



**Hochschule
Bonn-Rhein-Sieg**
University of Applied Sciences

Semesterproject

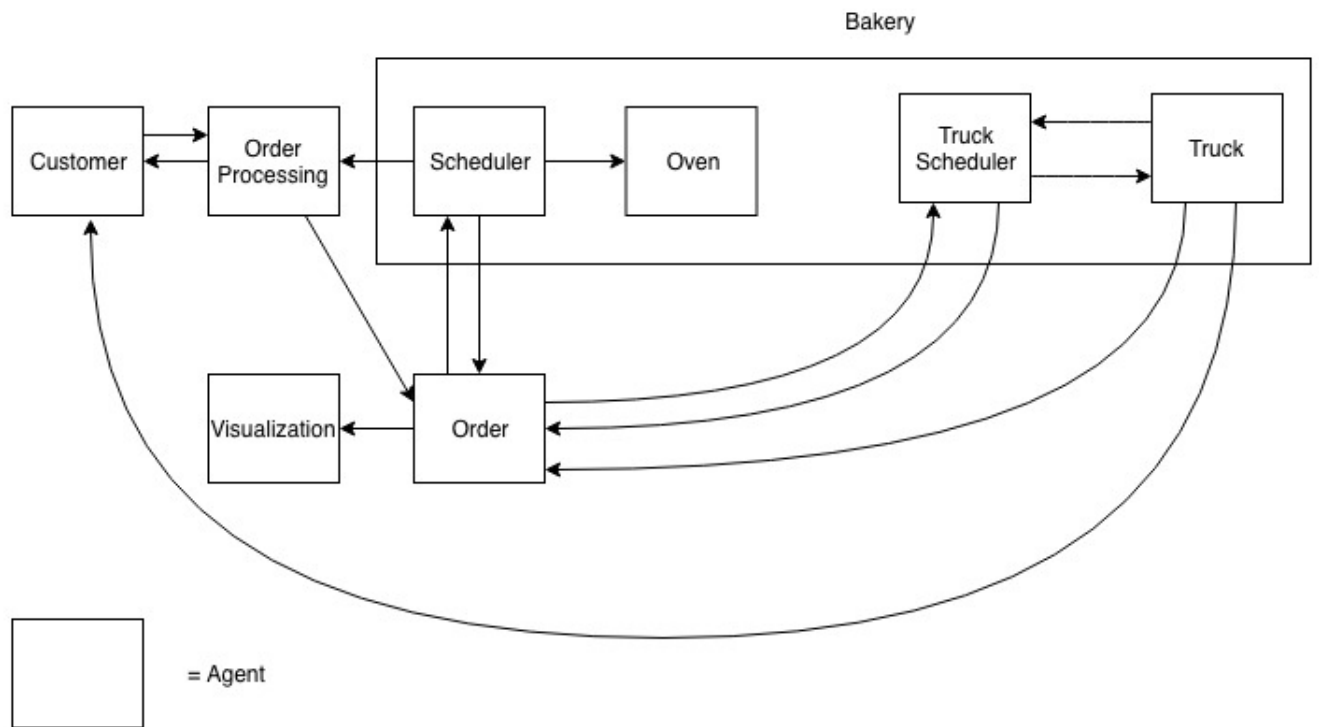
Multi Agent and Agent Systems

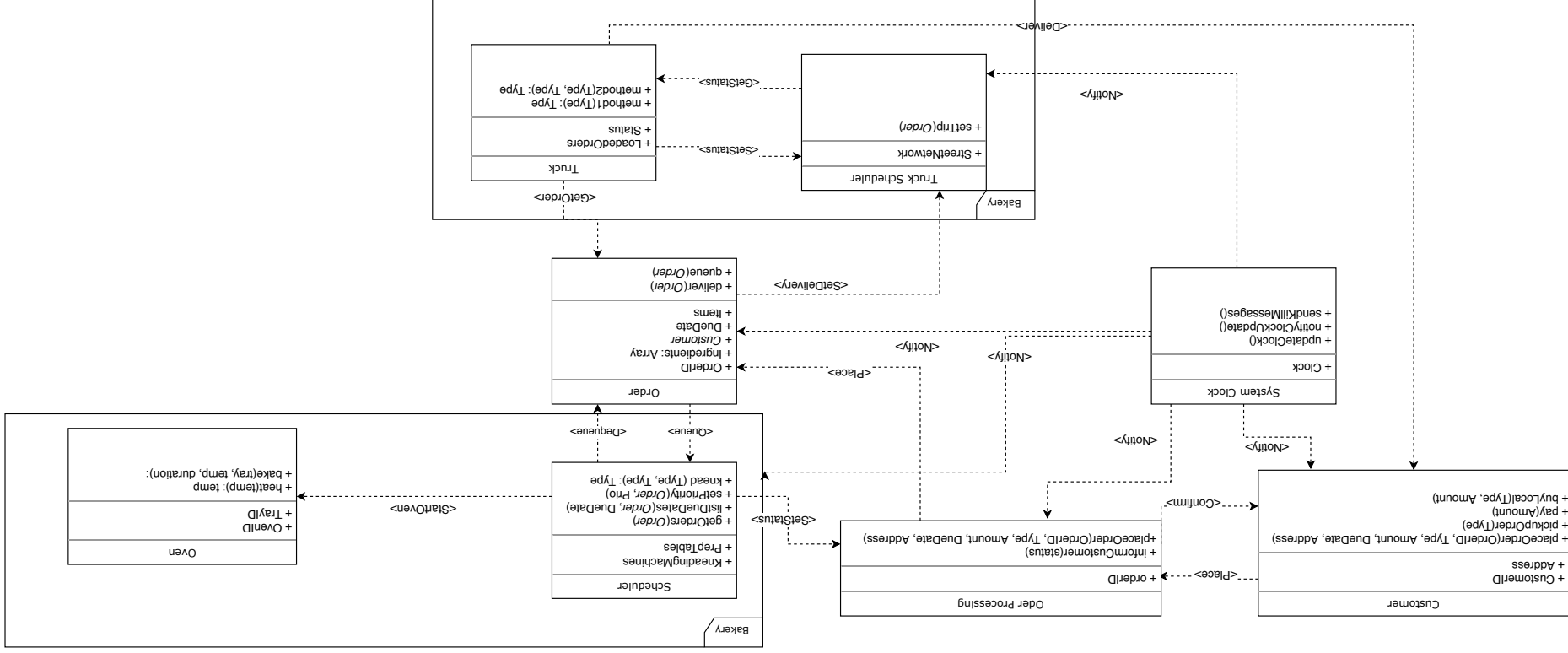
Department of Autonomous Systems
Referent: Prof. Dr. Gerhard Kraetzschmar

Submitted by: Team PJT

Sankt Augustin 5. November 2018

Architecture





Objects

Product		
f	guid	String
f	boxingTemp	int
f	salesPrice	float
f	breadPerOven	int
f	breadPerBox	int
f	itemPrepTime	int
f	doughPrepTime	int
f	bakingTemp	int
f	coolingRate	int
f	bakingTime	int
f	restingTime	int
f	productionCost	float

Node		
f	sName	String
f	sCompany	String
f	lLocation	Location
f	sGuid	String
f	sType	String
f	eEdgesFrom	Edges[]
f	eEdgesTo	Edges[]

Edges		
f	sSource	String
f	sTarget	String
f	sGuid	String
f	fDist	float

StreetNetwork		
f	hmNodes	HashMap<String, Node>
f	bDirected	boolean

Box		
f	iCapacity	int
f	bIsLoaded	boolean

Clock		
f	day	int
f	hour	int

Location		
f	fX	float
f	fY	float

Aggregation of order data

Aggregation of order data can be done in the following manner:

- **An aggregation of a customer's orders for each day or each date <ddd.hh>**
→ It depends which data structure you could use

- If it is really important to you that you access date by given data format you could use a hashmap. Key is date value is order. Worst performance of searching a hashmap is $O(n) = \log(n)$

```
HashMap<Date, Order> hmMapDaily = new HashMap<Date, Order>();  
hmMapDaily.put(new Date(), new Order());  
Order co = hmMapDaily.get(date);
```

- If it is not that important to use the given dateformat you could use an array. Index is day of a year. That means here worst performance of searching an array given that you know which day you want to search is $O(n) = 1$

- **An aggregation of all orders for a particular product for each day or each date**
→ Hashmap of Hashmaps. One entry within Hashmap represents one product. Key is product value is a hashmap. One Hashmap within Hashmap has as key a date, as value an array of orders.

```
HashMap<ProductId, HashMap<Date, Orders[]>> hMapProduct;  
hMapProduct.put(new ProductId(), HashMap<Date, Orders[]>());  
HashMap<Date, Orders[]> hmDate = hMapProduct.get(ProductId());
```

So hMapProduct would look the following way:

$$hMapProduct = \begin{pmatrix} \{ProductId, HashMap < Date, Orders[] >\} \\ \cdot \\ \cdot \\ \cdot \\ \{ProductId, HashMap < Date, Orders[] >\} \end{pmatrix}$$