FMC Presentation

Model – abstraction of existing/planned system.
Used to communicate, construct, analyze.

Modeling – process of creating a model
– choosing most important facts
– leaving others out
This abstraction heavily depends on the purpose of a model and its addressees.

FMC – modeling techique and notation.
– developed to support communication about techical systems

- Insert Fig 1.2
- we should apply:
- Retrieve Info From Others:
- Iterative Modeling – instead of creating acc. model after you coll. all the facts – start with incomplete/wrong model, which shows what you understand so far – let the domain experts correct you.
- Fig 1.3

(2) Share info with others

non domain experts

Fig 1.4

Didactical modeling - use diagrams of the system to transport into to others. Get feedback on comprehensibility, and improve diagrams.

(3) Help with communication

Fig 1.5

Ad-hoc modeling - modeling currently discussed system with pen/paper.
- obtain agreement on common terminology, common level of detail, helps...
- focus discussion

(4) Structure a team around work

Architecture of a system can be captured in a

System map model - shows complete system at a reasonable level of detail.
- shows what every team member is responsible for ~~must know about~~

I like FMC because it is able to do

Abstraction
able to describe conc. archit in many diff levels

Simplicity
few elementary concepts not atomic elements

Universality
- enough power to cover many system types
- not bound to spec. paradigm

Sep of Concerns
Able to express different aspects of a system

Aesthetics
should support proper layout (and easy formation of graphical patterns)

FMC — created to support communic. concepts about inf. proc. systems.

Created by Siegfried Wendt

3 Diagram Types

FMC sees 3 diff aspects in inf. proc. systems:

① Composition Structure – describes a system in terms of active and passive components.

Agents – active components, process information

Channels (storages) – passive components which keep/transport info.

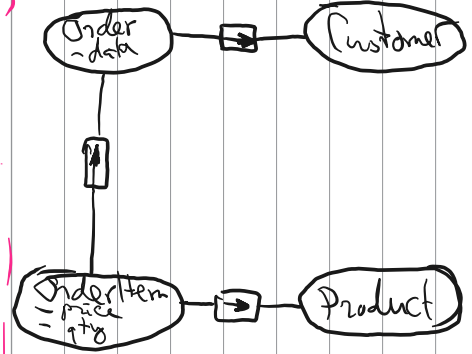
② Behavior of agents that operate on data, and respond to requests which they recieve via channels

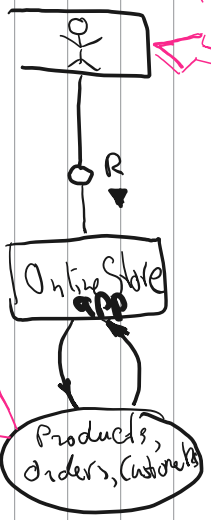
③ Data/value structure – structure of inf. found in channels/storages + relationships between data in diff storages.

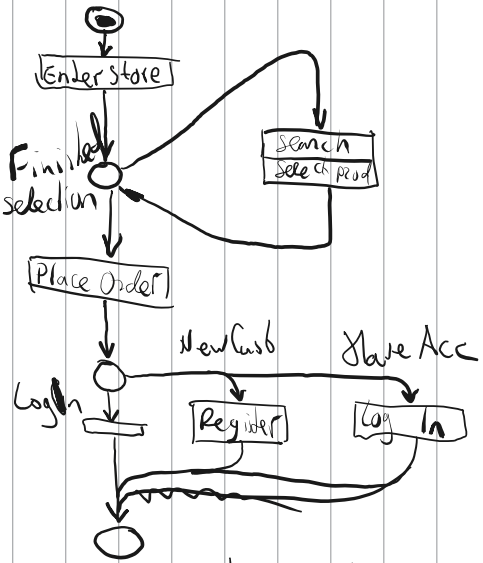
Data/Value Structure

Compositional Structure

Behavior







Entity/Relationship Diagram

Block diagram

Petri Net

FMC diagram type

Compositional Structure

A.

(see p.46 for B)

(Dynamic) System - physical entity made up of components and interactions between them
- usually shows observable behaviour

Fig 4.1

Locations (1)

Locations - possible components of the system. Observed and changed by agents to retrieve/store/transmit information.

2 types

Channels

- TCP connection

Storages

e.g. mailbox ~~yard~~
- counter var

Agents (2)

Agents - active components of the system.
Read, write/modify information to locations.
- Provide the dynamics of the system this way
- Provide/create functionality of the system.

e.g. mail-man who deposits letter into mailbox
- client ~~(web) socket~~ who writes bytes to a web-socket.

Fig 2.2 key

Information ≠ Physical goods

abstract away

Information - data that is of interest to someone / unknown

It is:
1. Passive - does not perform/do anything
2. May be stored, read, modified, sent, received

e.g. hotel reservation, picture of a dog, receipt.

So it is that we model **NOT** physical entites, but informational tasks.

These are part of an...

Information Processing System - is a (dynamic) system that may be understood by its
- state and
- interfaces

B. My def I:

Information Processing System (dynamic) - a system in which the components manipulate information (not physical objects) and are themselves ~~models~~ abstractions of the real world components.
e.g. - travel agency, database system, comp game.

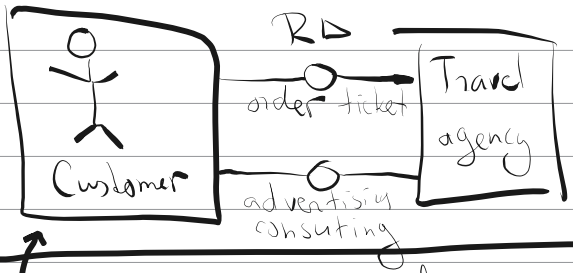


Fig 2.3

Focuses on informational aspects

Refining the model

Model does not help much to know:

1. What does it have to do in order to do its job? (info agency needs)
2. What info/services does it provide a customer?
3. Where is info stored
4. Who accesses this info?

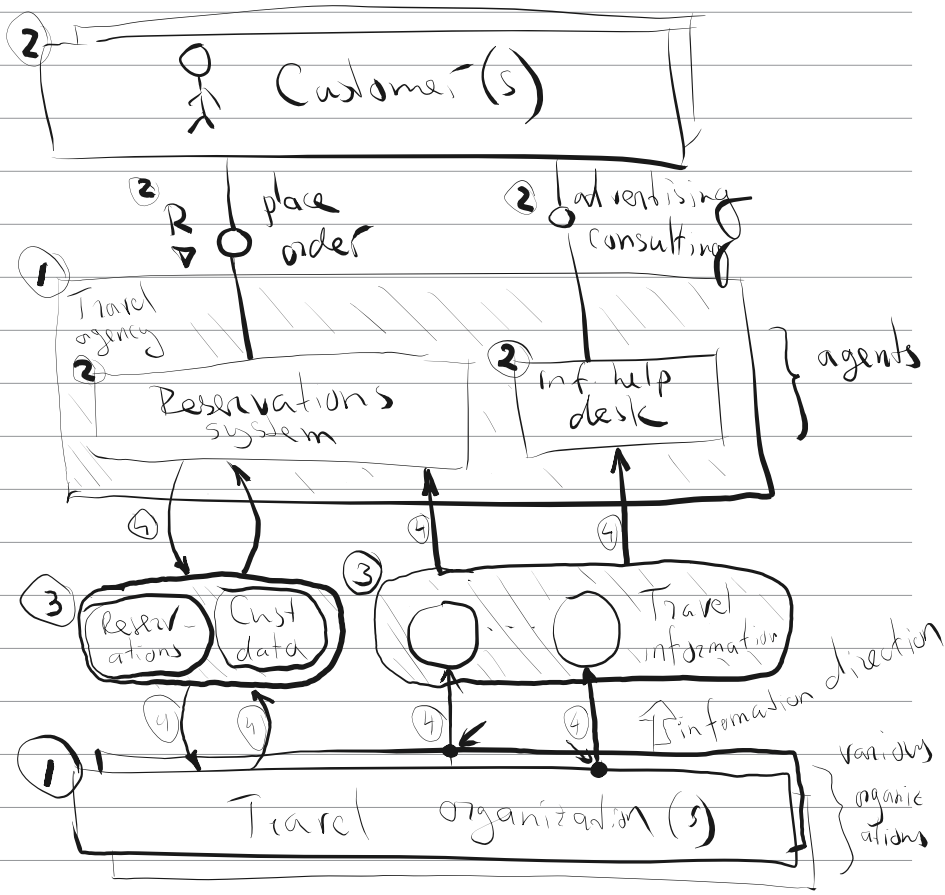


Fig 2.4

Any information processing system can be described by model to integrates:

1. Compositional structure
2. Dynamic structure
3. Set of observable values (structures)
   (e.g. observable information)

Notation: Storages & Agents

Storages are:

~~Nodes~~ ⇒ Locations

→ agents store values in

→ value persists until overwritten

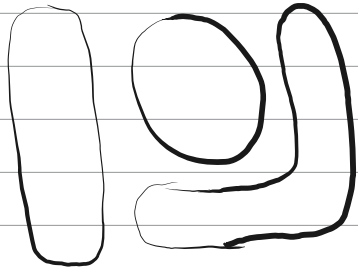
→ e.g. hard-disk, memory, vars, black boards

Agents:

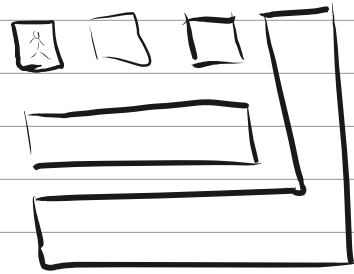
→ access storage when need to write, read, modify information

→ e.g. controller chips, CPU, function, object and its methods, humans.

Storages - rounded



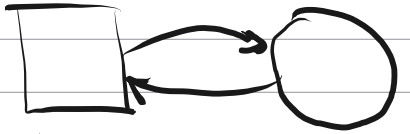
Agents - rectangular



Access types

Read access

write access



Modifying access

Notation:

Agents & Channels

Channels - locations by which agents communicate. Small circles: o

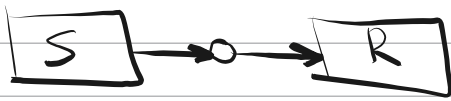
- values on channels are transient
- reciever may read a message, but may not be able to re-read it until sender re-sends

e.g. Protocol bus, TCP conn., telephone connection, air (speaking!)

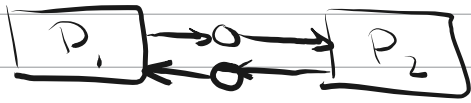
Point-to-point communication

S - sender, R - reciever, Partner - P_i

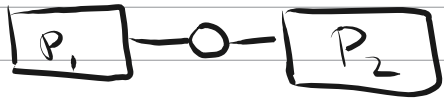
Simplex connection



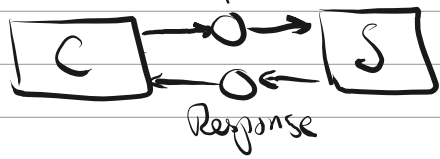
Duplex Connection



Half-duplex conn. (undirected channel) e.g. walkie-talky



Req/Response channel



Abbreviation

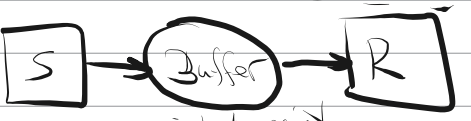
Channels vs. storages?

- Both are valid for communication
- Depends on the problem/need/context

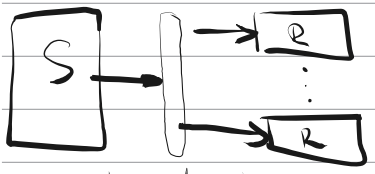
Unbuffered comm. via channels

point-to-point

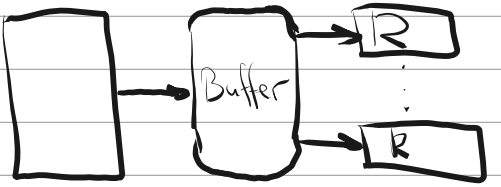
Buffered comm. via storages



point-to-point



broadcast



broadcast

Process of identifying Compositional Structure.

1. Identify agents
2. Identify Storages
3. Identify acess to storage
4. Identify channels

Determine Usefulness of Model

Question:

Can you give each system component a proper name?

- Yes → You're on the right track
- No → Your model is not very useful
  - ambiguous
  - misty
  - non-specific

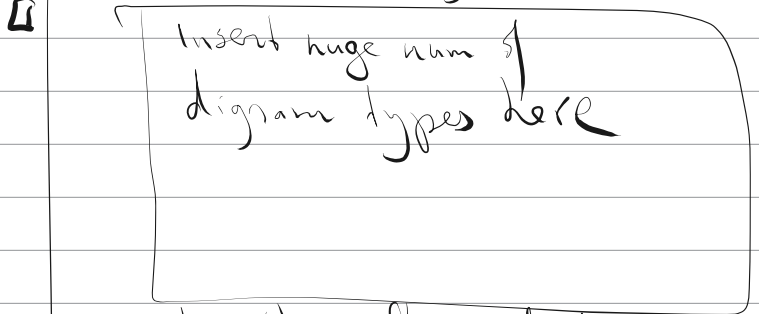
UML vs FMC

- description of software structures
- ~~mode~~ machine-based model processing

- communication of conceptual system structures

UML 2.0 very complex

- instead of dropping and unifying diagram types
+ MORE diagrams were added



- heavily influenced by programing languages
- heavily redundant - 13 diff, almost interchangeable diagram variations:
  - 4 diagram types for behavior modeling.
  - steep learning curve

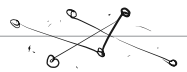
FMC

- Focus on one problem – comm. about conceptual system structures
- Unbiased → [redacted] – omits unnecessary technology-specific constructs
- Simple – avoidance of notational redundancy
  - only 1 diagram type per aspect of the system.
- Optimized for readability and ad-hoc usage (can be used without huge notation manuals and software tools – pen+paper!)
- Can be learned "as you go", presented to non-technical people e.g. customers, domain experts

Different conceptually

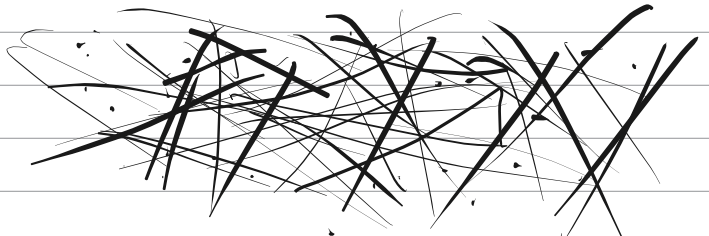
UML – roots in OO methods

– OO became popular in 80–90 to build complex, large systems.



Larger

– But software systems became even larger:



– So "Objects" were too granular now (e.g. millions of them)

– needed "components" and "connectors" diag.

FMC – provides such structures.

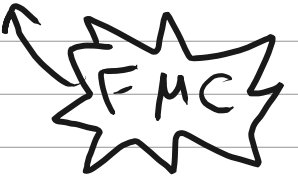
– provides long-lived compositional structures

UML – objects ≡ short-lived complex snapshots – millions created/destroyed during runtime.

Modularization of the code

≠

System Architecture



UML - introduced component diagrams
BUT they are still tied to specific platforms like .NET, CORBA, EJB. → Not enough!
- does not have passive/active seperation

Can be used complementary

Bonus material not covered in the lecture.

NHTs - Non-hierarchical transformations

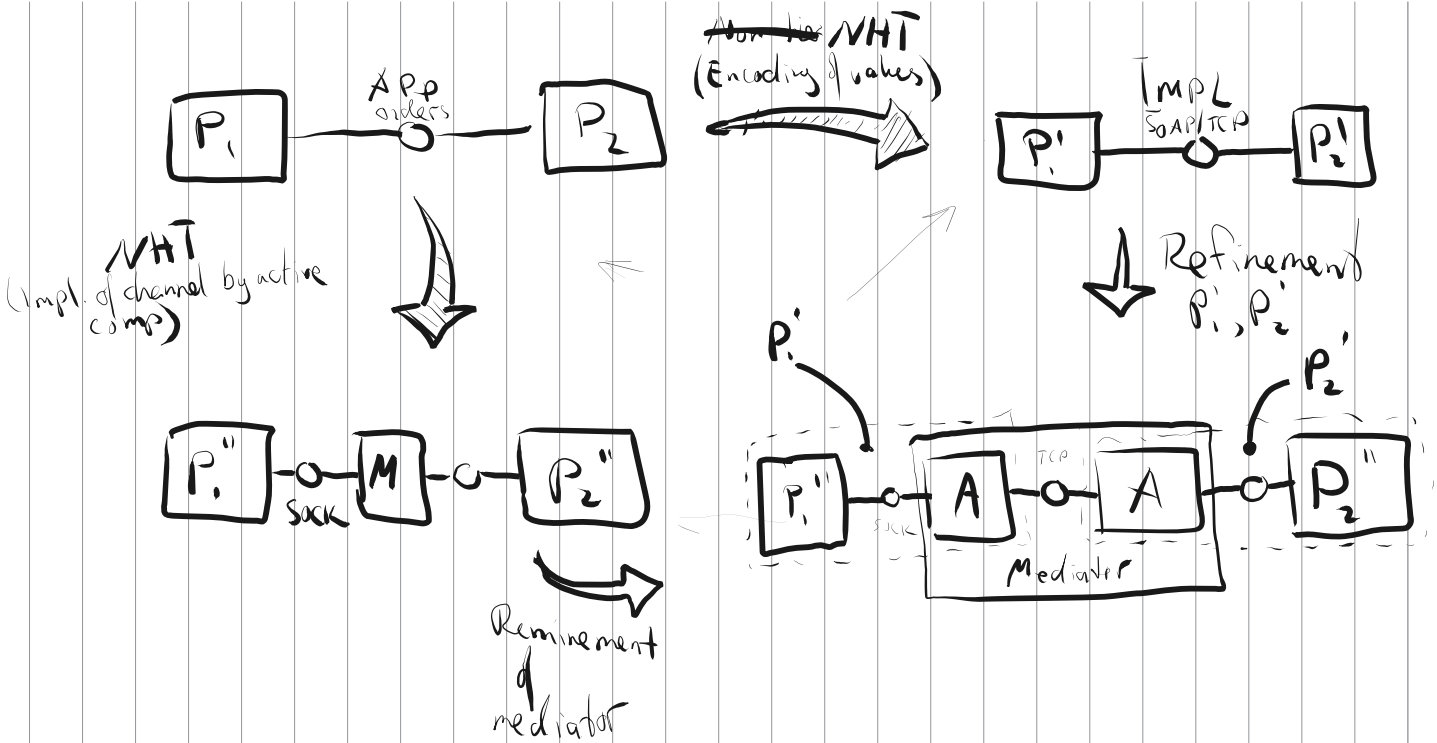
Transformations of one model into another model that cannot be explained by

- strict refinement of value,
- behaviour or compositional structures.

~~Here we will only provide 1 and 3 possible NHTs, as an example:~~

Ex:

P_1 – communication partner 1
P_2 – com. partner 2
APP – dialog (betw partners) at App level – e.g. exchange of orders
IMPL – dialog (betw parts) at impl. level – e.g. orders via SOAP, TCP/IP
sock – dialog with mediator via socket sockets
M – mediator
A – adapter


P1
App orders
P2
VHT
(Encoding of values)
P1'
Impl SOAP/TCP
P2'
VHT
(Impl. of channel by active comp)
Refinement P1', P2'
P1''
SOAP
M
P2''
P1'
P2'
P1''
SOAP
A
TCP
A
P2''
Mediator
Refinement of mediator

Structure of the system is ≠ different from the structure of the system description

Just like

structure of Africa vs structure of a book about AFRICA