Coding Philosophy

Coding requires mental effort and presents constant new challenges that reward a creative mind. Integrating this creativity across teams requires some structure and process but ultimately the test of performance is how fit the product is for use by it's market, and how easily that market can adopt it to get a benefit.

The design process often involves collaboration with customers and adaption of plans to what usage patterns and expectations develop within the marketplace. This can often lead to renegotiation of earlier agreements and care must be taken to avoid contractual commitments to plans which may make this adaption difficult. There is however a need to have a plan both to allow collaborative effort on a common goal, and so that investors in a project know what is proposed and can buy into that strategy.

The development of code is not just about the raw functionality, performance and reliability. It is also about the quality perceptions of a product: that it can be relied upon to produce an expected outcome, the aesthetic elements of design that make the product desirable and that it is a subject of pride for those who created it.

Prior work on the Rational Unified Process and the later Agile methodologies leads to a small number of principles driving an interactive design process. Rapid iteration receives customer reaction sooner and shortens the at risk effort of new features. New features which might not be accepted by customers, may change business workflows and negate the benefit of other planned features, or may make the need for alternative features apparent. So iterate early.

Design

Represent software visibly where possible so as to improve understanding and communication. Manage the requirements process to avoid scope creep, force prioritization and set expectations. Iterations should be challenging and yet achievable at a stretch. Close cooperation between developers, designers, analysts and business customers involving frequent daily communication is to be preferred.

Development

Develop repeated iterations which expose the riskiest aspects of a problem as soon as possible. Team structures need to be fluid to allow regular changes of roles through the project lifecycle as well as adaption to other changing circumstances. Iterative development should result in frequent delivery of usable software components to beta testers and availability to customers every few weeks. More strategic changes should be structured to minimize interruptions of the delivery cycle.

Quality

Keeping designs simple by avoiding unnecessary complexity and effort is essential. Continuous attention needs to be given to technical excellence and good design by incorporating regular reviews of design, code, user feedback and verifications of quality. Established systems with dependent businesses need particularly careful control of changes to minimize unexpected costs as well as to understand and mitigate reputational risk.