A Taxonomy of Live Updates

Cristiano Giuffrida Andrew S. Tanenbaum

Vrije Universiteit Amsterdam

16th Annual Conference of the
Advanced School for Computing and Imaging

Veldhoven, The Netherlands
November 1-3, 2010

Live update


John Doe O.S.
WARNING!
You are running an old version of you.
Please update as soon as possible.
Update Yourself

Update mechanisms

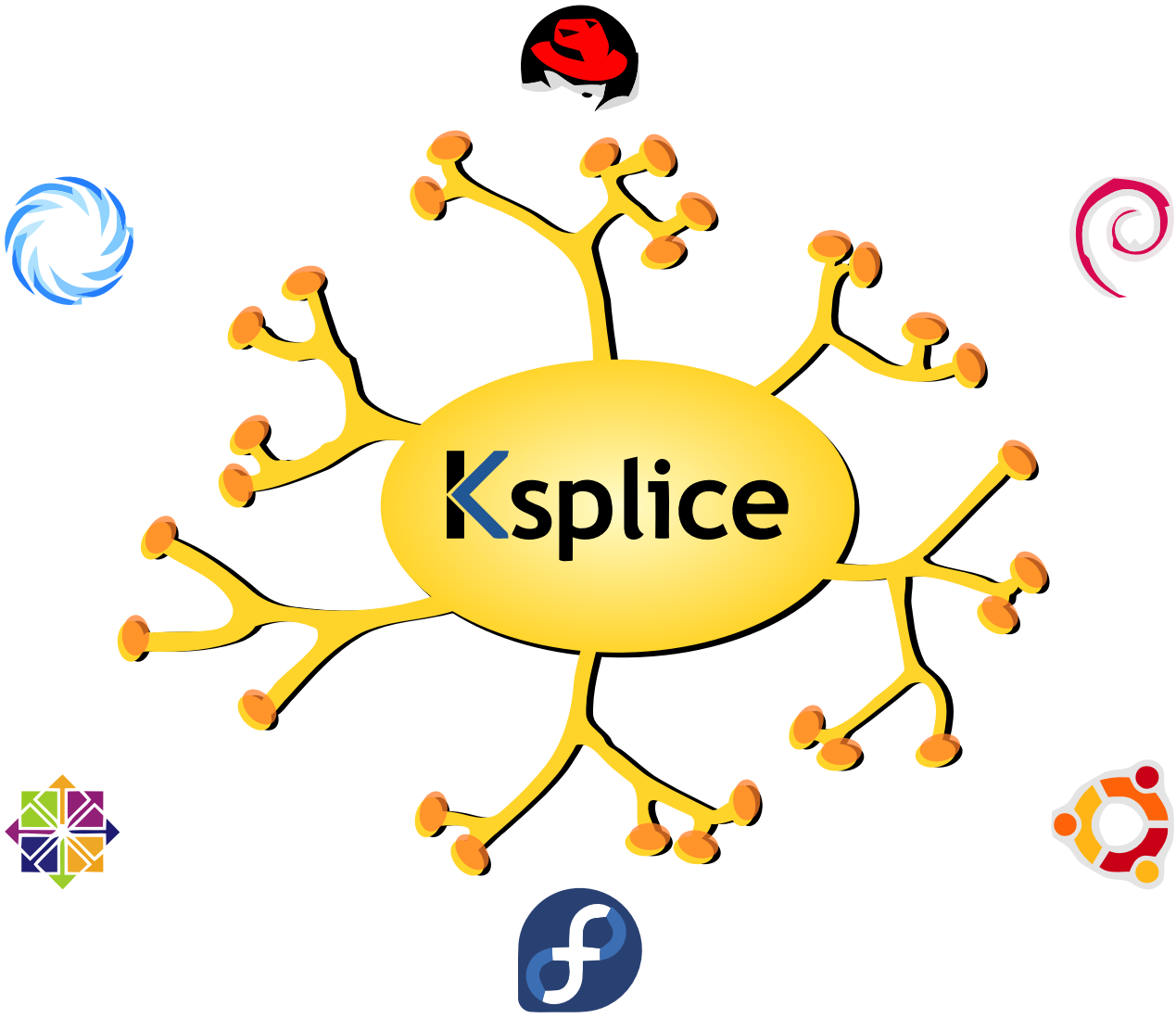
What

- Hardware-based and software-based solutions
- Unit of change: function, object, component, system
- Type of software: generic applications, operating systems

How

1. Load the update: dynamic linking or specialized support
2. Transfer the state: author-provided state transfer function
3. Redirect execution to the new version: indirection mechanisms

Existing solutions



Update safety


Safe to update when functions inactive


Safe to update when objects inactive


Safe to update when contextual effects consistent


Safe to update when no transaction in progress

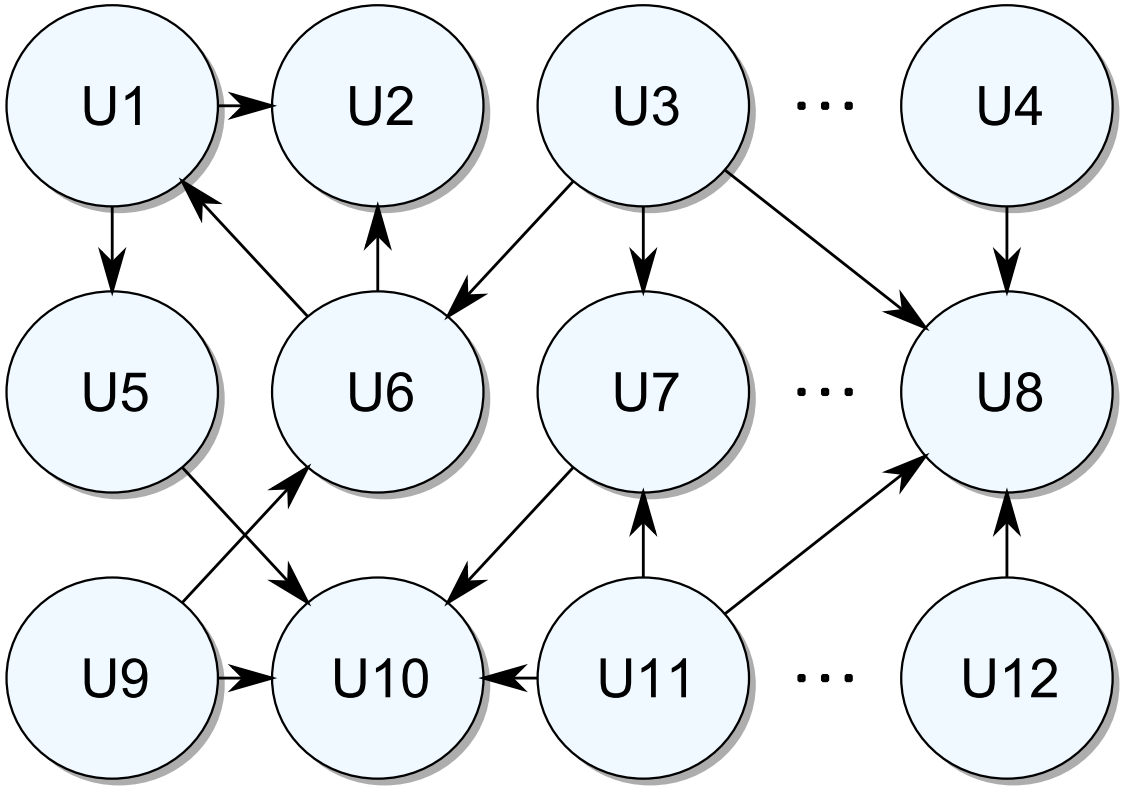

Safe to update when no event in progress

The problem

A taxonomy of live updates

- Goal: understand properties and limitations of live update
- We propose a taxonomy through real-life update scenarios
- Update scenarios with an increasing level of severity
- Evaluate update complexity and effects on the system

Reference model used



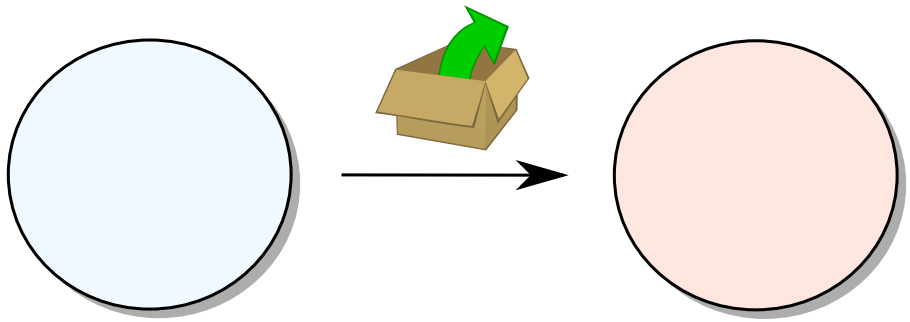
The taxonomy

1. Update affects one structural unit
2. Update affects protocol
3. Update affects global data
4. Update affects global algorithm
5. Update affects data on the disk
6. Update affects hardware requirements

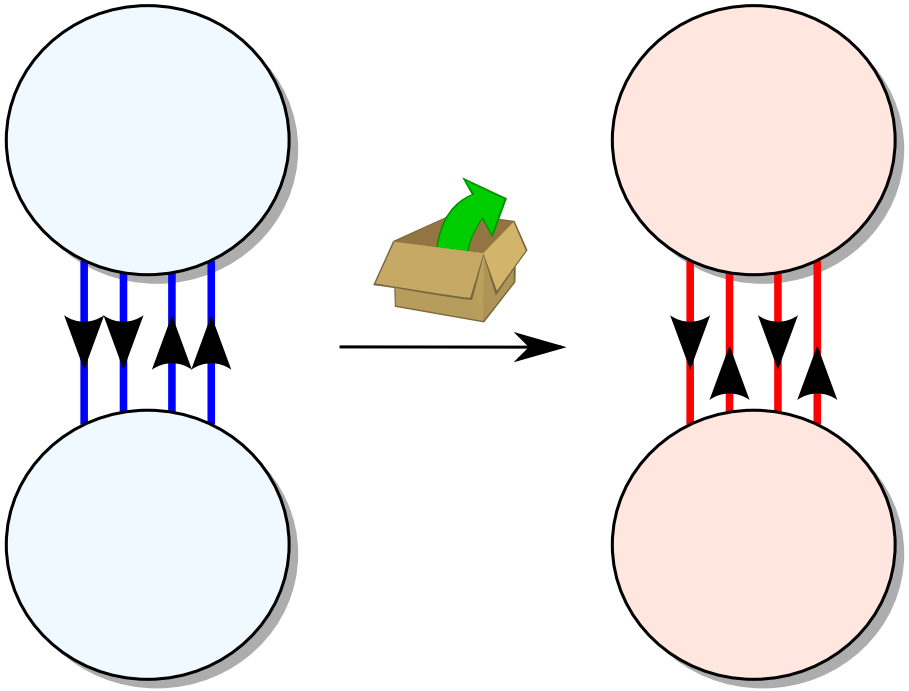
The taxonomy

1. Update affects one structural unit
2. Update affects protocol
3. Update affects global data
4. Update affects global algorithm
5. Update affects data on the disk
6. Update affects hardware requirements

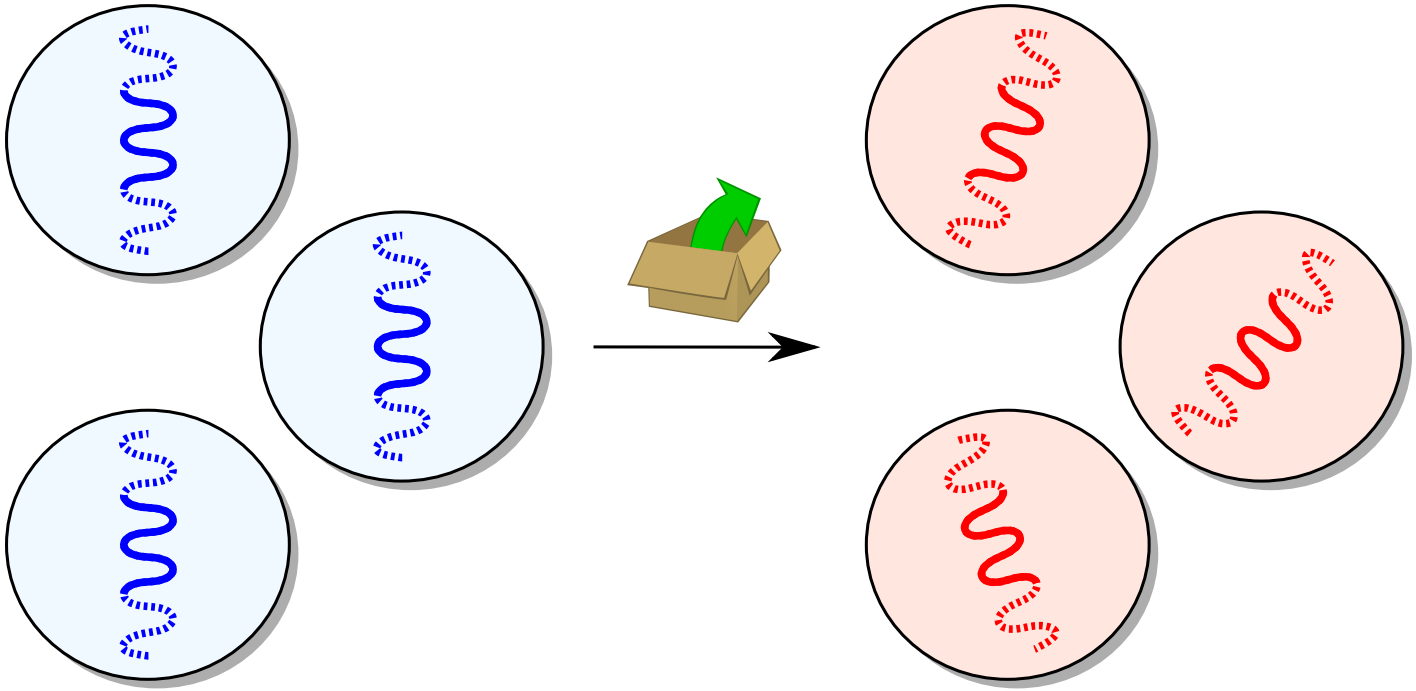
Update affects one structural unit



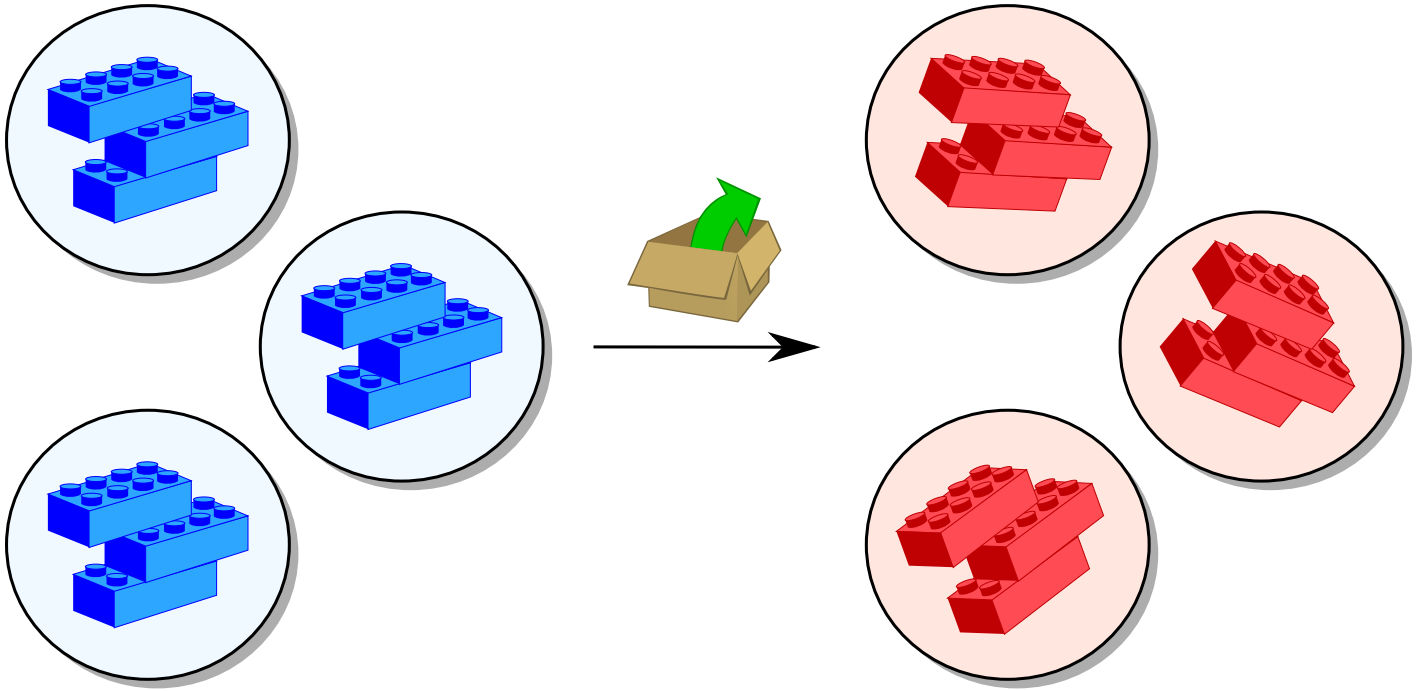
Update affects protocol



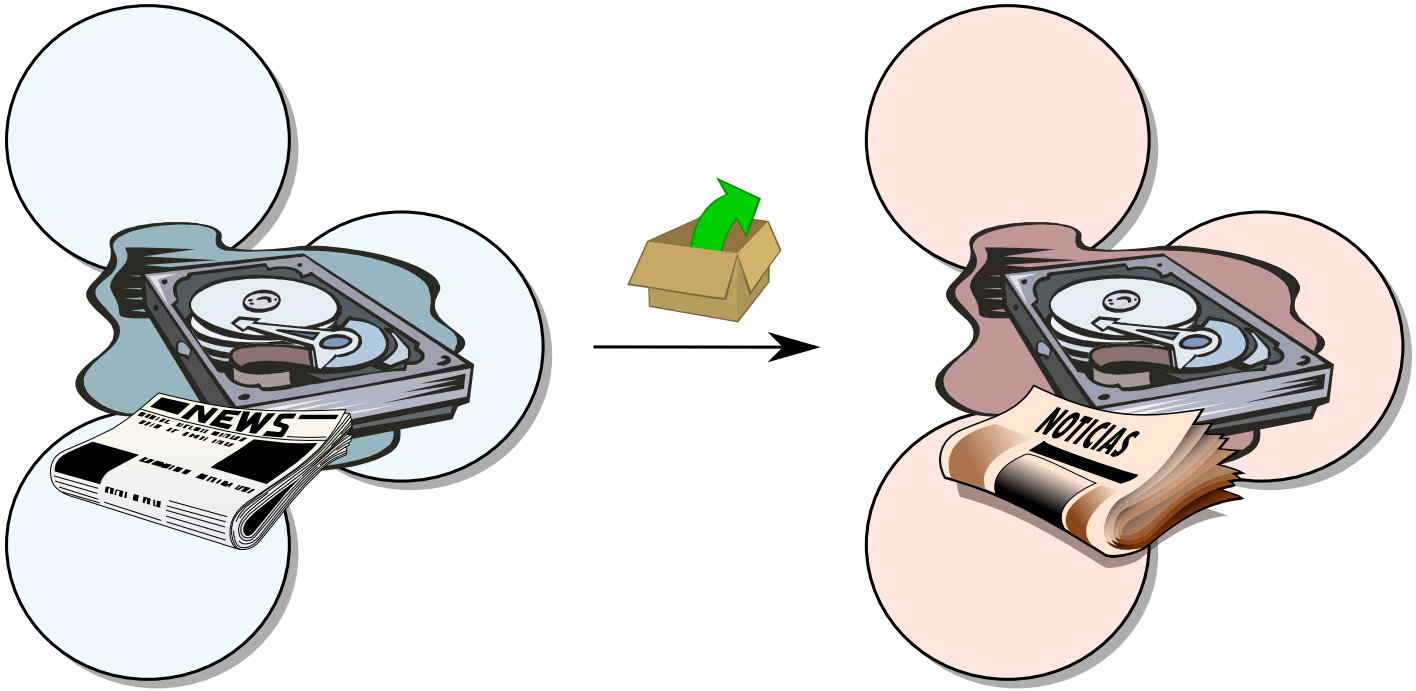
Update affects global data



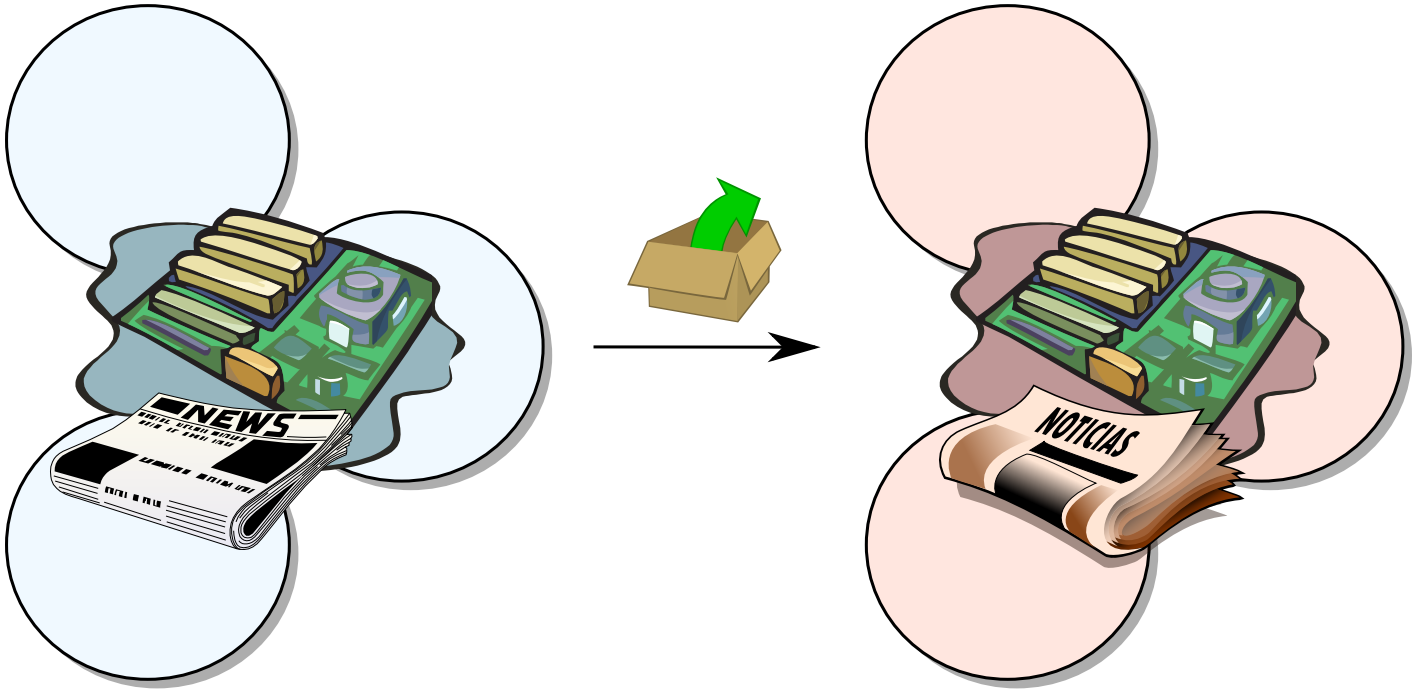
Update affects global algorithm



Update affects data on the disk



Update affects hardware requirements

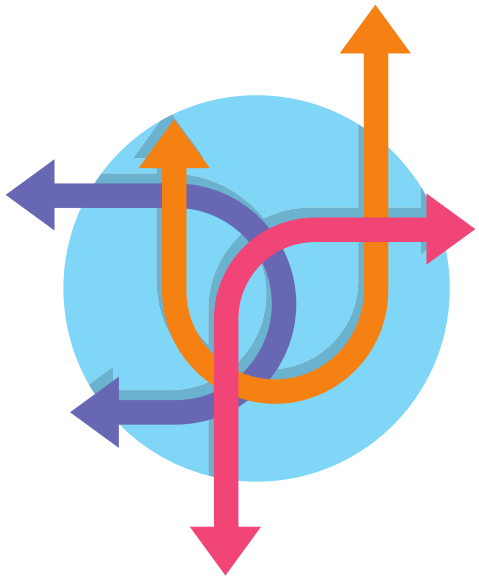


Lessons learned

Lessons learned

Live update is not always feasible

Lessons learned



Live update is not necessarily desirable

Lessons learned

Update constraints required may vary

Lessons learned

The nature of the update is central

Towards Update-aware Systems

- An update-centric model necessary to support many updates
- Update-aware systems to support live update by design
- Achieve better dependability properties
- Ability to support more and larger updates efficiently

Our approach

Software developers document changes

Our approach

The system cooperates during the update

Can this be done in practice?

- We have designed an update-aware operating system
- Multiserver microkernel-based OS architecture
- OS components can converge to a given state upon request
- The prototype system runs on top of the *Minix 3* microkernel
- The implementation of the system is underway

Summary

- We have proposed a taxonomy of live updates
- Evaluated update complexity and effects on the system
- Examined the role of the nature of the update
- Justified the need for update-aware systems
- A prototype system is part of ongoing work

A Taxonomy of Live Updates

Thank you!
Any questions?

Cristiano Giuffrida, Andrew S. Tanenbaum
{giuffrida,ast}@cs.vu.nl

Vrije Universiteit Amsterdam