**Code complete**

*Chapter 1*

Software **BUILDING**not writing, coding or programming

Consists:

* **Building and debugging**
* Software design
* Creating a plan of building
* Testing
* Integration

Software building is the main chapter of developing, and the main part of a project. Sales, marketing and etc are important, but the software itself is more significant. Project can’t live without a product.

Developer and coder are different specializations. Dev must be able to do more. Dev is more creative. Dev often works not only with code, but with people and very often he must design soft, not only code it.

Devs are the core of the team.

*Chapter 2*

**Metaphor** is a very powerful method of understanding and explaining. Metaphor helps you to think more effectively and to understand complicated systems. The history of science is full of discoveries that were made through metaphors.

Metaphor should be sharp, related to real material world, and easy to understand.

Metaphor does not give you an answer. It just explains how to find the answer and how to ask right questions.

We can mix and combine metaphors to look on the subject from different sides and to be more creative and acceptable to innovations.

Metaphor can be dangerous when you are too related to it. It can lead you to incorrect conclusions.

The key metaphors in programming:

* Literary metaphor of coding
* Metaphor of a pearl (slow accretion of soft). Incremental approach
* Building metaphor (like building a house)

*Chapter 3*

Importance of planning and developing previous conditions

1. Define the type of software you want to build
2. Planning and defining of preparatory requirements is very important
3. Correcting of mistakes and bugs costs **mush less** at the beginning of the work
4. Clients are stupid creatures. It’s extremely important to clearly define requirements at the beginning because they always want to change them and bring some new unreal details to project. Tell them that it will affect price and terms – always works!
5. Preparation == risk-off
6. Attention to the quality == CONST during all work from start to end
7. **Define the problem** of customer (what he or she really wants??) clearly
8. **Explain** importance of preparatory requirements to colleagues and customer
9. Always think about architecture / structure of the project
10. Preparation usually takes 10-15% of working on project time