Fault Injection

18-849b Dependable Embedded Systems

Robert Slater

March 9, 1999

Required Reading: Stott, D.T.; Ries, G. Mei-Chen Hsueh, Iyer, R. K. "Dependability Analysis of a High-Speed Network Using Software-implemented Fault Injection and Simulated Fault Injection" Proceedings of the 27th International Fault Tolerant Computing Symposium, p108-119

Best Tutorial: Mei-Chen Hsueh; Tsai, T. K.; Iyer, R. K. "Fault injection techniques and tools" Computer, April 1997. P75-82

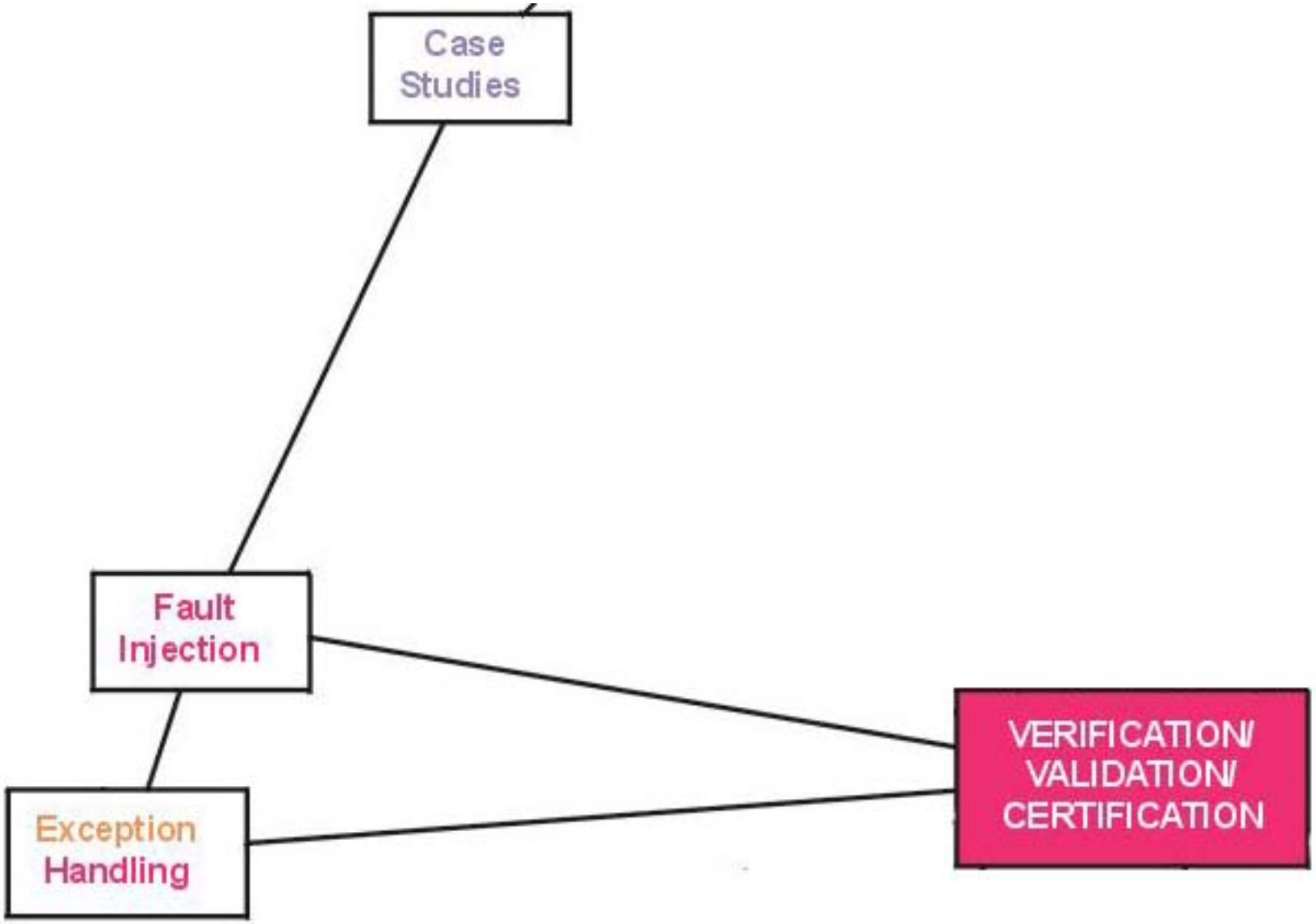
Authoritative Paper: Arlat, J.; Aguera, M.; Amat, L.; Crouzet, Y.; Fabre, J.-C.; Laprie, J.-C.; Martins, E.; Powell, D. "Fault injection for dependability validation: a methodology and some applications" IEEE Transactions on Software Engineering, Feb 1990. p 166-82.

Carnegie Mellon

Overview: Fault Injection

- **Introduction**
  - Connected
- **Key concepts**
  - Hardware Fault Injection
  - Software Fault Injection
  - What is Fault Injection good for?
- **Tools / techniques / metrics**
  - Lots of tools
  - Metrics under proposal
- **Relationship to other topics**
  - Reliability, fault tolerance, and others
- **Conclusions & future work**

YOU ARE HERE MAP



Description of Fault Injection

- **What is Fault Injection?**
  - It's the deliberate introduction of faults into a system, and the examination of the behavior of the system afterwards.
  - Used for both hardware and software to test fault-tolerant mechanisms.
- **What are the issues?**
  - Invasive vs. non-invasive
  - Simulation vs. execution
- **Problems**
  - Fault injection often used for development rather than testing
  - Often very application specific, rather than generalized
  - Needs some way to inject the faults

Hardware Fault Injection

- **Simulation**
  - Use SW model of hardware to cause faults in the interface and the internals of the system
  - Basically non-intrusive, but costs simulation time
- **Execution**
  - Manipulation of pins to cause faults (non-intrusive)
  - E-M fields and Radiation sources to cause random transient faults (non-intrusive)
  - Laser Fault Injection (LFI) to cause specific transient faults (non-intrusive)
- **Hardware fault injection is largely non-intrusive, because the testing mechanism is not run on the hardware.**

Software Fault Injection

- **Simulation**
  - Protocol simulation for distributed systems (non-invasive)
- **Execution**
  - Fault injection into distributed protocol (invasive, potentially non-invasive)
  - Fault injection into data sets (non-invasive)
  - Fault injection into hard real time systems (invasive, potentially non-invasive)
  - Program mutation (invasive)
- **Software fault injection tends to be invasive because it runs on the system it is testing, and can affect timing and system state.**

What do we use fault-injection for?

- **Not for debugging.**
  - While there can be debugging benefits to fault injection, it's got a lot of overhead, and it isn't really designed to isolate faults in the program
- **Proposed use for validation**
  - Make sure the system can withstand/recover from faults of certain types for validation
- **Metric for robustness**
  - It's a way to directly measure how well a system responds to faults
  - The only problem is that faults are often application specific
  - Also, how do we bias/construct our metric

Tools / Techniques

- **Hardware**
  - MEPHISTO for VHDL fault injection
  - Messaline - pin level fault injection, FIST and MARS - E-Mag and radiation, LFI - laser fault injection
- **Software tools**
  - Ferrari - CPU, memory, bus faults through CPU traps
  - Ftape - CPU, memory, and disks through OS & driver modifications
  - DOCTOR - CPU, memory, and network faults through time-out, trap, anc code modification
  - Xception - multiple faults through hardware exception trigger
  - ORCHESTRA - distributed system protocol tester through Protocol Fault Injection (PFI) layer

Metrics

- **What metrics are there?**
  - Fault generated and type of response for specific test suite
  - Still trying to figure out computation of test suites and evaluation of results
- **What are they good for?**
  - So far they're proposed as benchmarks for robustness and fault-tolerance of systems.
  - Other quality metrics measure process, as opposed to product.

Relationship To Other Topic Areas

- **SW Fault-tolerance, Robustness, SW Testing**
  - Possible metric
- **Ultradependability**
  - can accelerate testing of a system when naturally-occurring faults are too sparse
- **Safety Critical Systems Analysis, Software Safety**
  - Test system's ability to prevent faults from becoming hazards
- **Case Studies**
  - Provides them
- **Verification Validation/Certification**
  - Proposed step towards safety/quality validation/certification
- **Exception Handling**
  - Faults are often excptions, method for testing exception handling capabilities.

Conclusions & Future Work

- **Mechanisms for inserting faults into systems fairly well understood**
- **Proper use of fault injection is testing, not debugging**
- **Results are often difficult to interpret or make conclusions about**

- **Future work**
  - Focusing of test suites and automation of test suite creation
  - Investigation into relationship with robustness and fault-tolerance
  - Generalization of results
  - Potential uses as a metric and a validation technique

Dependability Analysis of a High-Speed...

- **Simulation and execution performed on the same system**
- **These bullets summarize the major points/approach**
  - Test through simulation, then through execution, and correlate the results.
- **These bullets summarize the results/contribution**
  - If you corrupt micro-controller code, it doesn't work. (Wow!)
  - But wait … deficiencies in the simulation approach were revealed
  - Potential results of targeting high-usage areas
  - Importance of the specifications