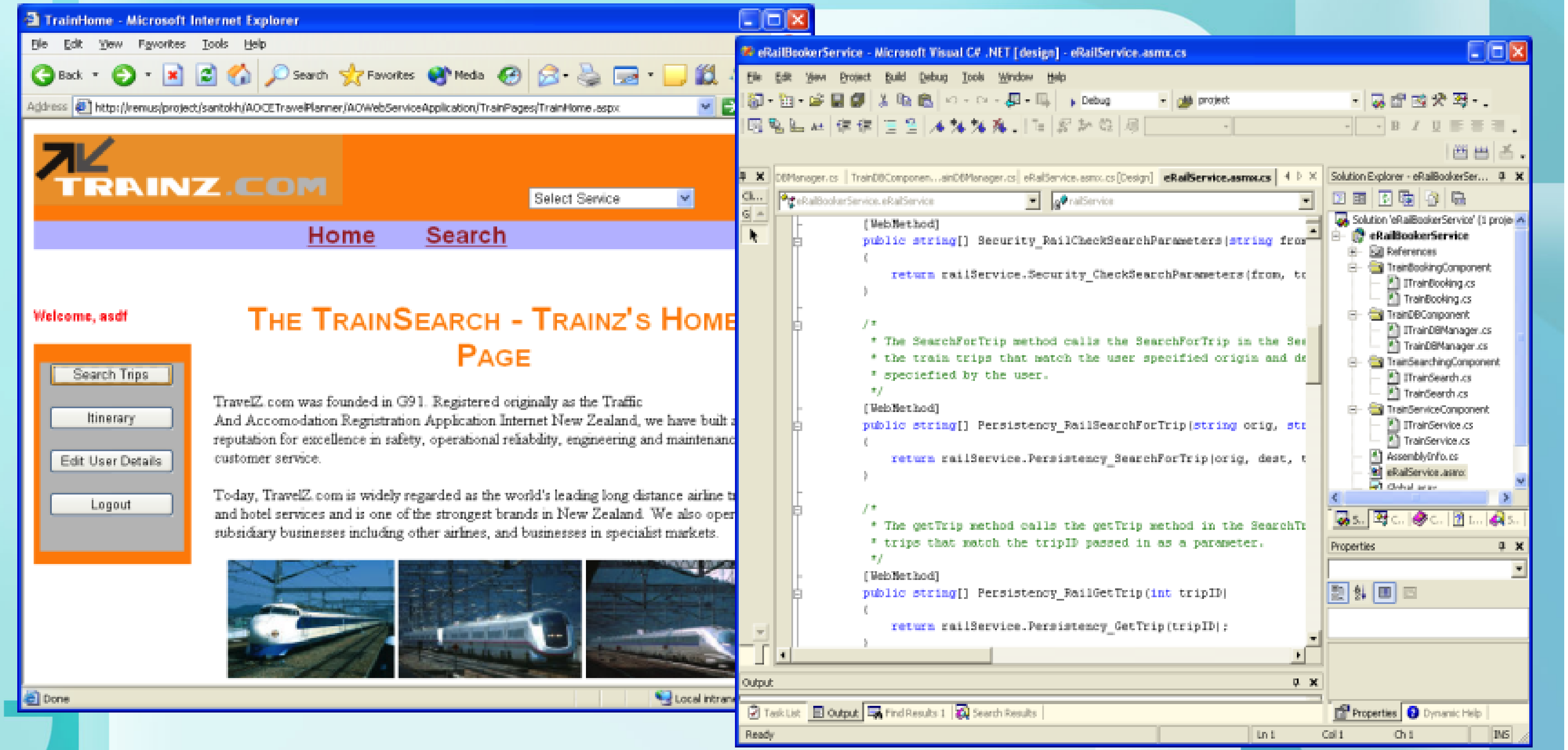
AOWS Architecture for Development



Santokh, Grundy, Hosking, Jing Sun

ECOWS 2005(C)

Travel planner implementation



TrainHome - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://remus/project/santokh/AOCETravelPlanner/AOWebServiceApplication/TrainPages/TrainHome.aspx
TRAINZ.COM
Select Service
Home
Search
Welcome, asdf
Search Trips
Itinerary
Edit User Details
Logout
Done
Local intrane

THE TRAINSEARCH - TRAINZ'S HOME PAGE

TravelZ.com was founded in G91. Registered originally as the Traffic And Accomodation Registration Application Internet New Zealand, we have built a reputation for excellence in safety, operational reliability, engineering and maintenanc customer service.

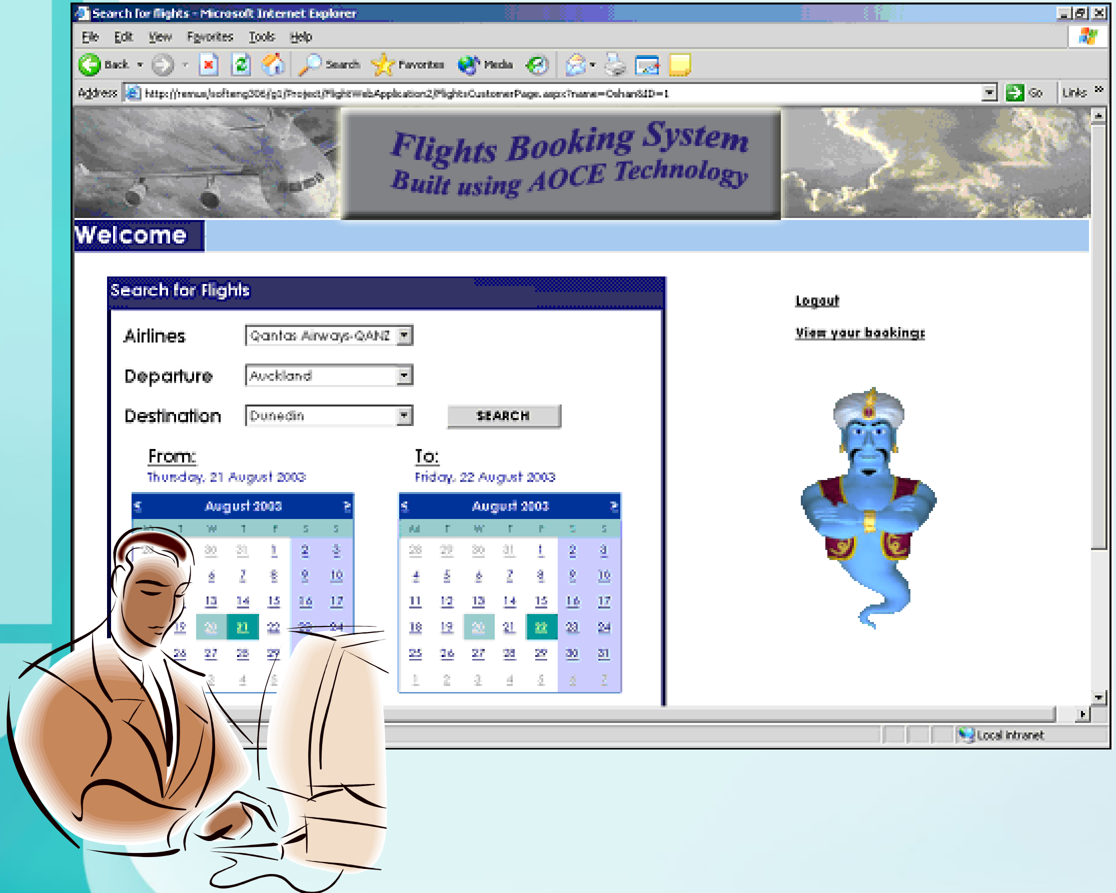
Today, TravelZ.com is widely regarded as the world's leading long distance airline t and hotel services and is one of the strongest brands in New Zealand. We also oper subsidiary businesses including other airlines, and businesses in specialist markets.

```
[WebMethod]
public string[] Security_RailCheckSearchParameters(string from
{
    return railService.Security_CheckSearchParameters(from, to
}

/*
 * The SearchForTrip method calls the SearchForTrip in the Sea
 * the train trips that match the user specified origin and de
 * specified by the user.
 */
[WebMethod]
public string[] Persistency_RailSearchForTrip(string orig, str
{
    return railService.Persistency_SearchForTrip(orig, dest, t
}

/*
 * The getTrip method calls the getTrip method in the SearchTr
 * trips that match the tripID passed in as a parameter.
 */
[WebMethod]
public string[] Persistency_RailGetTrip(int tripID)
{
    return railService.Persistency_GetTrip(tripID);
}
```

(a) Travel planner GUI

(b) Example C# code implementing aspects

Current/Future work



Search for flights - Microsoft Internet Explorer

http://remus/softeng306/g1/Project/FlightWebApplication2/FlightsCustomerPage.aspx?name=Oshan&ID=1

Flights Booking System
Built using AOCE Technology

Welcome

Search for Flights

Airlines: Qantas Airways-QANZ

Departure: Auckland

Destination: Dunedin

From:
Thursday, 21 August 2003

<			August 2003			>
[illegible]	T	W	T	F	S	S
[illegible]	[illegible]	30	31	1	2	3
[illegible]	[illegible]	6	7	8	9	10
[illegible]	[illegible]	13	14	15	16	17
[illegible]	19	20	21	22	23	24
[illegible]	26	27	28	29	[illegible]	[illegible]
[illegible]	[illegible]	3	4	5	[illegible]	[illegible]

To:
Friday, 22 August 2003

<			August 2003			>
M	T	W	T	F	S	S
28	29	30	31	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
1	2	3	4	5	6	7

Logout

View your bookings

- Investigating automated AOWS component discovery and components composite construction.
- Incorporating Multi-Agents into AOWS.
- Semantic AOWS-based systems

Conclusions

AOWS can address issues regarding Web Service's description, dynamic discovery and integration mechanisms.

We successfully carried out the formal specification, analysis and verification of AOWS using Alloy.

AOWS architecture using the AOConnector object made web service based systems more

- modular,
- understandable
- maintainable,
- reusable
- scalable and
- clients more lightweight

This work is supported by the
New Zealand Foundation for Research,
Science and Technology
and
the University of Auckland Research
Committee.

Thank you
very much☺

References

- Singh, S., Grundy, J.C., Hosking, J.G. and Sun, J. An Architecture for Developing Aspect-Oriented Web Services, In Proceedings of the 2005 European Conference on Web Services, Vaxjo, Sweden, Nov 14-16 2005, IEEE Press.
- Wang, Y., Singh, S., Hosking, J.G. and Grundy, J.C. An Aspect-Oriented UML Tool for Software Development with Early Aspects, Proceedings of ICSE 2006 Workshop on Early Aspects at ICSE: Aspect-Oriented Requirements Engineering and Architecture Design, Shanghai, May 2006.
- Singh, S. Chen, H.C. Hunter, O., Grundy, J.C. and Hosking, J.G. Improving Agile Software Development using eXtreme AOCE and Aspect-Oriented CVS, in Proceedings of the 12th Asia-Pacific Software Engineering Conference, Taiwan, December 2005, IEEE CS Press.
- Singh, S., Hosking, J.G. and Grundy, J.C. Deploying Multi-Agents for Intelligent Aspect-Oriented Web Services, In Proceedings of the 2005 Pacific Rim Workshop on Intelligent Multi-agents, Kuala Lumpur, 14-16 September 2005, Lecture Notes in Artificial Intelligence, Springer.
- Singh, S., Grundy, J.C., Hosking, J.G. Developing .NET Web Service-based Applications with Aspect-Oriented Component Engineering , In Proceedings of the Fifth Autralasian Workshop on Software and Systems Architecures, Melbourne, Australia, 13-14 April 2004.
- Grundy, J.C., Panas, T., Singh, S., Stoeckle, H. An Approach to Developing Web Services with Aspect-oriented Component Engineering, In Proceedings of the 2nd Nordic Conference on Web Services, 2003.
- Grundy, J.C. Multi-perspective specification, design and implementation of components using aspects, International Journal of Software Engineering and Knowledge Engineering, Vol. 10, No. 6, December 2000, World Scientific.