Part 2

Exercise 1.2

Generic software development has to be aimed much more general and is often more bloated than custom software products. It has to be applicable to lots of different users compared to customized software products, which can be directed towards specific use cases.

Exercise 1.3

Maintainability, dependability, efficiency and acceptability.
Other important attributes could be: Scalability, trust (partially lies in dependability), comfortability (“feel” good to work with), effective to use (fast and easy to use).

Exercise 1.8

It could be relevant in critical software such as hospital systems, rocket software and such, but in general we do not think it should be a certified right.

Exercise 2.1

A system to control anti-lock braking in a car:
A waterfall model, as we know exactly what the system needs to do and no major changes will occur during the process. You might want to adjust when to engage the anti-lock and such, but that will be minor changes, which should (!) already be changeable in the system.

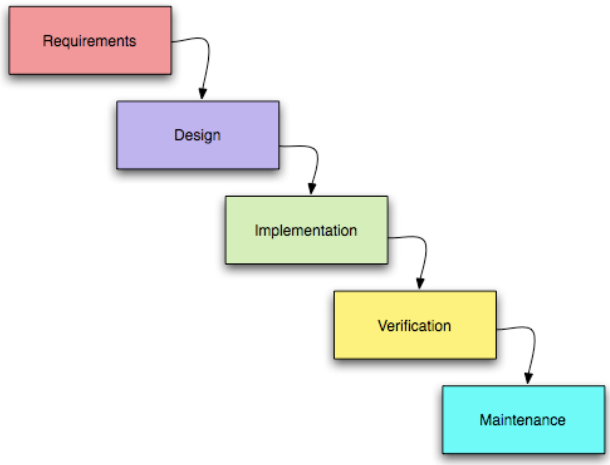
A virtual reality system to support software maintenance:
An iterative software process model, as we need to develop the tests incrementally as the software being maintained is developed. Also as the requirements are revised throughout the development process the virtual reality system has to change accordingly.

A university accounting system that replaces an existing system:
A spiral model, as we already should have a good idea about what the existing system has of functionality and how we want to improve it with a new system. Several prototype runs would help in that fashion and should result in a better product.

An interactive travel planning system that helps users plan journeys with the lowest environmental impact:
An agile software process model, as it uses the iterative process, but is more focused on the user and the user’s needs. That would be ideal for a system like this developed for the purpose of the user’s need to make a travel plan and at the same time be open for changes.

Exercise 2.5

The main activities in a software design processes are (order may vary):
- problem specification,
- solution design,
- implementation (development),
- validation (testing),
- documentation,
- deployment, and
- maintenance (evolution).



The diagram shows an example of the waterfall model, which takes a very straight forward stab at our main activities listed above. Alternatives could be more spiralling.

Exercise 2.9
Each of the dynamic workflows has different values for each static phase and this gives a far more structured view of a very dynamic process. Each workflow has its focus and is naturally weighted for its relevancy for each phase.