Robustness Assurance for Systems at Scale

John Goodenough

November 2011

© 2011 Carnegie Mellon University

Limitations of Component Certification

Certifying components is insufficient to ensure against catastrophic system failure because:

- The component specification omits some behavior that is critical to the context in which the component is used
- Unidentified assumptions on which the component's certification is based do not hold under some circumstances of actual system operation
- The specification is interpreted differently by the developer, the certifier, and/or by those deploying the component in a larger system context
- Despite certification, the component will occasionally fail to satisfy its specification
  - Software bug
  - Hardware failure

© 2011 Carnegie Mellon University

Going Beyond Component Certification

System certification is the goal

- Must consider the effects of possible component misbehavior

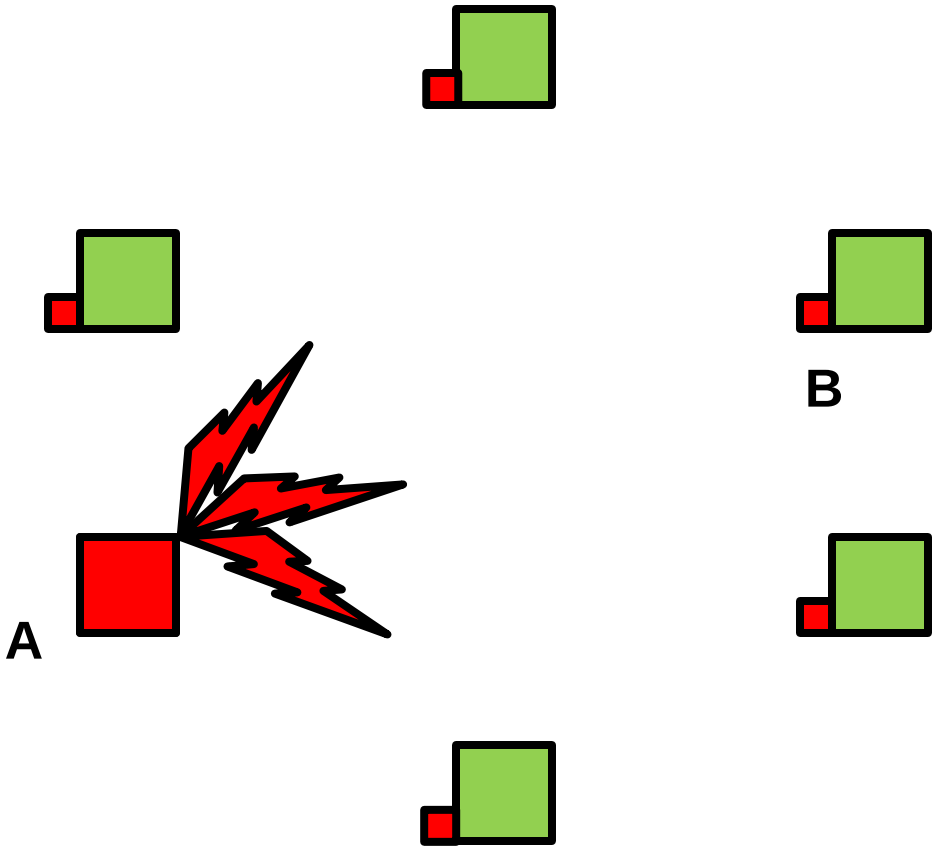
The context of a component's use determines

- Whether its behavior is appropriate
- The impact of its behavior on the rest of the system

What can be done to better assure ourselves (and others) that systems are robust against "unlikely" component misbehaviors?

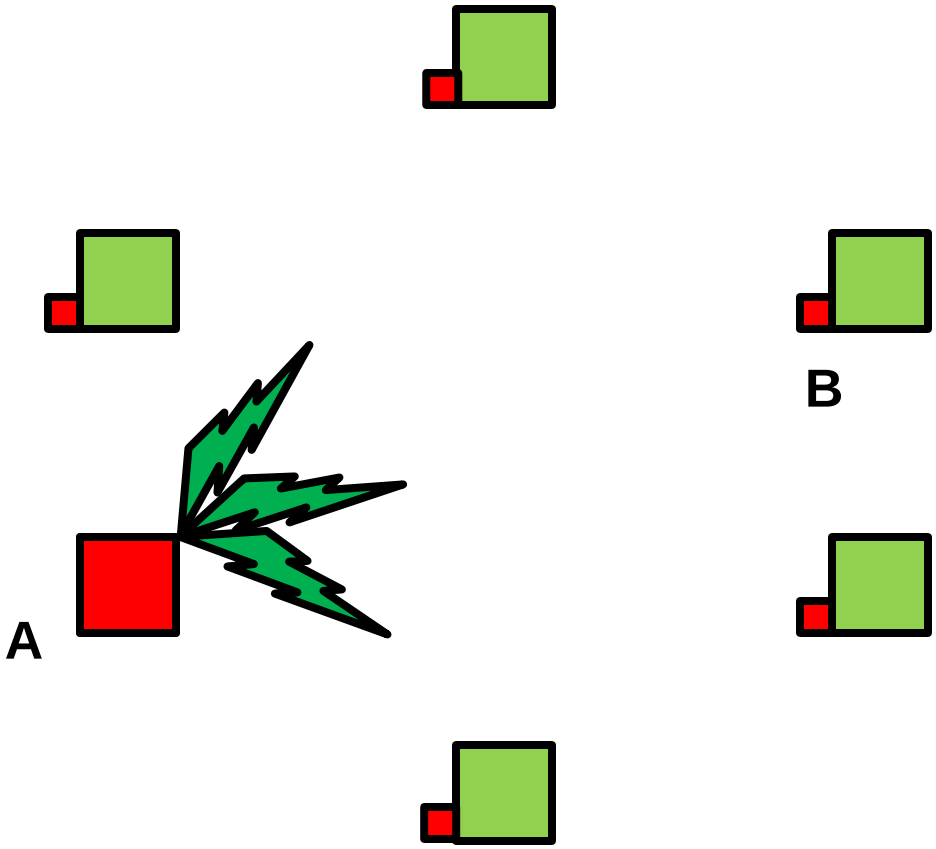
© 2011 Carnegie Mellon University

A Large-Scale System Failure Context



© 2011 Carnegie Mellon University

A Large-Scale System Failure Context



© 2011 Carnegie Mellon University

The incident

Node A had a hardware failure and notified node B that it was out of service

Node A recovered and signaled node B that it was working by sending a call request to B

While updating its status tables, node B received two additional requests from node A, which derailed B's updating process, corrupting some data

Node B went down and signaled other nodes that it was not working. Its backup system failed too (same software)

Node B came back up very quickly and signaled other nodes it was up by sending call requests, which arrived at just the right time to cause some other nodes to go down while updating their status tables

The amount of time spent updating status tables began to increase, making it more likely that call requests would be received when the node software was vulnerable

Eventually essentially all processing time was spent in updating the system state because nodes were failing and recovering so rapidly

© 2011 Carnegie Mellon University

Analysis

The nodes were behaving according to spec wrt sending out messages about their status, updating the changing status of other nodes, and auto-recovering from self-detected failures

The bug (a race condition) was revealed by external stimulus patterns (the arrival of call requests in quick succession *while* updating status)

- This is a bug that is revealed only by how the component is being used by the rest of the system
- It is a bug that depends on the timing of external requests
- The external event pattern was very unlikely, but there were large numbers of events

There was an assumption that the interval between auto-recovery and the next failure would not be rapid

© 2011 Carnegie Mellon University

Robustness Assurance

Hazard identification

- Interconnected, replicated software increases vulnerability to cascading failures
- Auto-recovery actions in such systems can be a source of cascading failure
- Auto-recovery is designed for small scales; at larger scale, such actions can prevent recovery by increasing “non-productive” network load

Hazard mitigation is architectural, e.g.,

- Throttle dissemination of status messages or rate of auto-recovery actions
- Design to be less vulnerable to inconsistent state

System design makes a system robust against unlikely but potentially catastrophic behaviors

© 2011 Carnegie Mellon University

Summary: Robustness Assurance

Components cannot be guaranteed against unexpected failure, i.e., behavior that does not satisfy their specification

What unexpected behaviors are conceivable? (These are hazards)

- Examine “typical” mistaken assumptions for large-scale systems
- Look at behavior allowed by the design if components misbehave
- Unlikely events are not necessarily rare

What collective misbehavior effects are possible?

- What mitigations are in place to correct or mitigate these effects
- Consider monitoring and on-the-fly correction

If we aren’t aware of these potentials for catastrophic failure, how can we justifiably conclude that system behavior is adequately constrained?

© 2011 Carnegie Mellon University

Contact

John B. Goodenough

Fellow

System of Systems Software Assurance

Telephone: 412-268-6391

Email: jbg@sei.cmu.edu

U.S. Mail:

Software Engineering Institute

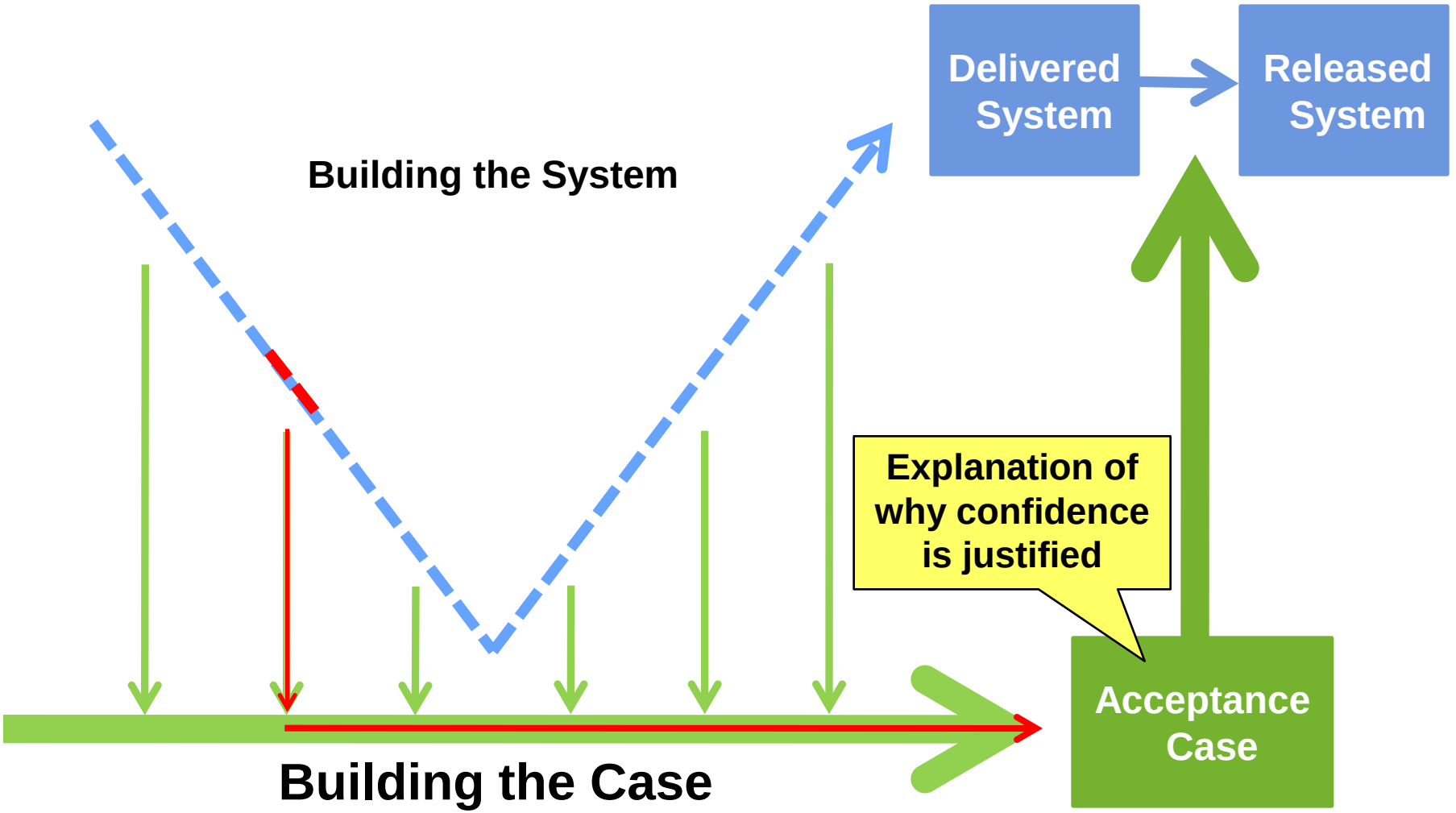
Carnegie Mellon University

4500 Fifth Avenue

Pittsburgh, PA 15213-3890

© 2011 Carnegie Mellon University

Building Justified Confidence



© 2011 Carnegie Mellon University

NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this presentation is not intended in any way to infringe on the rights of the trademark holder.

This Presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.