Classical software verification focuses on answering the question if the implementation of a piece of software conforms to a specification. Verification plays an essential role in safety-critical domains like railway, automotive, aviation, and also medical devices. Another crucial aspect in those domains is the analysis what happens if a specification-conforming system is embedded into a dangerously behaving environment or if parts of the system (e.g., sensors or radio-devices) are malfunctioning. Even under such problematic circumstances, the operation of a safety-critical system should not lead to accidents or cause any other form of harm. Traditional safety techniques like the fault tree analysis describe a way how an upper bound of the hazard probability can be estimated using the probabilities of the component faults, but these traditional safety-analysis techniques have not been designed for software-intensive systems.

Because of their complex behavior, such software-intensive systems are hard to analyze. This thesis presents an approach how such systems can be modeled and analyzed probabilistically using executable modeling languages, i.e., modeling languages that model behavior in an executable way; as a consequence, the approach mitigates problems that arise in the probabilistic analysis of software-intensive systems.