Övning 3 -Prototyping (PROTO1/2), Funktionella krav (Lau:3-5)

1. Plan prototyping in your own project (for elicitation and for validation) by considering the aspects *Purpose*, *Prototype Scope, Prototype Media*, *Prototype Use* [PROTO1].
   a. what you want to learn from prototyping,
   b. what kind of prototype is needed, and
   c. how should this prototype be used to gain these learnings; with whom, which review method and usage environment?

2. Which functional styles were used in the Shipyard (Ch 11), in the Bruel & Kjaer case (Ch 14), and in the Tax Payers' case (Ch 15)? [Lau övn 3.2]:
   Leta efter stilar från alla kapitel 2-5!
   Danish Shipyard: Kap 5.2.1.1.1, Kap 5.4, Kap 5.2.4, Fig 11.1, Fig 11.10
   Bruel & Kjaer: R-4
   Tax Payers': R2, R3,

Hitta gärna fler egna exempel på kravtyper i specarna!

3. For your own project (or Ticket Machine, page 543) [Lau övn 3.7]:
   b) Write **Task descriptions** (Ch 3.6) for two to three user tasks. If suitable, write Tasks & Support (Ch 3.8) rather than simple Task descriptions.
   e) Make a **dataflow diagram** for part of the system (choose an appropriate level). Discuss the differences between the dataflow diagram and the task description.
   f) Which **functional requirement styles** would you suggest using in this case, for each requirements level? Consider all of the styles mentioned in Chapter 3 (+2, 4) and not only the styles you used above.

NOTE: *If you already have draft requirements, work with these. Consider how to specify using other techniques*

4. For your project, now that your understanding of the domain & product has increased, revisit your **context diagram** (ch 3.2) and consider updating it with
   a. additional actors and interacting systems
   b. high-level features flowing between your product and actors and other systems. (Mark this on the arrows.)

Att göra hemma

- Compare Dataflow diagrams and Task descriptions, for instance based on these factors [Lau övn 3.5]
  -Precision of input/output
  -Precision of function
  -Precision of user tasks
  -Customer understanding
  -Developer understanding
  -Problem description
  -Design independence
  -Verification

Se kap 3.6 och 3.14