File 20110715.1100: Notes from Grady Booch talk on transforming legacy systems. He happened to be in Denver, and mentioned at the start of his lecture being caught in the same rain and flash flooding conditions.

Michael Feathers said, 'legacy code is code without tests'. Jeff Jonas said, 'all systems are legacy systems'. Bjarne Stroustrup said, 'legacy code is different because it works and it scales'. Martin Fowler said, 'technical debt incurs interest payments'.

Tribal memory: why things are the way they are. '…loss of information from design to manifestation in code'.

ISO/IEC WD4 42010 Architecture Description standard.

[paraphrase] After you've extracted the economic value from a system, either blow it up or open source it. Can get advantages from your code influencing standards setting process in future.

The hardest way to handle legacy code is to transform it: in the small, refactor; in the large, architectural transformation. Requires a steady, long-term vision of what you want it to look like some day. This is what you do with the most important code, the code you depend on.

Refactoring by Martin Fowler is a delightful book.

References