Autonomicity vs. Complexity

Stefan Schmid

NEC Europe, Network Laboratories, Germany

stefan.schmid@netlab.nec.de

WAC 2005, Athens, Greece, 3-5 October 2005

Two Sides of the Coin

One side of the coin:

- Complexity calls for autonomicity
  - Systems that are very complex require autonomic support (esp. dynamic systems)
  - How else can they be managed in an economic manner?

The other side of the coin:

- Achieving full autonomicity in large systems is very complex (e.g., Internet)
  - Where to start?
  - How to 'divide' a large, comples system in order to 'conquer' it?
  - What if we 'divide' the problem in the wrong way?

A way out …

- However, complexity also depends on the approach
- My proposal:
  - Let's start bottom up
  - Build simple autonomic components (that solve some aspects of the overall problem space)
  - Put them together
- But, what if 1 + 1 ≠ 2
  - What happens if you combine 2 autonomic components – is the result a autonomic component?
    - To what extend?
    - How optimal is the composite?
    - What functionality is missing?
  - What is the likelihood that 2 autonomic components after they are put together interfere with each other?

⇨ **Iterative / evolutionary approach necessary!**

Further thoughts on discussed issues …

- Standards are still required – for interoperability of autonomic systems
  - at different levels though – depending what is made autonomic
  - and hopefully not that many

- Autonomicity introduces complexity (and hence CAPEX – but only initially for the 1st time development), but reduces management cost in the long-run (and hence OPEX)!

- Autonomicity is a principle that can be built in all systems/functions – not just a new middleware that is applied at one point in the network