On-the-Fly Computing – a new Paradigm for Software Development and Operation

Gregor Engels
University of Paderborn
That’s my life

Gregor Engels
engels@upb.de

University of Paderborn
Information Systems Group

s-lab – Software Quality Lab

C-LAB – Cooperative Computing & Communication Laboratory

Capgemini, CSD Research, Munich
The Vision of On-The-Fly Computing

- Specific & Individual Wish
- Search for Matching Services
- Analysis & Decomposition
- Analysis & Functionality Check
- Configuration
- Matching
- Custom-made Product

- ≤ 10s
Software Market of the Future

How to find?
- efficient / effective search
- appropriate matching

How to be found?
- high fitting accuracy
- high reputation

Service Requestor

Service Provider

Service 1

Service 2

required interface

provided interface
SFB 901

ON - THE - FLY COMPUTING

- Collaborative Research Center (CRC)
- funded by DFG (Deutsche Forschungsgemeinschaft)
- 2011 – 2015 (…- 2023)
- 50 scientists (computer science, economics)
## SFB 901: Structural Overview

### Project Area A

**Algorithmic and Economic Foundations**
- **A1:** Local Strategies in Dynamic Networks
- **A2:** Overlays over Physical Networks
- **A3:** Market of Services

### Project Area B

**Modeling, Composition & Quality Analysis of Services**
- **B1:** Parametric Service Specifications
- **B2:** Configuration & Rating
- **B3:** Composition Analysis in Uncertain Contexts
- **B4:** Proof-Carrying Services

### Project Area C

**Reliable Runtime Environments & Application Scenarios**
- **C1:** Robustness & Security
- **C2:** On-The-Fly Compute Centers
- **C3:** Modeling of Optimization Problems
Project Area B: Structural Overview

Project Area A
Algorithmic and Economic Foundations

Project Area C
Reliable Runtime Environments & Application Scenarios

Project Area B
Modeling, Composition & Quality Analysis of Services
- B1: Parametric Service Specifications
- B2: Configuration & Rating
- B3: Composition Analysis in Uncertain Contexts
- B4: Proof-Carrying Services
Project Area B: Modeling, Composition & Quality Analysis of Services

- **B1** Languages
  - GoogleMaps
  - PaderTouristik
  - PaderSprinter

- **B1** Matching
  - ≤ 10s

- **B2** Configuration
  - TripPlanner
  - Rating

- **B3** Analysis

- **B4** Certification
  - ≤ 10s
Subproject B1: Parametric Service Specifications

Goal

Enable the specification of service requests of a client and provided services in OTF Markets and the matching between them
Subproject B1: Parametric Service Specifications

Example

OTF Market

MoveMeAround

WSDL

Petri Net

RDSEFF

Client

?A

≤ 10s

WSDL

UML Statechart

BPEL

© Heinz Nixdorf Institut, Universität Paderborn
10 – July 10, 2013 SATToSE 2013, Bern (CH)
Subproject B1: Parametric Service Specifications

Example

OTF Market

Client

WSDL  UML Statechart  BPEL

WSDL  Petri Net  RDSEFF

WSDL  Statechart  RDSEFF

WSDL  Story Diagram  BPEL

WSDL  Statechart  BPEL

Idl  Statechart  BPEL

Sequence D.

© Heinz Nixdorf Institut, Universität Paderborn
Hotel Reservation at HRS

Step 1: Search

Online Hotel Reservation (www.hrs.com)

Step 2: Show Results

Step 3: Book Room

How it works

HRS Platform

www.hrs.com

Independent Hotels

Chain Hotels

etc.

etc.

HRS: Hotel Reservation Service
Rich Service Descriptions & Automatic Matching

Service Requestor

Service Provider

Structural Aspects:
\[ \text{op1}(p1:A) \]
\[ \text{op2}(p1:B):A \]
\[ \text{op3}(p1:C) \]

Service Request

Service Offer

Behavioral Aspects:
\[ \text{loop} \]
\[ \{ \text{pre} \} \text{op1}(\ldots) \{ \text{post} \} \]
\[ \ldots \]

Behavioral Aspects:
\[ \{ \text{pre} \} \text{opa}(\ldots) \{ \text{post} \} \]
\[ \ldots \]
\[ (\text{opa}(\ldots) \text{opb}(\ldots) \text{opc}(\ldots) )^* \]
Matching Parameters

- Ontological Heterogeneity
- Operation Matching (1:1, 1:n, n:1, n:m)
- Protocol Matching
- Service Composition

HRS Service Request
- checkAvailability(...)
- viewDetails(...)
- makeReservation(...)
- makePayment(...)

Hotel X Service Offer
- getAvailableRoom(...)
- makeABooking(...)
- validateCredentials(...)
- payForBooking(...)

VC: checkAvailability()
(a)
(b)
(c)

VC: getAvailableRoom()
(a)

ProfileType
RoomType

ProfileType
RoomType

::ProfileType
::RoomType

Profile
Room

Profile
Room

Client

hotel

HTTP

HTTP

getAvailableRoom

s1

s3

s5

s4

HRS

Hotel X

HarmoNET-based
Local Ontology

OTA-based
Local Ontology
1. **Requestor operation**

   VC: makePayment() (Translated)

   - Booking
   - : Client
   - : Payment
   - : PaymentMode

2. **Result = validateCredentials(...)**

   VC: validateCredentials()

   - : Client
   - : Client
   - : PaymentMode

3. **Result = Result + payForBooking(...)**

   VC: payForBooking()

   - Booking
   - : Client
   - : Payment
   - : PaymentMode
   - : Client
   - : Payment
   - : PaymentMode
   - : Receipt
1. Requestor operation

   VC: makePayment() (Translated)

   : Booking  : Booking  : Receipt
   : Client   : Client   : Payment : PaymentMode

2. Result = validateCredentials(…)

   VC: validateCredentials()

   : Client  : Client  : PaymentMode

3. Result = Result + payForBooking(…)

   VC: validateCredentials()

   : Client  : Client  : PaymentMode

4. Result = validateCredentials(…) + payForBooking(…)
# Operation Mappings

<table>
<thead>
<tr>
<th>Type</th>
<th>HRS Operations</th>
<th>HotelX Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:1</td>
<td>makeReservation()</td>
<td>makeABooking()</td>
</tr>
<tr>
<td>1:n</td>
<td>makePayment()</td>
<td>validateCredentials()</td>
</tr>
<tr>
<td></td>
<td></td>
<td>↓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>payForBooking()</td>
</tr>
<tr>
<td>n:1</td>
<td>checkRoomAvailability()</td>
<td>getAvailableRoom()</td>
</tr>
<tr>
<td></td>
<td>↓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>viewDetails()</td>
<td></td>
</tr>
</tbody>
</table>
Subproject B1: Parametric Service Specifications
Example
Subproject B1: Parametric Service Specifications

Approach

parametric core language

Behavior
Performance
Protocols
Contracts

matching

mapping-by-example

PetriNets
UML Statecharts
Visual Contracts

concrete languages
Vision: OTF Computing in 2023

Project Area A
- Algorithmic & Economic Foundations
  - Data Management
  - Model Generation

Model Solver
- Visualization
- Decision Support

Project Area B
- Modeling, Composition & Quality Analysis

Project Area C
- Reliable Execution Environments & Application Scenarios
  - Model Generation
  - Model Solver
  - Visualization
  - Decision Support

OTF Service Provider for Optimization
- User

OTF Compute Center
- System
Software Market of the Future

Service Requestor

Service Provider

How to find?
- efficient / effective search
- appropriate matching

How to be found?
- high fitting accuracy
- high reputation
Software Market of the Future

How to be found?
• high fitting accuracy
• high reputation

Enterprise Architecture

Business Architecture

IT Architecture

Business Services

IT services

Business / IT alignment

Service Provider

provided interface

Service 2
A Method for Designing IT-Services

- Business Architecture → Business Services
- Enterprise Architecture → Business / IT alignment
- IT Architecture → IT services

Quasar Enterprise
Anwendungslandschaften serviceorientiert gestalten

Engels · Hess · Humm · Juwig · Lohmann · Richter · Voß · Willkomm

Capgemini
CONSULTING. TECHNOLOGY. OUTSOURCING
dpunkt.verlag
Integrated Architecture Framework (IAF)
Integrated Architecture Framework (IAF)

Integrated Architecture Framework (IAF), Capgemini
## A Method for Designing IT-Services

<table>
<thead>
<tr>
<th>Architectural Aspects</th>
<th>Business Aspects</th>
<th>IT Aspects</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Business</strong></td>
<td><strong>Information</strong></td>
<td><strong>Information system</strong></td>
</tr>
<tr>
<td><strong>Technical</strong></td>
<td><strong>Infrastructure</strong></td>
<td></td>
</tr>
</tbody>
</table>

### Questions

- **Why?**
  - What are strategic goals, principles, context?

- **What?**
  - What are the business services?
  - What is the relevant information?
  - What are the relevant IT services?
  - What are the relevant technical services?

- **How?**
  - How is the business logically structured?
  - How is the information logically structured?
  - How are the services logically structured into components?
  - How are the technical services logically structured into TI-components?

- **With what?**
  - What are the relevant operational business units?
  - How is the information managed?
  - How are the components physically realized?
  - How are the TI-components physically realized?

---

Integrated Architecture Framework (IAF), Capgemini
Example Application Domain: Travel Agency

- Christopher Columbus Travel (CCT)
  - fictitious travel agency
  - sells package travels and (individual) custom travels
Identification of Level 1 business services

Level 1 business services
- Internally and externally offered core services of an enterprise to fulfill its business objectives.

Def.

- Evaluate last travel season and plan new one
- Buy hotel beds and flight seats
- Design travel packages and fix a price
- Book travels
- Help customers before, during, after travel
Refined Business Services

Notation: UML-like Use Case Diagram
## A Method for Designing IT-Services

### Architecture Framework (IAF), Capgemini

<table>
<thead>
<tr>
<th>Questions</th>
<th>Business Aspects</th>
<th>IT Aspects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Why?</td>
<td>What are strategic goals, principles, context?</td>
<td></td>
</tr>
<tr>
<td>What?</td>
<td>What are the business services?</td>
<td>What is the relevant information?</td>
</tr>
<tr>
<td>How?</td>
<td>How is the business logically structured?</td>
<td>How is the information logically structured?</td>
</tr>
<tr>
<td>With what?</td>
<td>What are the relevant operational business units?</td>
<td>How is the information managed?</td>
</tr>
</tbody>
</table>

---

**Integrated Architecture Framework (IAF), Capgemini**

---

30 – July 10, 2013  SATToSE 2013, Bern (CH)
Designing Domains

**Def.**

**Domain**
- business-oriented clustering of components of an application domain
- maybe hierachically structured
- components are associated to leaf domains

**Notation:**
- UML-like Package and Component Diagrams
Core business services are candidate domains

- Planning
- Purchasing
- Production
- Sales
- Support
Domains of Christopher Columbus Travel

- **Customer Access**
  - Travel agency (TRA)
  - Internet (INT)
  - Call Center (CCE)

- **Core Business**
  - Planning (PLA)
  - Purchasing (PUR)
  - Production Custom Tours (PCT)
  - Production Package Tours (PPT)
  - Booking (BOK)
  - Support (SRV)

- **Resources**
  - Resource Management (RSM)
  - Customer Management (CUM)
  - Order Management (ORM)

- **Support**
  - Accounting (ACC)
  - Reporting (REP)
  - Personnel (PER)

- **Note:**
  - Derived from business objects
  - Derived from supporting business services
Categories of IT services

Def.

IT service
- is a business service (or part of it) which is realized by a software application

Def.

Categories of IT services
- data: managing business objects
- function: providing algorithmic business logic
- process: providing flow-oriented business logic
- interaction: allowing users to interact with applications

Examples: Categories of IT services
- data: Maintain customer
- function: Select resource
- process: Book custom travel
- interaction: Compose custom travel
Domains with associated categorized IT services

Legend:

- **YYY**: Domain
- **XXX**: Application service
- **... of category function**
- **... of category data**
- **... of category process**
- **... of category interaction**

---

**CCT**

- **TRA**: Sell custom travel
- **INT**: Compose custom travel

**PLA**

- **PUR**: Book resource
- **PCT**: Check plausibility

**BOK**

- **RSM**: Check Availability
  - **Book resource**
- **CUM**: Maintain customer

**SRV**

- **ORM**: Maintain travel order

---

**Legend**

- **Application service**
- **Domain**
- **Function**
- **Process**
- **Interaction**
Designing components

**Def.**

**Component**
- realizes IT services
- has explicit interfaces for provided and requested operations

Diagram of components and their interactions:

- Domain 1:
  - C 1.1
  - C 1.2
  - C 1.3
  - C 1.4
  - C 1.5
  - Int 1.3.3
  - Int 1.5.1

- Domain 2:
  - C 2.1
  - C 2.2
  - C 2.3
  - C 2.4
  - C 2.5
  - Int 2.3.1

...
Determine candidate components and refine

Candidate component
- All IT services of the same domain and category become a candidate component

Design Rules for refining candidate components
- a component belongs to exactly one domain.
- all operations of a component shall be of the same category.
- business logic that changes at a different pace shall be separated.
- components of category data have responsibility of business objects.
- components shall not have cyclic dependencies.
- components of different categories shall have layered dependencies according to interaction → process → function → data.
- components shall have low coupling and high cohesion.
- …
Final components of Christopher Columbus Travel
Designing interfaces

Design rules for interfaces and operations

- **business-oriented**: every operation shall provide business logic only and must not reveal implementation details.

- **coarse-grained**: operations shall comprise as much business logic as possible.

- **idempotent**: multiple invocations of an operation with the same parameters shall have the same effect as a single invocation.

- **compensating**: for each operation there shall be a compensating operation which undoes its business implications.

- **context free**: operations shall have minimal knowledge on the context in which they are invoked (session context, transaction context, batch / online).
Software Market of the Future

Service Requestor

Service Provider

Service 1

Service 2

required interface

provided interface

How to find?
- efficient / effective search
- appropriate matching

How to be found?
- high fitting accuracy
- high reputation
The Vision of On-The-Fly Computing

Specific & Individual Wish

Custom-made Product

Analysis & Decomposition

Analysis & Functionality Check

Configuration

Matching

Search for Matching Services

≤ 10s

≤ 10s

≤ 7s

≤ 7s
Thank you for your attention!

Prof. Dr. Gregor Engels  
University of Paderborn  
Zukunftsmeile 1  
33102 Paderborn  
Germany  

Phone: +49 52 51 / 60 33 37  
E-mail: engels@upb.de  
Web: http://sfb901.uni-paderborn.de/