

# **C4 – Documentation to blow (up) your mind!**

## Wer bin ich?

- Charlotte Scharbert, 31
- Duales Studium angewandte Mathematik und Informatik
- Neun Jahre lang Legacy-Code ohne Doku gepflegt
- Dann Wechsel zur pharma mall

## Wer ist die pharma mall?

- Der größte Dienstleister für Transaktionsprozesse zwischen Herstellern und Kunden in der Pharmabranche
- Ziel ist außerdem die Digitalisierung in der Gesundheitsbranche voran zu treiben
- Unsere Webseite: <https://www.pharma-mall.de>



## Worum soll es hier gehen?

- Wieso ist Doku so unbeliebt?
- Was ist C4?
- Was unterscheidet C4 von anderen Ansätzen?

# Wieso ist Dokumentation so unbeliebt?

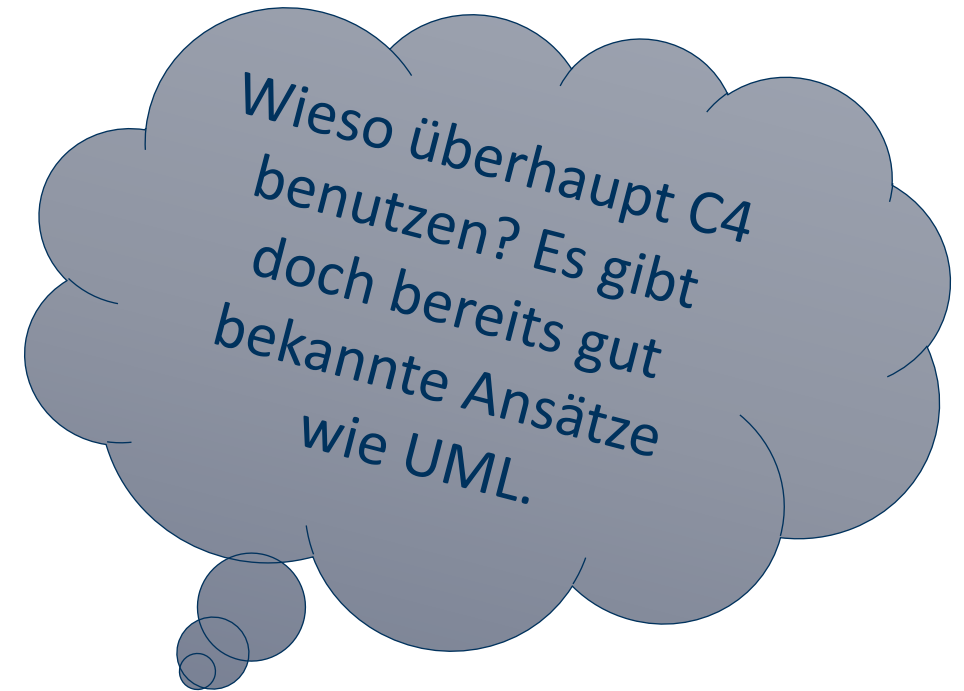
## Wie würde Dokumentation in einer perfekten Welt aussehen?

- Doku ist aktuell/vollständig
  - Doku wird regelmäßig aktualisiert
  - Dokumentieren fällt leicht
- Doku ist verständlich
  - Einheitliches Konzept/Sprache für jedes Projekt
- Doku ist hilfreich
  - Doku enthält nur die Infos, die benötigt werden

# DIE REALITÄT?

## Umfrage bei unseren Entwicklerinnen

- Doku ist pflege-/zeitintensiv
- Bilder / „optische“ Doku ist hilfreich
- Viele Diagrammarten benannt
  - Keins der Diagramme regulär in Benutzung



# DIE REALITÄT?

Trifft überhaupt nicht zu

C4 – Documentation to blow (up) your mind!

Trifft voll und ganz zu

24-07-25

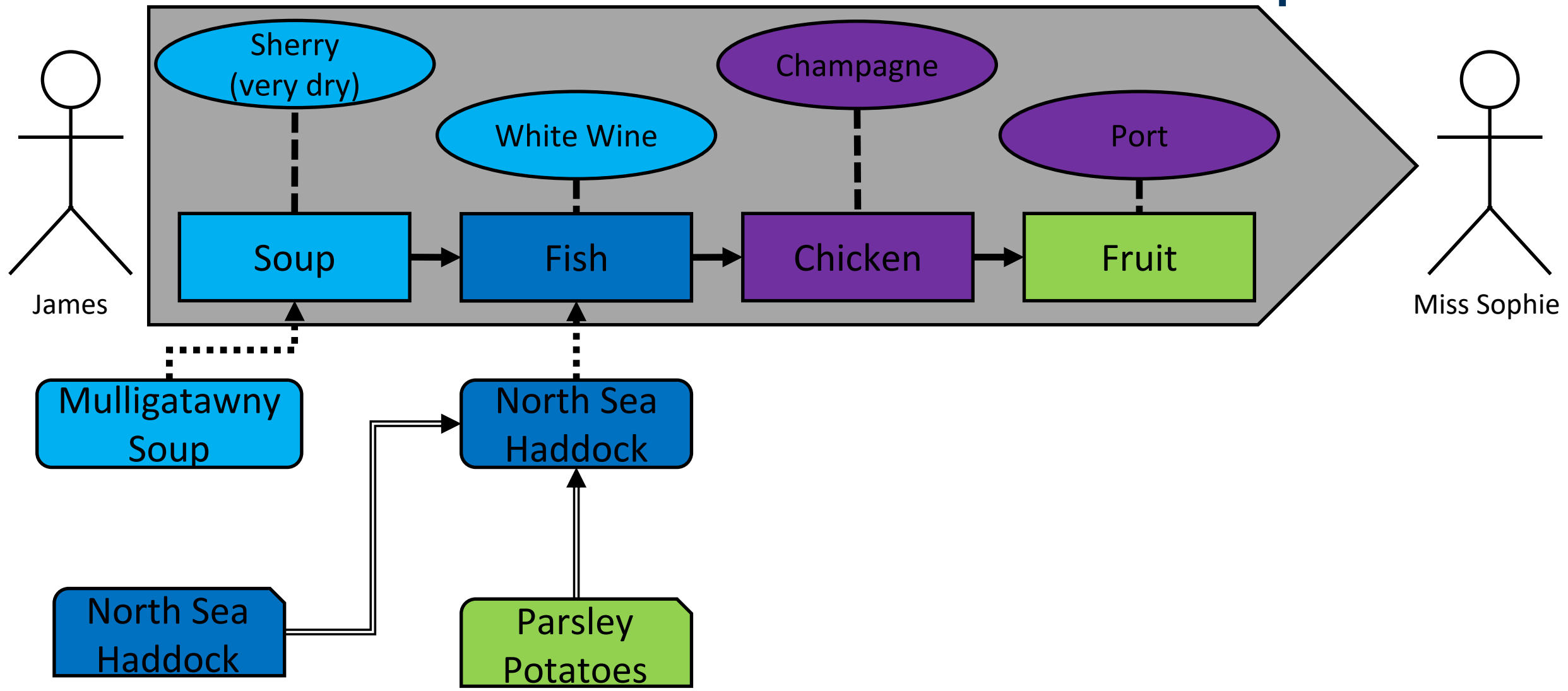
## Warum ist die Utopie nicht Realität?

- Doku in Textform ist schwerer verständlich
- Doku in Bildform ist aufwändiger zu pflegen
- Dokumentieren ist zeitaufwändig
- Dokumentieren ist kein fester Bestandteil der Ausbildung
- Diagramme und Formulierungen sind nicht einheitlich
- Infos die man sucht sind schwer zu finden



**Wie könnte es besser  
gehen?**

# DINNER FOR ONE 1.0



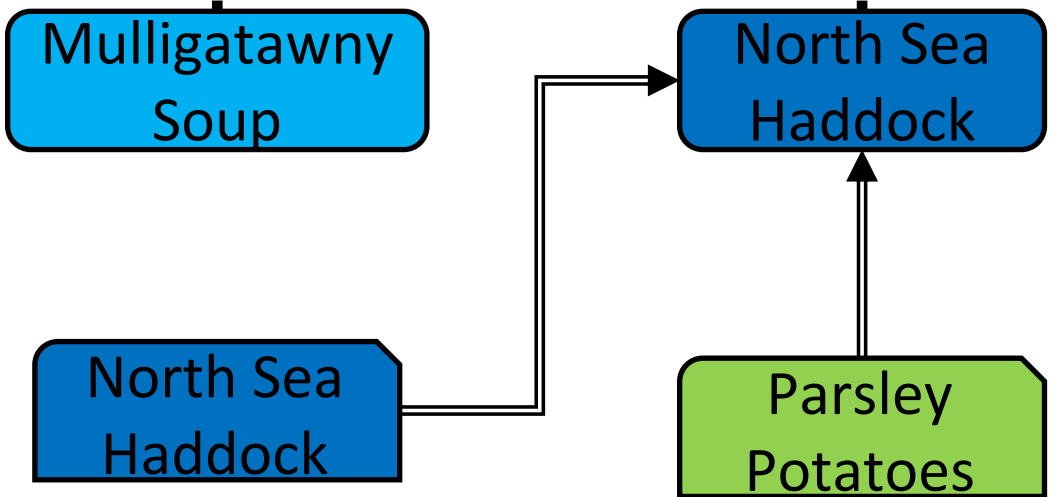
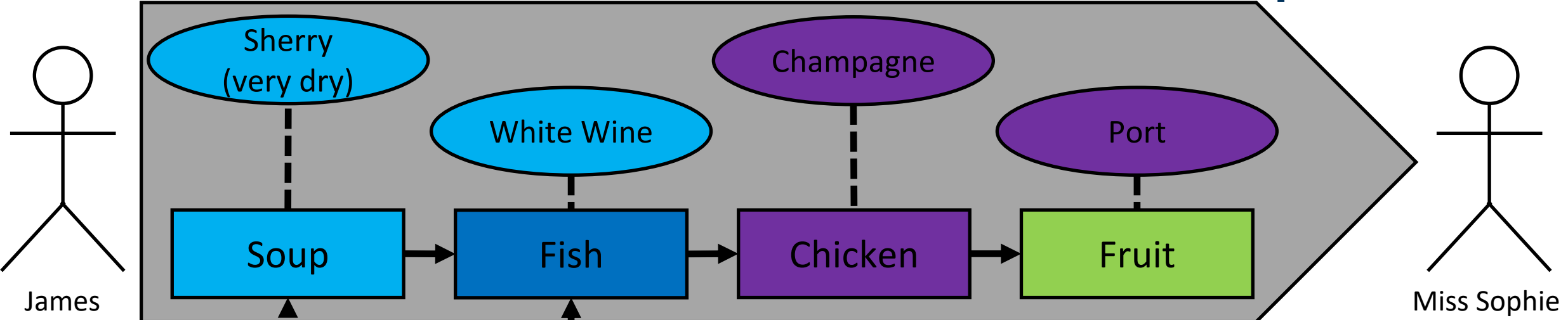
# DINNER FOR ONE 1.0

- Bietet Überblick
- Bedeutung von Farben, Formen, Pfeilen unklar
- (Unübersichtlich)



**Also: Legende hinzufügen!**

# DINNER FOR ONE 1.1



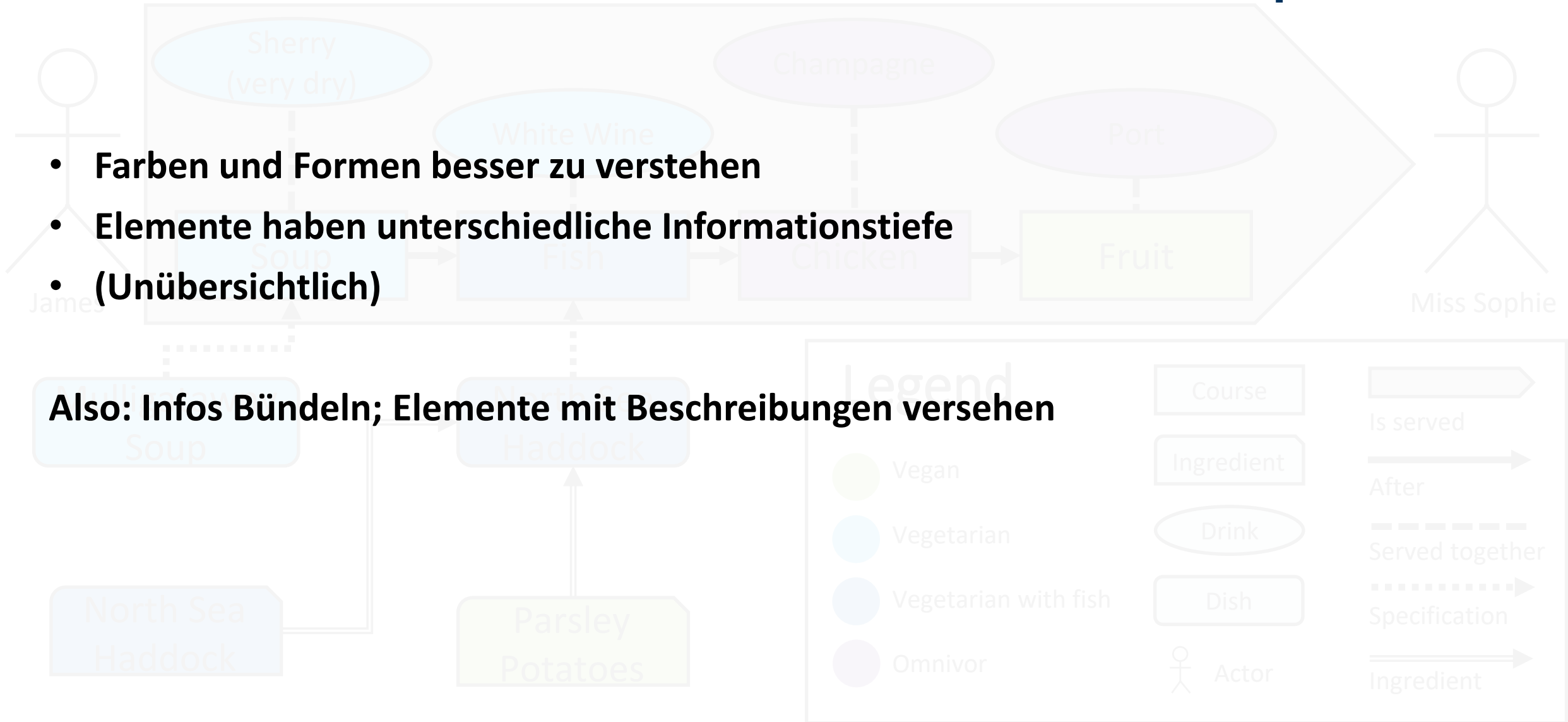
### Legend

<span style="color: green;">●</span> Vegan	<span style="border: 1px solid black; padding: 2px;">Course</span>	<span style="border: 1px solid black; padding: 2px;">Is served</span>
<span style="color: cyan;">●</span> Vegetarian	<span style="border: 1px solid black; padding: 2px;">Ingredient</span>	<span style="border: 1px solid black; padding: 2px;">After</span>
<span style="color: blue;">●</span> Vegetarian with fish	<span style="border: 1px solid black; padding: 2px;">Drink</span>	<span style="border: 1px solid black; padding: 2px;">Served together</span>
<span style="color: purple;">●</span> Omnivor	<span style="border: 1px solid black; padding: 2px;">Dish</span>	<span style="border: 1px solid black; padding: 2px;">Specification</span>
	Actor	<span style="border: 1px solid black; padding: 2px;">Ingredient</span>

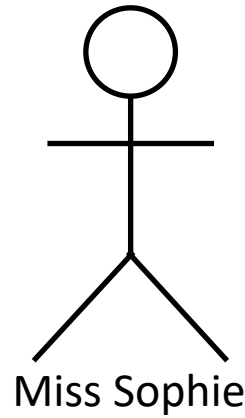
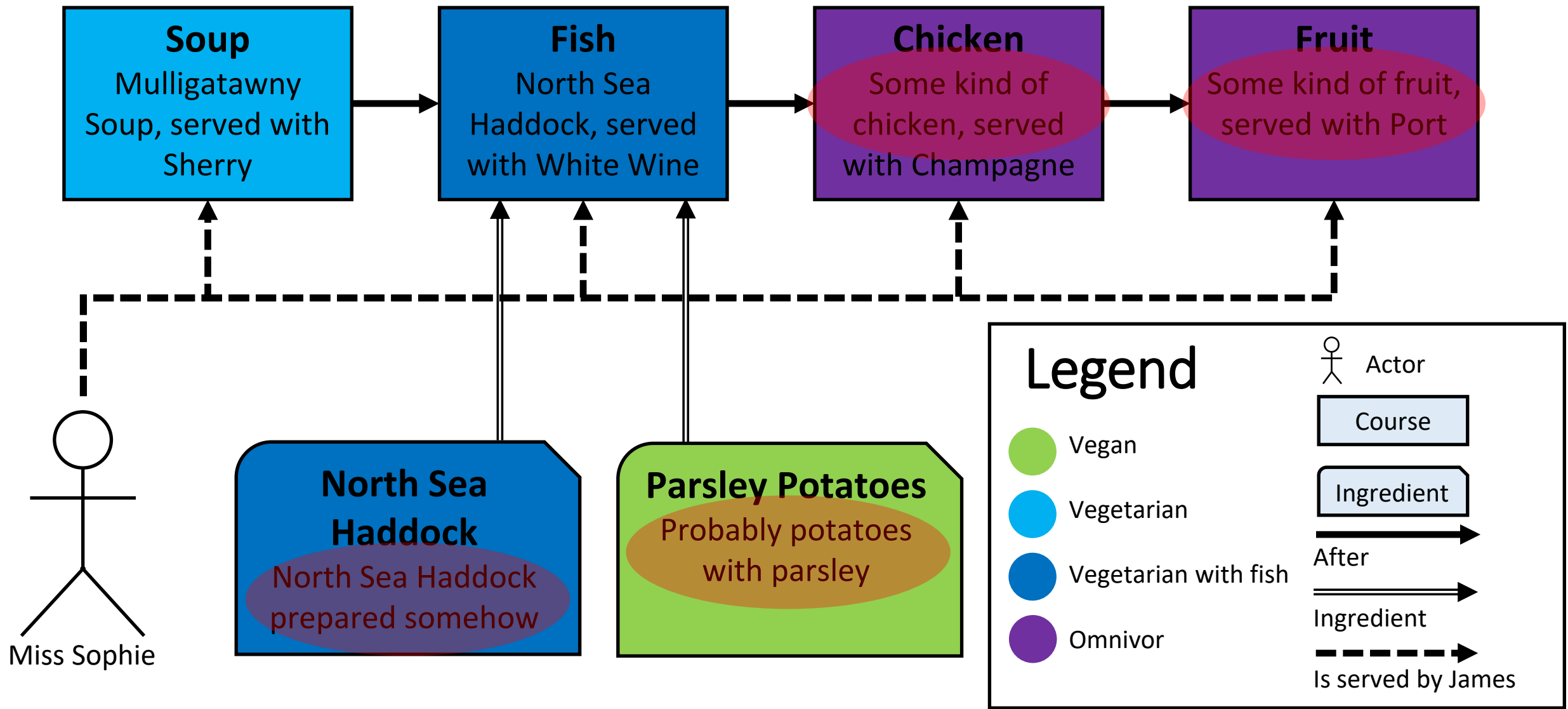
# DINNER FOR ONE 1.1

- Farben und Formen besser zu verstehen
- Elemente haben unterschiedliche Informationstiefe
- (Unübersichtlich)

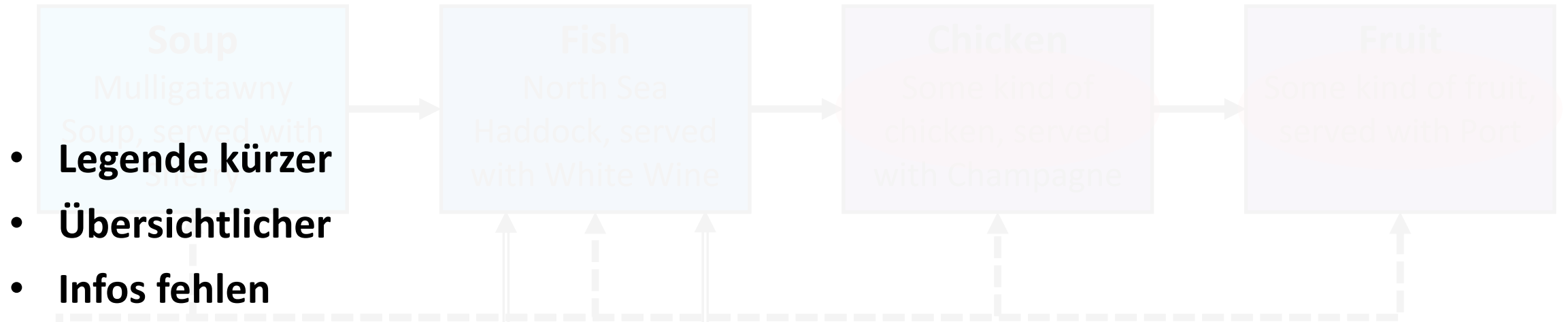
Also: Infos Bündeln; Elemente mit Beschreibungen versehen



# DINNER FOR ONE 1.2

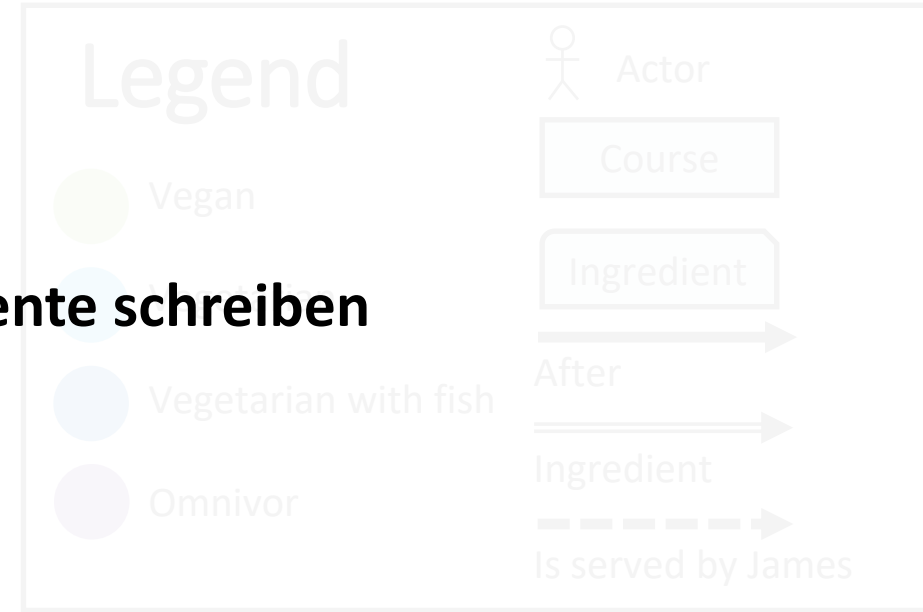
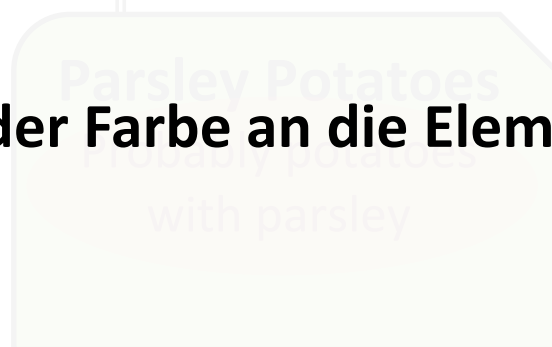
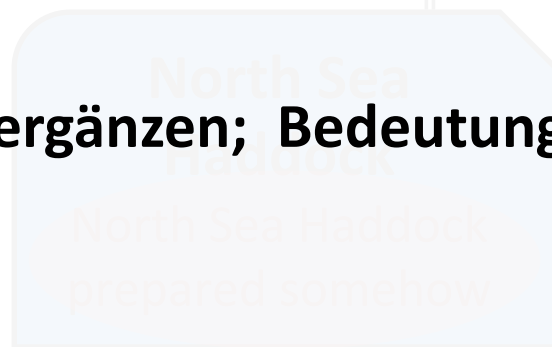


# DINNER FOR ONE 1.2

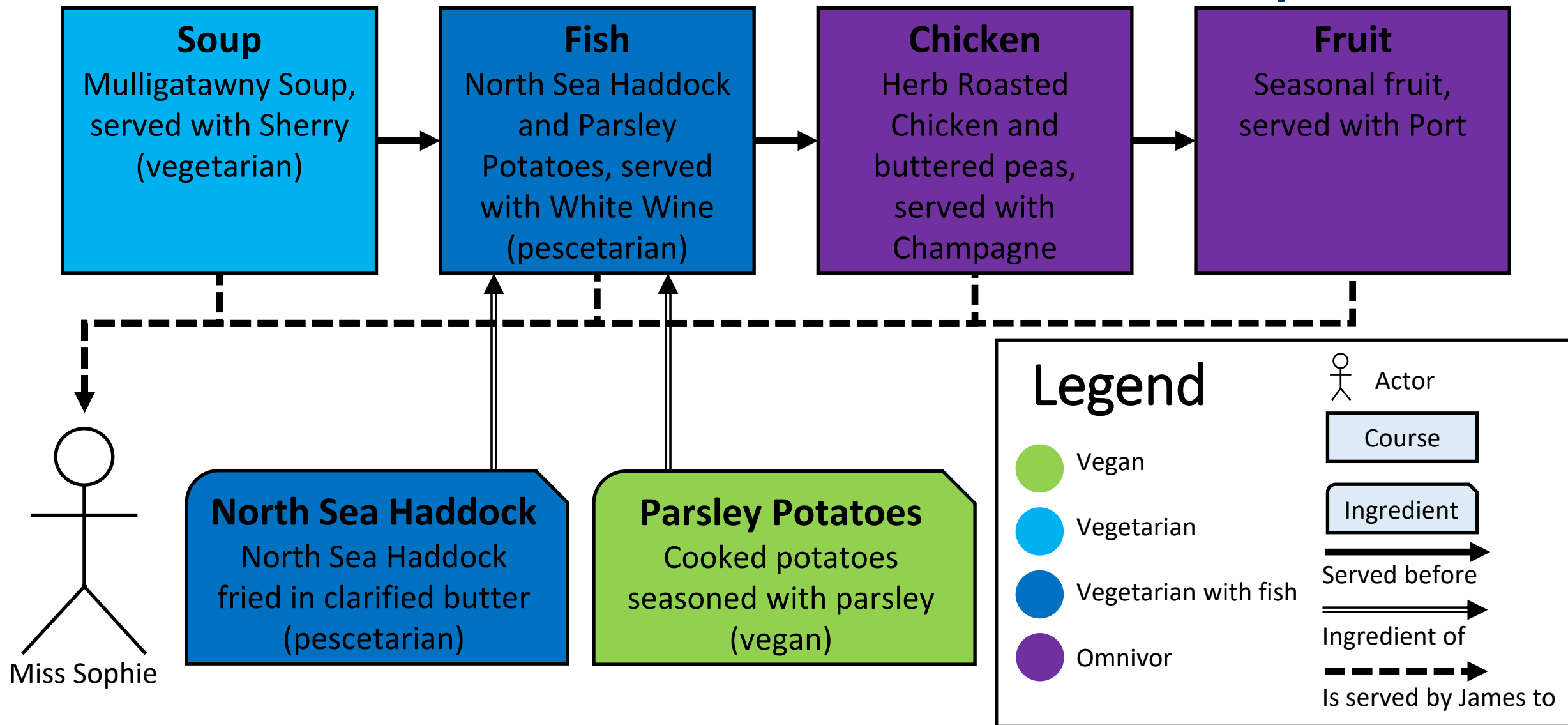


- **Legende kürzer**
- **Übersichtlicher**
- **Infos fehlen**
- **Beschriftung der Pfeile ist zum Teil falsch herum**
- **Farbkennzeichnung nicht immer deutlich**

**Also: Infos ergänzen; Bedeutung der Farbe an die Elemente schreiben**

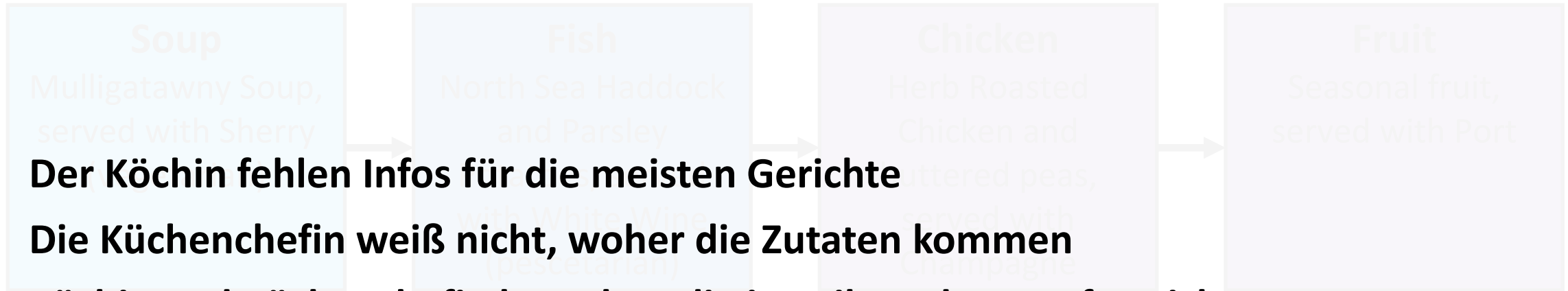


# DINNER FOR ONE 1.3



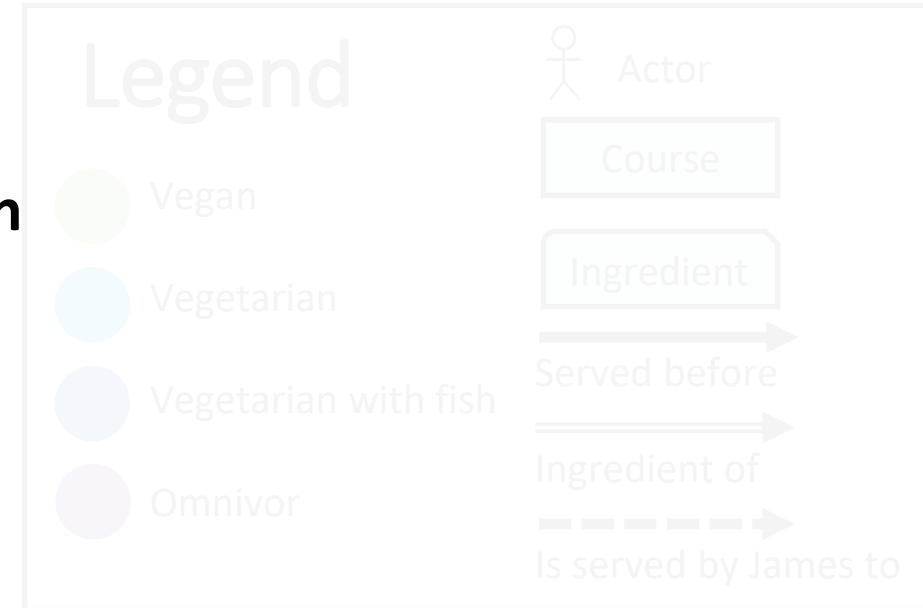
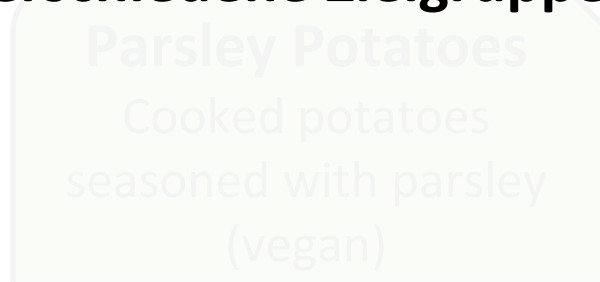
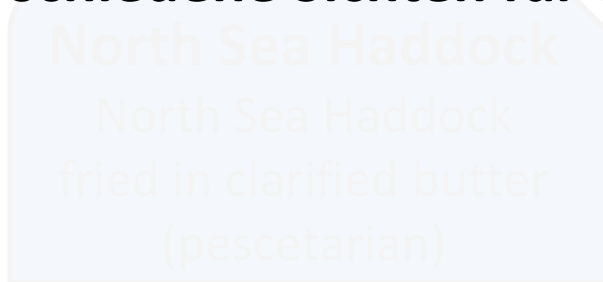


# DINNER FOR ONE 1.3

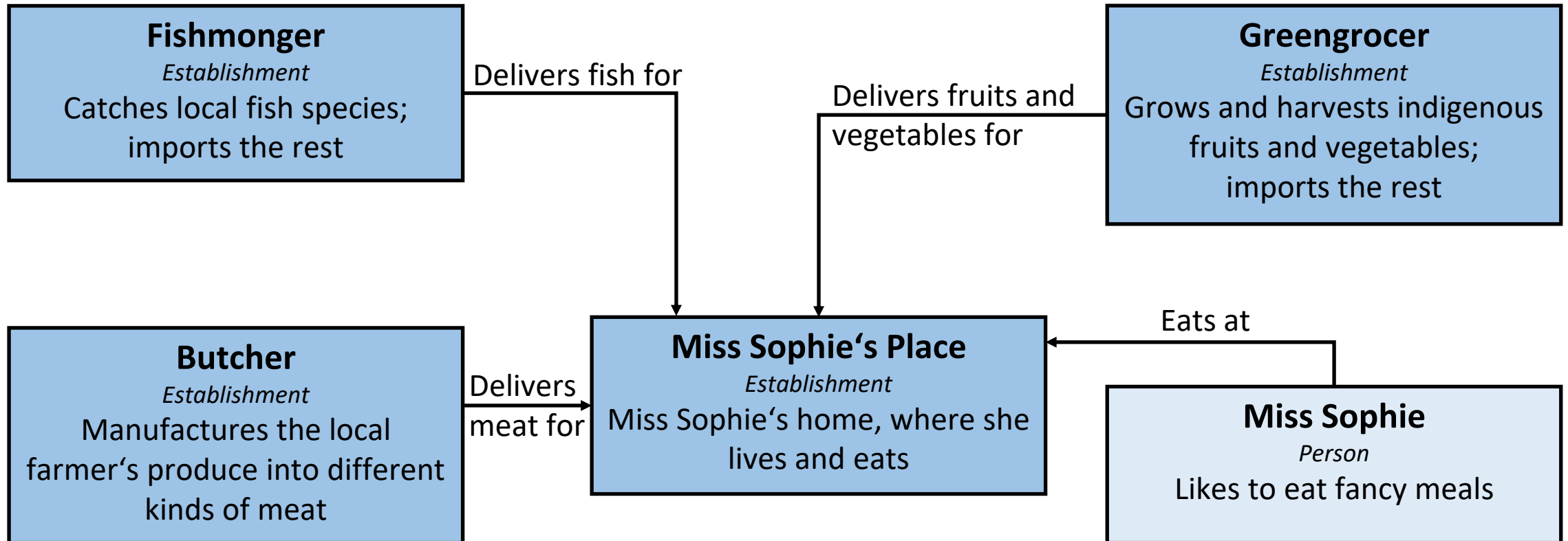


- Der Köchin fehlen Infos für die meisten Gerichte
- Die Küchenchefin weiß nicht, woher die Zutaten kommen
- Köchin und Küchenchefin brauchen die jeweils anderen Infos nicht
- Infos verloren gegangen

**Also: Verschiedene Sichten für verschiedene Zielgruppen**



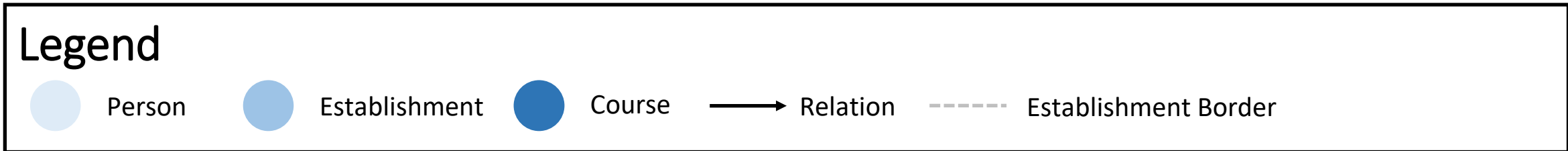
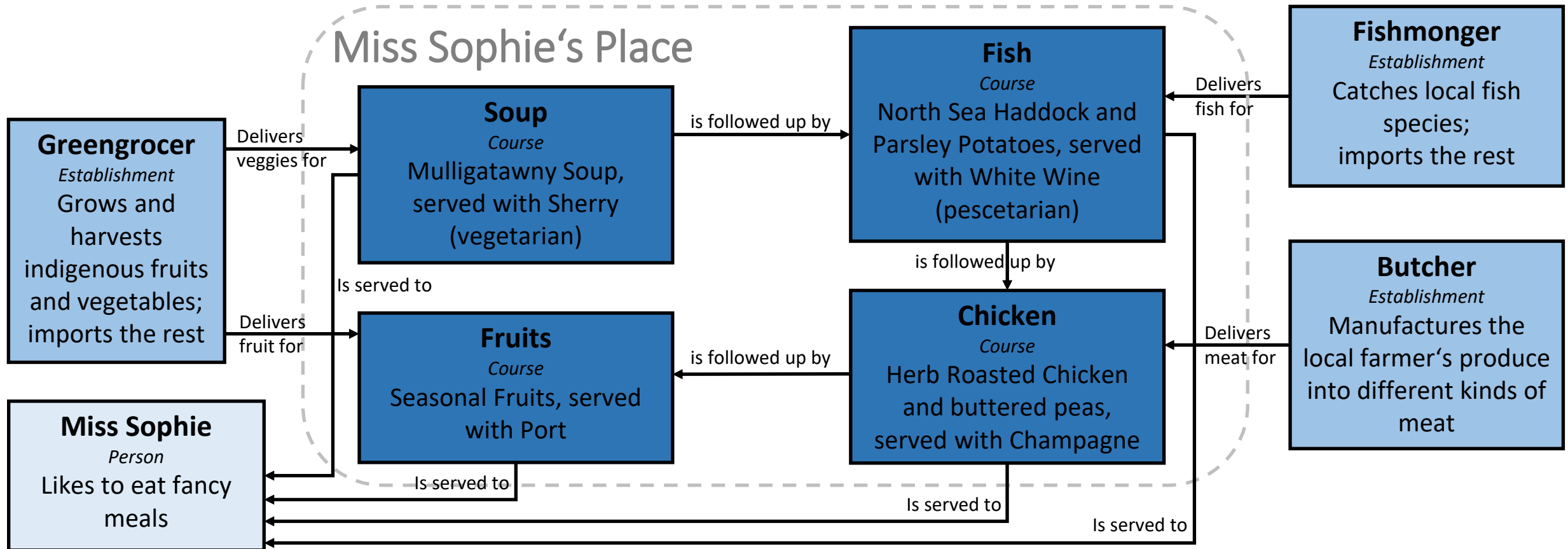
# SICHT FÜR KÜCHENCHEFIN



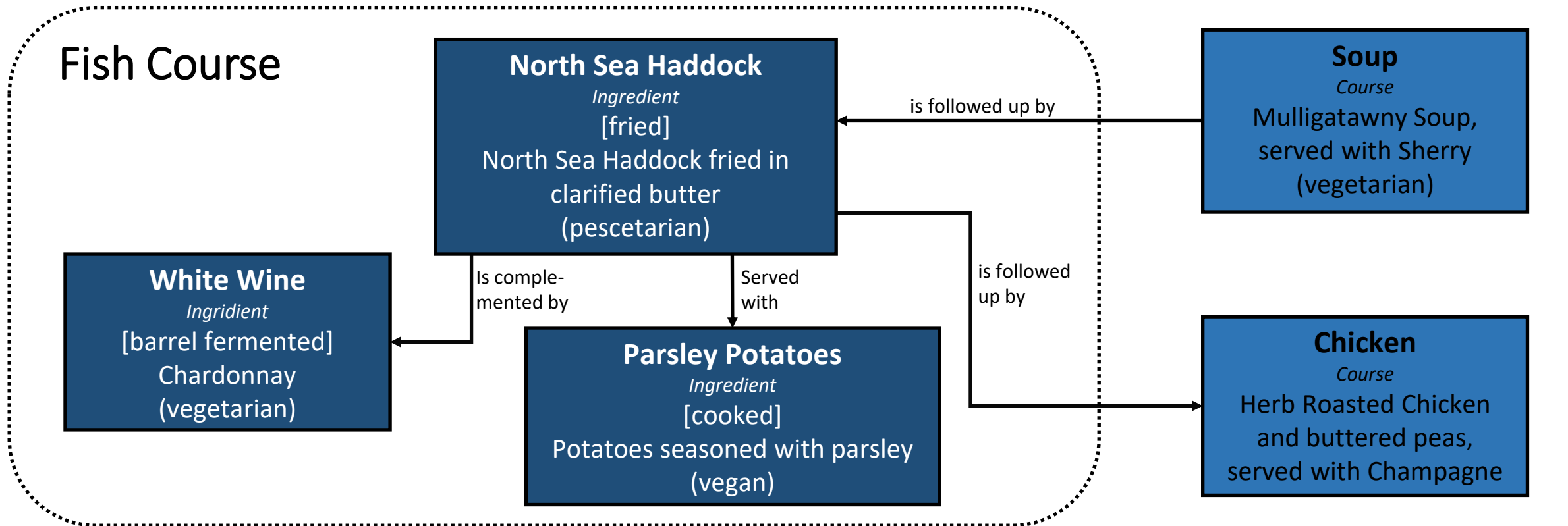
## Legend



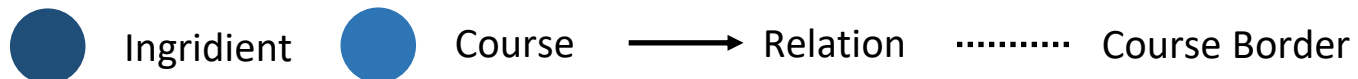
# SICHT FÜR JAMES, (KÜCHENCHEFIN)



# SICHT FÜR KÖCHIN, (JAMES)



## Legend




# Was bedeutet das für Software-Architektur?

- Abstraktion ist wichtiger als Notation
- Verschiedene Zielgruppen brauchen verschiedene Notationen
- Diagramme sollen die Realität widerspiegeln



Common abstractions  
over a common notation



Different stories for  
different audiences



The „Model-Code Gap“

# COMMON ABSTRACTIONS OVER A COMMON NOTATION

- **Einheitliche Abstraktionsebenen** sind ein guter Anfang, wenn einheitliche Notation nicht funktioniert
- **Einheitliches Vokabular** erleichtert Kommunikation und beugt Verständigungsproblemen vor

## C4 Vokabeln:

- Person
- Software System
- Container
- Component

# DIFFERENT STORIES FOR DIFFERENT AUDIENCES

- Gute Doku liefert NUR die Infos, die die Leserin braucht
- Daher: Unterteilung der Architektur in verschiedene Sichten
- Diagramme mit variierendem Detailgrad

## C4 Diagramme:

- System **C**ontext
- **C**ontainer
- **C**omponent
- ~~C~~ode



# THE „MODEL-CODE GAP“

- Dokumentation nützt nichts, wenn sie nicht die Realität abbildet
- Technologie-Entscheidungen spiegeln einen Teil der Realität wider
- Abstrakte Diagramme führen zu Spekulationen; Realistische Diagramme zu bereichernden Diskussionen



Technologie in  
Diagrammen ermöglicht  
realistische Designs

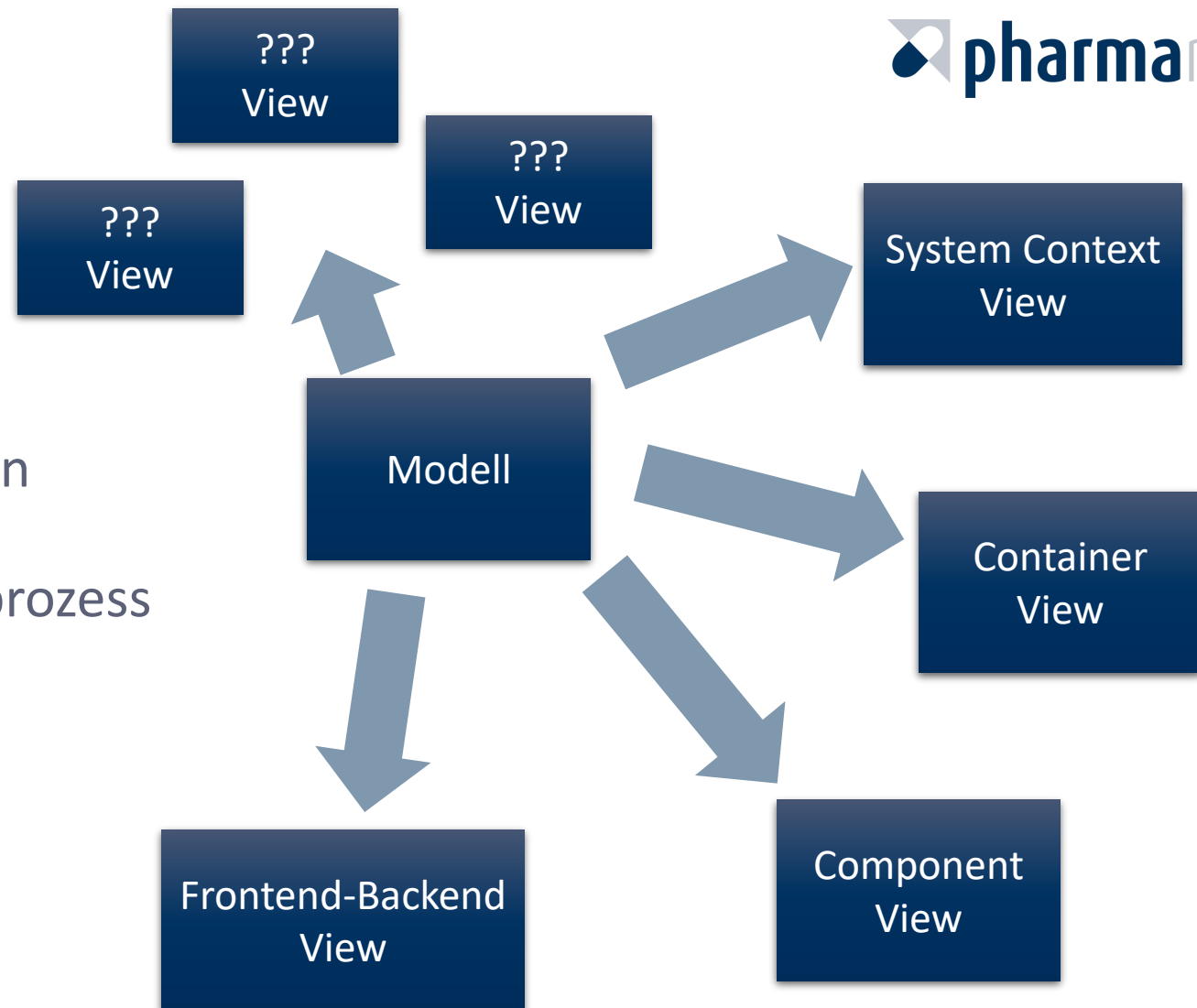
# ZWISCHENFAZIT:

- Doku in Textform ist schwerer verständlich
  - Doku in Bildform ist aufwändiger zu pflegen
  - Dokumentieren ist zeitaufwändig
  - Dokumentieren ist kein fester Bestandteil der Ausbildung
  - Diagramme und Formulierungen sind nicht einheitlich
  - Infos die man sucht sind schwer zu finden
- 😊 ➤ Keine Textform
  - 🚫 ➤ Sogar schlimmer mit C4
  - 🚫 ➤ Einzelne Diagramme erzeugen auch
  - 🚫 ➤ Kästchen malen auch nicht
  - 😊 ➤ Einheitliche Struktur und klar definiertes Vokabular
  - 😊 ➤ Verschiedene Sichten

# Wo ist jetzt der Mindblow?

# DIAGRAMS-AS-CODE

- Architektur
- ~~Diagramme~~ in einer DSL oder Programmiersprache beschreiben
  - Einfach(er) in den Programmierprozess einzubinden
  - Geringere Hürde beim Erstellen





```
1 // model
2 ECommerce = softwareSystem "eCommerce Platform" "internally developed platform to provide an online shop for pharmaceuti
3
4 ECommerce_shop = container "eCommerce Shop" "Shop to buy pharmaceutical goods" "Vue" {
5     ECommerce_shop_backend = component "eCommerce Shop Backend" "Specialized backend for shop requests" ".NET applicati
6     ECommerce_shop_frontend = component "eCommerce shop Frontend" "Shop frontend for customers" "Vue"
7 }
8 ECommerce_vendorPortal = container "Vendor Portal UI" "Frontend for vendors to maintain products and offers" "Vue"
9 ECommerce_portal = container "Operator Portal UI" "Interface for front- and backoffice users to maintain eCommerce-dat
10 ECommerce_backend = container "eCommerce Backend" "backend services for core eCommerce functionality" ".NET applicatio
11     ECommerce_backend_core = component "eCommerce Backend Core" "Manages core business entities like customers, products
12     ECommerce_backend_graphql = component "eCommerce Backend GraphQL Services" "Provides GraphQL queries and mutations;
13 }
14 ECommerce_database = container "eCommerce Database" "database service for core and additional services" "PostgreSQL"
15 }
16
17 // relationships
18 ECommerce_buyer -> ECommerce_shop_frontend "browses"
19 ECommerce_vendor_user -> ECommerce_vendorportal "maintains vendor data via"
20 ECommerce_portal_user -> ECommerce_portal "maintains eCommerce data via"
21
22 ECommerce -> idm "uses for authentication and authorization"
23 ECommerce_shop -> ECommerce_backend "sends CRUD operations regarding orders to" "GraphQL (HTTPS)"
24 ECommerce_shop_frontend -> ECommerce_shop_backend "calls single endpoint of"
25 ECommerce_shop_backend -> ECommerce_backend_graphql "sends requests to"
26 ECommerce_portal -> ECommerce_backend "sends CRUD operations regarding product and customer data to" "Rest API (HTTPS)"
```

```
21 ECommerce -> idm "uses for authentication and authorization"
22 ECommerce_shop -> ECommerce_backend "sends CRUD operations regarding orders to" "GraphQL (HTTPS)"
23 ECommerce_shop_frontend -> ECommerce_shop_backend "calls single endpoint of"
24 ECommerce_shop_backend -> ECommerce_backend_graphql "sends requests to"
25 ECommerce_portal -> ECommerce_backend "sends CRUD operations regarding product and customer data to" "Rest API"
26 ECommerce_vendorPortal -> ECommerce_backend_core "sends CRUD operations regarding product data to" "Rest API"
27 ECommerce_backend_graphql -> ECommerce_backend_core "forwards requests to"
28 ECommerce_backend_core -> tmsSystem "sends orders to"
29 ECommerce_backend_core -> ECommerce_database "stores business data in"
30
31
32 // views
33 systemContext ECommerce "ECommerce_SystemContext" {
34     title "ECommerce - System Context"
35     include *
36     autoLayout
37 }
38
39 container ECommerce "ECommerce_Container" {
40     title "ECommerce - Container"
41     include *
42     autoLayout
43 }
44
```

## Architektur

- ~~Diagramme~~ in einer DSL oder Programmiersprache beschreiben
- Einfach(er) in den Programmierprozess einzubinden
- Geringere Hürde beim Erstellen
- Separation of concerns: Diagramme sind getrennt von Modell
- Single Source of Truth: Geringere Fehleranfälligkeit
- DRY: Weniger Arbeitsaufwand



# FAZIT:

- Doku in Textform ist schwerer verständlich
  - Doku in Bildform ist aufwändiger zu pflegen
  - Dokumentieren ist zeitaufwändig
  - Dokumentieren ist kein fester Bestandteil der Ausbildung
  - Diagramme und Formulierungen sind nicht einheitlich
  - Infos die man sucht sind schwer zu finden
- ☺ ➤ Keine Textform
  - ☺ ➤ Nicht mehr!
  - ☺ ➤ Nicht mehr!
  - ☺ ➤ Programmieren schon!
  - ☺ ➤ Einheitliche Struktur und klar definiertes Vokabular
  - ☺ ➤ Verschiedene Sichten

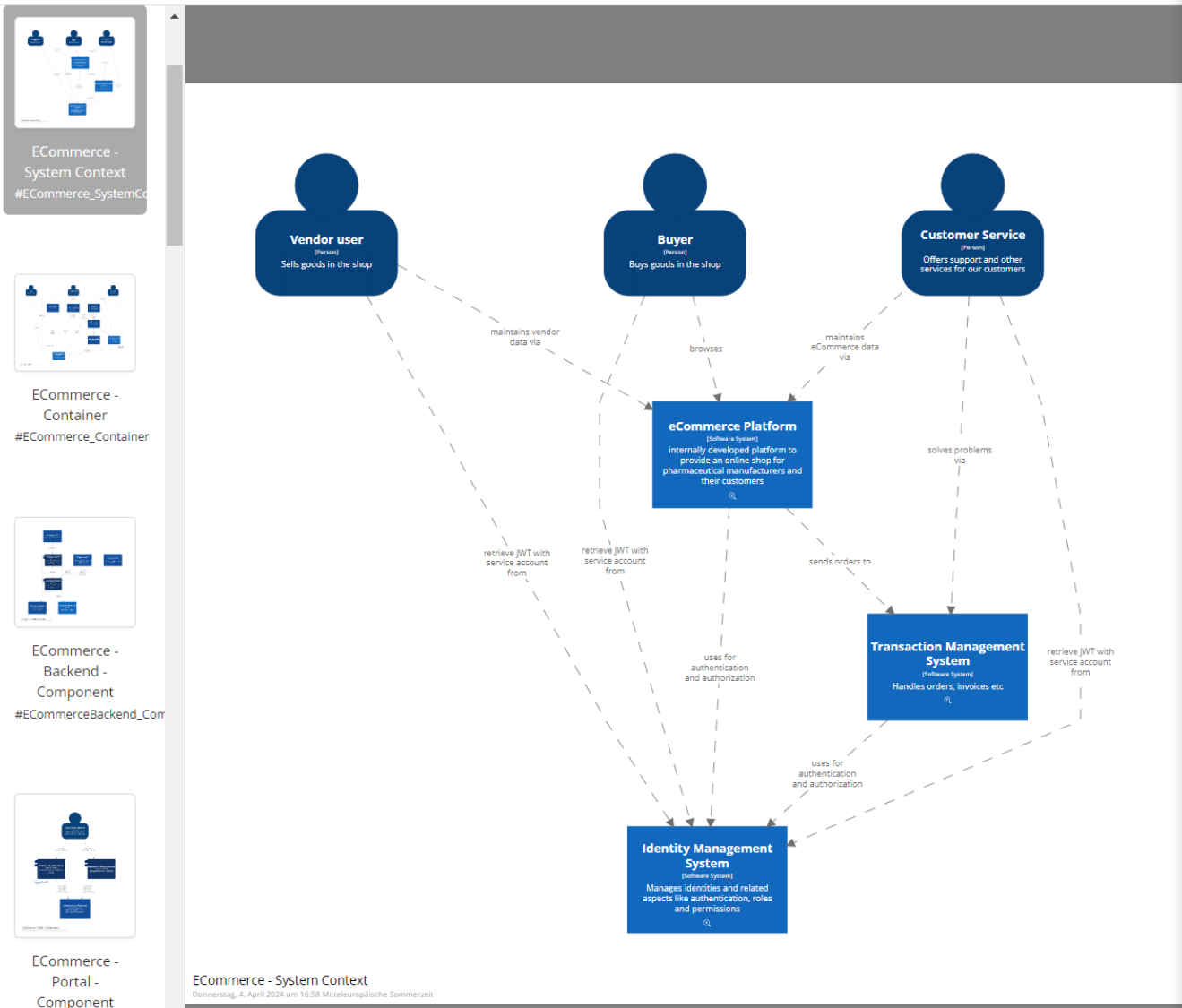
# Wie wird die C4-Doku erstellt?

# WIE WIRD DIE C4-DOKU ERSTELLT?

- Strukturizr: <https://docs.structureizr.com/lite>



Structureizr



```

1 // model
2 ECommerce = softwareSystem "eCommerce Platform" "internally developed platform to provide an online shop for p
3
4 ECommerce_shop = container "eCommerce Shop" "Shop to buy pharmaceutical goods" "Vue" {
5     ECommerce_shop_backend = component "eCommerce Shop Backend" "Specialized backend for shop requests" ".NET
6     ECommerce_shop_frontend = component "eCommerce shop Frontend" "Shop frontend for customers" "Vue"
7 }
8 ECommerce_vendorPortal = container "Vendor Portal UI" "Frontend for vendors to maintain products and offers"
9 ECommerce_portal = container "Operator Portal UI" "Interface for front- and backoffice users to maintain eCo
10 ECommerce_backend = container "eCommerce Backend" "backend services for core eCommerce functionality" ".NET
11     ECommerce_backend_core = component "eCommerce Backend Core" "Manages core business entities like customers
12     ECommerce_backend_graphql = component "eCommerce Backend GraphQL Services" "Provides GraphQL queries and m
13 }
14 ECommerce_database = container "eCommerce Database" "database service for core and additional services" "Pos
15 }
16
17 // relationships
18 ECommerce_buyer -> ECommerce_shop_frontend "browses"
19 ECommerce_vendor_user -> ECommerce_vendorportal "maintains vendor data via"
20 ECommerce_portal_user -> ECommerce_portal "maintains eCommerce data via"
21
22 ECommerce -> idm "uses for authentication and authorization"
23 ECommerce_shop -> ECommerce_backend "sends CRUD operations regarding orders to" "GraphQL (HTTPS)"
24 ECommerce_shop_frontend -> ECommerce_shop_backend "calls single endpoint of"
25 ECommerce_shop_backend -> ECommerce_backend_graphql "sends requests to"
26 ECommerce_portal -> ECommerce_backend "sends CRUD operations regarding product and customer data to" "Rest API
27 ECommerce_vendorPortal -> ECommerce_backend_core "sends CRUD operations regarding product data to" "Rest API
28 ECommerce_backend_graphql -> ECommerce_backend_core "forwards requests to"
29 ECommerce_backend_core -> tmsSystem "sends orders to"
30 ECommerce_backend_core -> ECommerce_database "stores business data in"
31
32 // views
33 systemContext ECommerce "ECommerce_SystemContext" {
34     title "ECommerce - System Context"
35     include *
36     autoLayout
37 }
38
39 container ECommerce "ECommerce_Container" {
40     title "ECommerce - Container"
41     include *
42     autoLayout
43 }
44
45 component ECommerce_portal "ECommercePortal_Container" {
  
```

c4-workshop workspace-1.dsl

workspace-1.dsl

1 file committed: throwaway (moments ago)

44:1 CRLF UTF-8 2 spaces

# WIE WIRD DIE C4-DOKU ERSTELLT?

- Structurizr: <https://docs.structurizr.com/lite>
- Structurizr-Site-Generatr: <https://github.com/avisi-cloud/structurizr-site-generatr/>



# Structurizr

# Welchen Problemen sind wir begegnet?

- Was ist was? (System, Container, Component)
  - Wie funktioniert C4 für Microservices?
  - Wie dokumentiert man einzelne Software-Sequenzen?
  - Wie dokumentiert man die Infrastruktur?
- Übung und Definitionen angucken
  - Microservices als Software Systems
  - Dynamic Diagrams
  - Deployment Diagrams

# Vielen Dank für Eure Aufmerksamkeit!

Link zu den Slides:





# Gibt es noch Fragen?

Link zu den Slides:



- “The C4 model for visualising software architecture” von Simon Brown
- <https://c4model.com/>
- <https://dev.to/simonbrown/diagrams-as-code-2-0-82k>
- <https://www.georgefairbanks.com/software-architecture/model-code-gap/>
- <https://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf>



Your partner for digital transactions  
in the pharmaceutical industry