Karlsruhe Institute of Technology

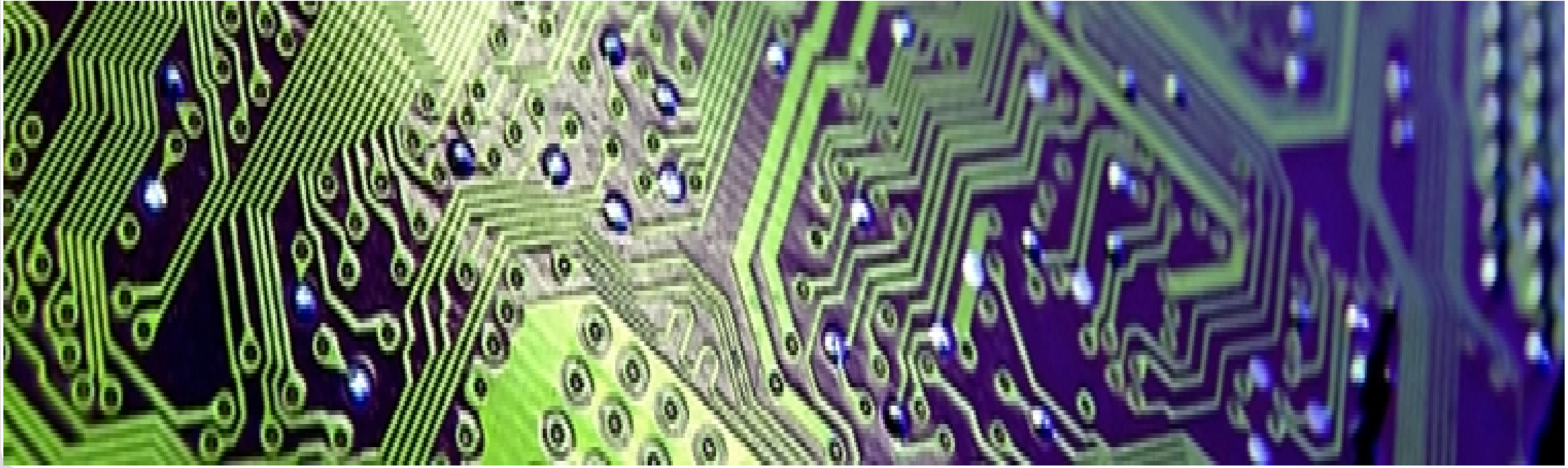
CDNC

nanomat

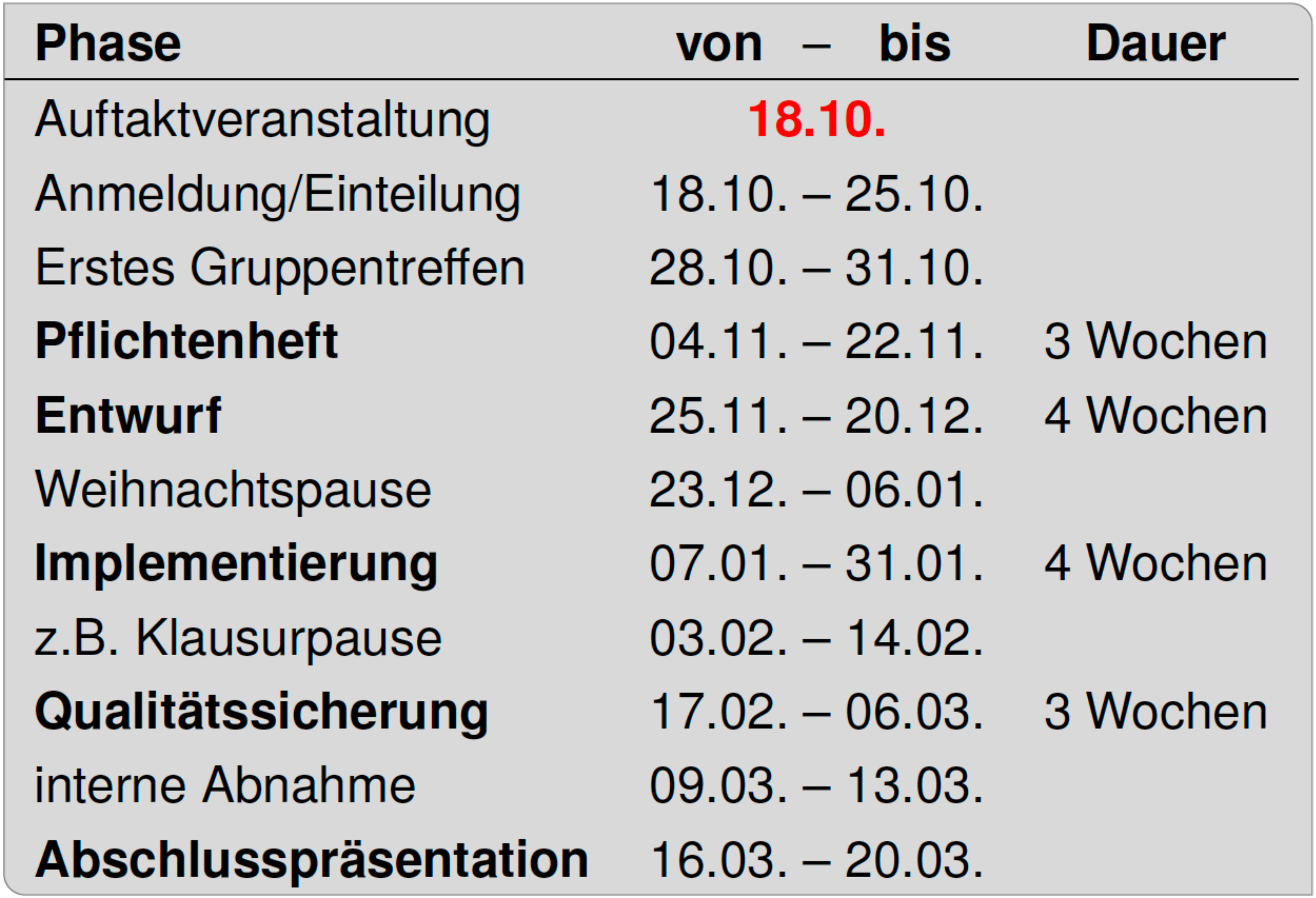
PSE – Neural Network based Image Classification System on Heterogeneous Platforms 2nd Meeting

Dennis Weller & Sarath Mohanachandran Nair

INSTITUTE OF COMPUTER ENGINEERING (ITEC) – CHAIR FOR DEPENDABLE NANO COMPUTING (CDNC)



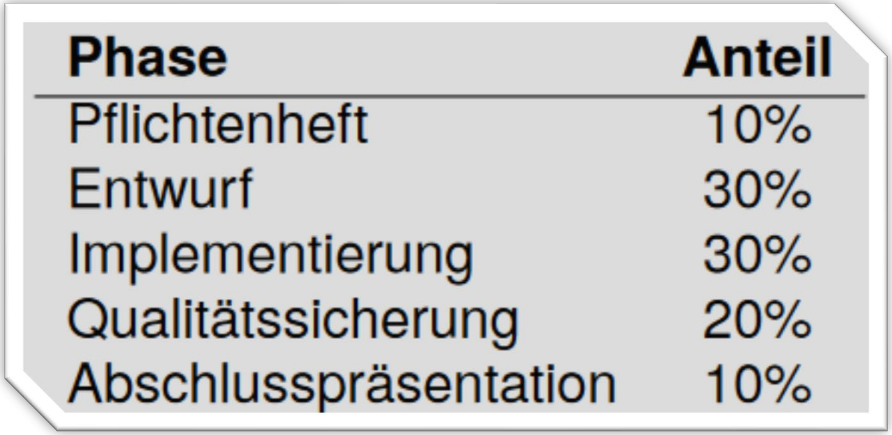
Roadmap



Phase	von – bis	Dauer
Auftaktveranstaltung	18.10.	
Anmeldung/Einteilung	18.10. – 25.10.	
Erstes Gruppentreffen	28.10. – 31.10.	
Pflichtenheft	04.11. – 22.11.	3 Wochen
Entwurf	25.11. – 20.12.	4 Wochen
Weihnachtspause	23.12. – 06.01.	
Implementierung	07.01. – 31.01.	4 Wochen
z.B. Klausurpause	03.02. – 14.02.	
Qualitätssicherung	17.02. – 06.03.	3 Wochen
interne Abnahme	09.03. – 13.03.	
Abschlusspräsentation	16.03. – 20.03.	

Responsible persons

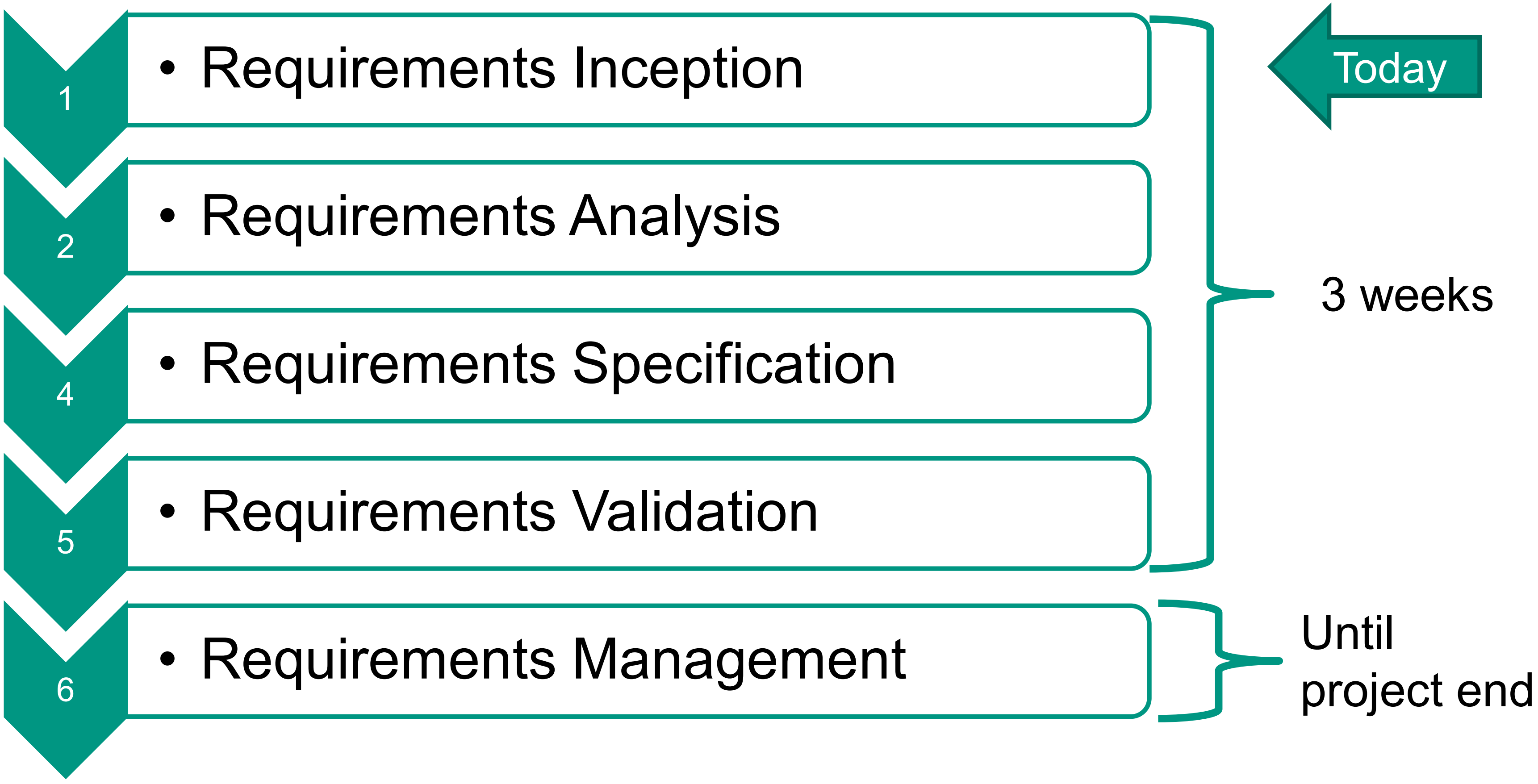
- Responsible persons for each phase
- One person for each phase



Phase	Anteil
Pflichtenheft	10%
Entwurf	30%
Implementierung	30%
Qualitätssicherung	20%
Abschlusspräsentation	10%

Requirements

Requirements Engineering



1
- Requirements Inception

2
- Requirements Analysis

4
- Requirements Specification

5
- Requirements Validation

6
- Requirements Management

Today

3 weeks

Until project end

Description: https://en.wikipedia.org/wiki/Requirements_engineering

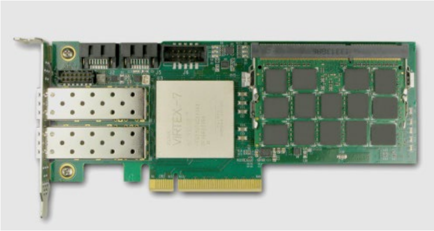
Requirements Engineering

- **Requirements Inception** – Meeting between developers and customers
  - We want to perform image classification
  - We want to run NN on heterogeneous platforms under different operating modes:
    - Highest Performance possible
    - Lowest Power Consumption possible
    - Highest Energy-Efficiency (Performance/(Power Consumption))
  - The heterogeneous platform should consist of an CPU and FPGA
  - We need a GUI from where we run our program

- Requirements Analysis – Requirements identification (Use cases, UML)
- Requirements specification – Formal artifact, only official after validation
- Validation – Checking if the documented requirements meet the needs
- Requirements Management – During the full life cycles of the project

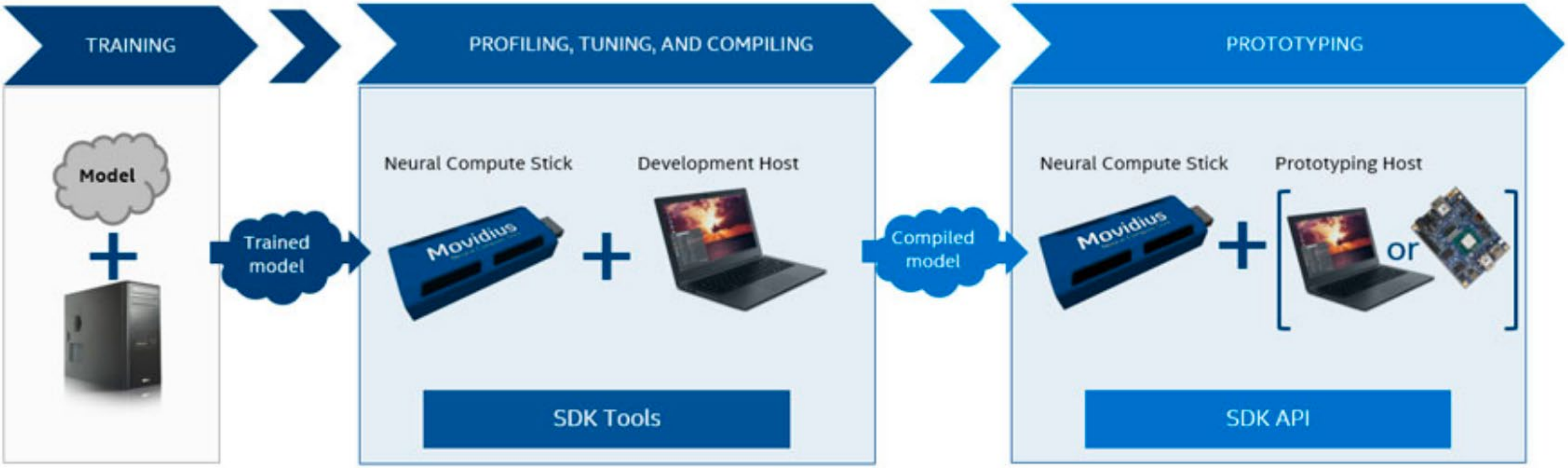
OpenCL - Framework

- Programming of heterogeneous platforms in a C -like fashion
  - Using OpenCL Programming language
  - No hardware knowledge required
- Always 2 components:
  - Host – Main program, control of the acellerator: data transfers, start computations, read results back
  - Acellerator – Acelleration of computations, calculations can be pipelined and parallel
  - Supported platforms: CPU, GPU, FPGA (most recently), DSP
- FPGA support
  - OpenCL compatibility provided since recent years (Science 2012 for Altera FPGAs, 2014 for Xilinx FPGAs)



Intel Movidius Neural Compute Stick

- Neural Compute SDK
  - Neural network frameworks: TensorFlow, Caffe
  - Compile, profile, and validate (check) DNNs
  - Neural Compute API (NCAPI) using Python or C
  - Accesses the neural compute device hardware to accelerate DNN inferences via the NCAPI



Must Requirements

- Backend:
  - Image Classification of at least one **pre-trained** Deep Neural Network Model (AlexNet, GoogleNet)
  - Interface classes from Host PC (Desktop PC with Ubuntu OS) to Accelerator (FPGA)
  - Class for Prediction of Performance (Bandwidth, FLOPS), Power Consumption for each accelerator and the implemented NN
  - Dispatching computationally expensive tasks to the accelerators (FPGA) according to the mode of operation and based on the assessment of the performance/power consumption of the prediction class
  - Interface for controlling other system (send images and get back results)

- Frontend:
  - GUI run on the Ubuntu system
  - Choose Operation Mode: High Performance, Low Power Consumption, High Energy Efficiency
  - Choose Input Image
  - Output Results

Can Requirements

- Backend:
  - Multiple Deep Neural Network Model
  - Other Accelerators:
    - Neural Compute Stick
    - GPU (on-board)
    - Other FPGAs with OpenCL support (Altera 5SGXEA7, Xilinx ADM-PCIE-7V3**)**
    - FPGA in the Cloud (Amazon Web Services)
  - Movidius NCS USB stick can also be attached to the DE1-SoC instead of the host PC
  - Implement Learning Algorithms (Supervised/Unsupervised Learning; Transfer Learning)
  - Other modes of operation

- Frontend:
  - Choose any NN type
  - Deploy training of an arbitrary NN or carry out transfer learning of the already implemented NN
  - Illustrate Neural Network Topology
  - Beautify Output Results

Non-functional Requirements

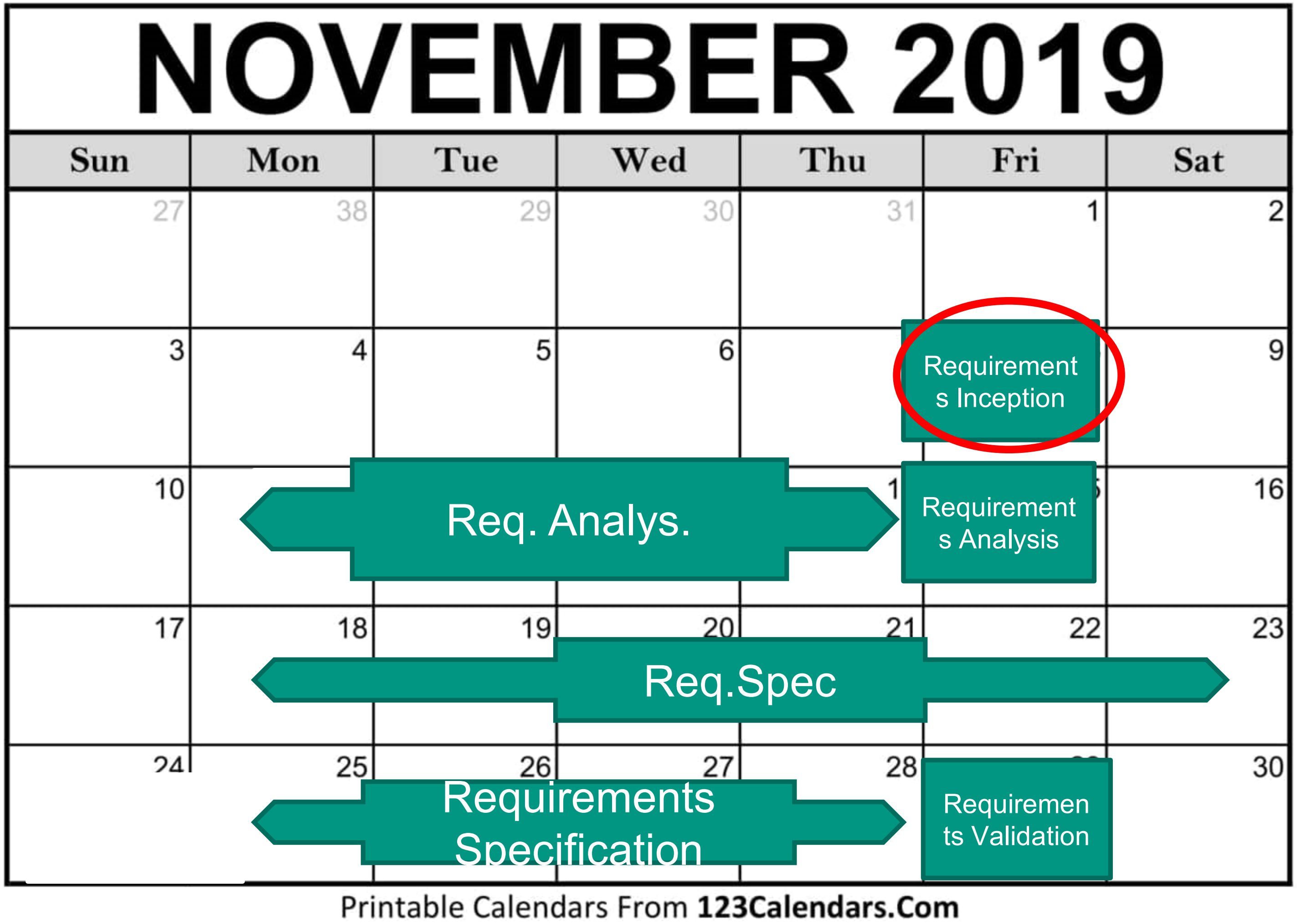
- Object-Oriented Design (40-80 classes without interfaces)
- 10k LOC
- Design in UML
- Design Pattern: Model-View-Controller (MVC)
- Implementation in C++ recommended, other languages can be included
- Quality Assurance (Gcov, Lcov, catch2, …)
- Product environment: Lab PC at CDNC institute
- Operating System: Xubuntu 18.04

Criteria of Dimarcation

What the software should not do…

- No real time requirements
- Product not developed for mobile platforms

Milestones



NOVEMBER 2019

Sun	Mon	Tue	Wed	Thu	Fri	Sat
27	38	29	30	31	1	2
3	4	5	6			9
10				1	5	16
17	18	19	20	21	22	23
24	25	26	27	28		30

Requirements Inception

Req. Analys.

Requirements Analysis

Req.Spec

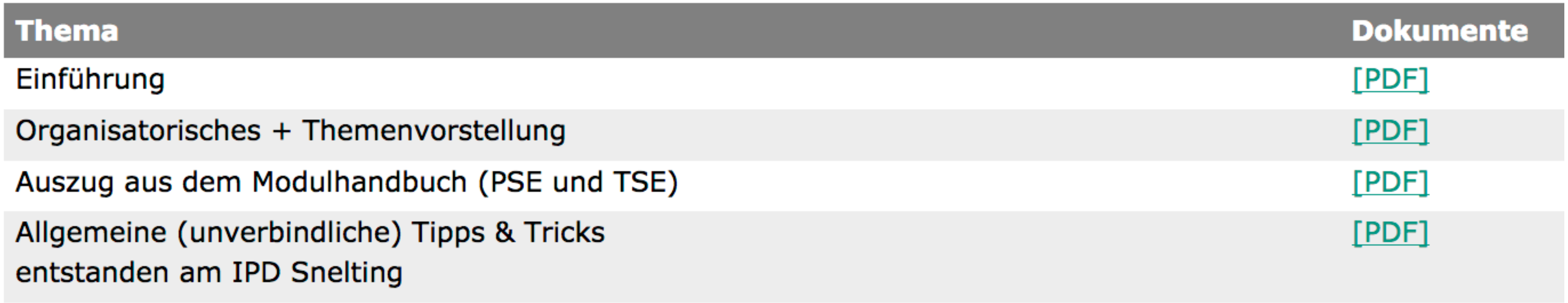
Requirements Specification

Requirements Validation

Printable Calendars From **123Calendars.Com**

Next steps

- Important Documents: https://pp.ipd.kit.edu/lehre/WS201920/pse/

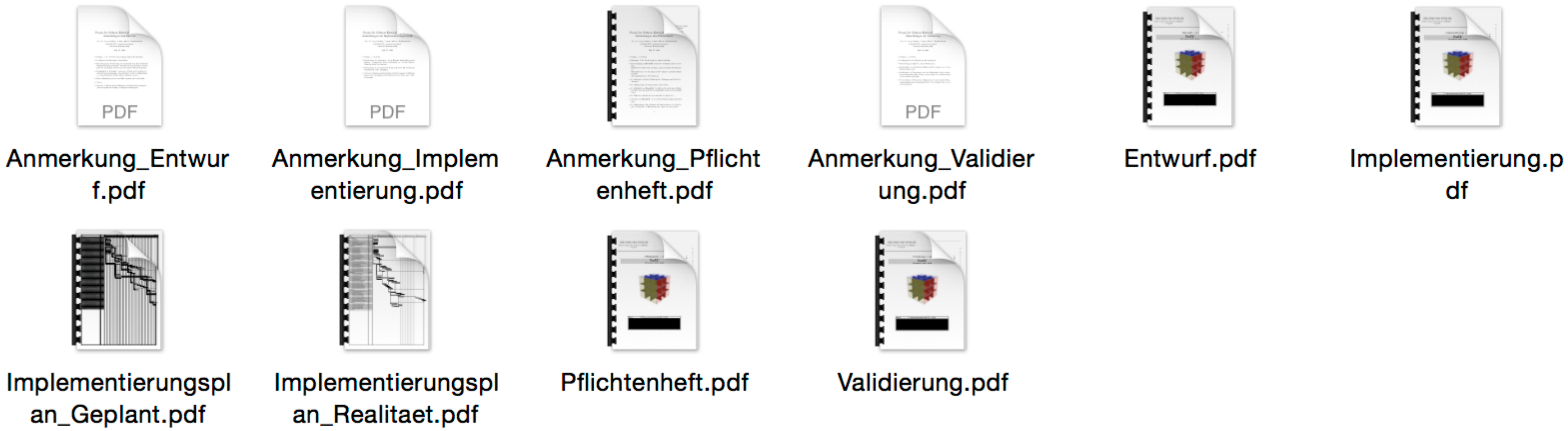


Thema	Dokumente
Einführung	[PDF]
Organisatorisches + Themenvorstellung	[PDF]
Auszug aus dem Modulhandbuch (PSE und TSE)	[PDF]
Allgemeine (unverbindliche) Tipps & Tricks entstanden am IPD Snelting	[PDF]

Recommended Materials

- Vorlesung Software Technik I
  - Folien auf Ilias
  - Folien, Übungsblätter & Co

- Musterdokumente:


Anmerkung_Entwurf.pdf
Anmerkung_Implementierung.pdf
Anmerkung_Pflichtenheft.pdf
Anmerkung_Validierung.pdf
Entwurf.pdf
Implementierung.pdf
Implementierungsplan_Geplant.pdf
Implementierungsplan_Realitaet.pdf
Pflichtenheft.pdf
Validierung.pdf

Good luck!