Configuration Management Summit

Cfengine
some facts

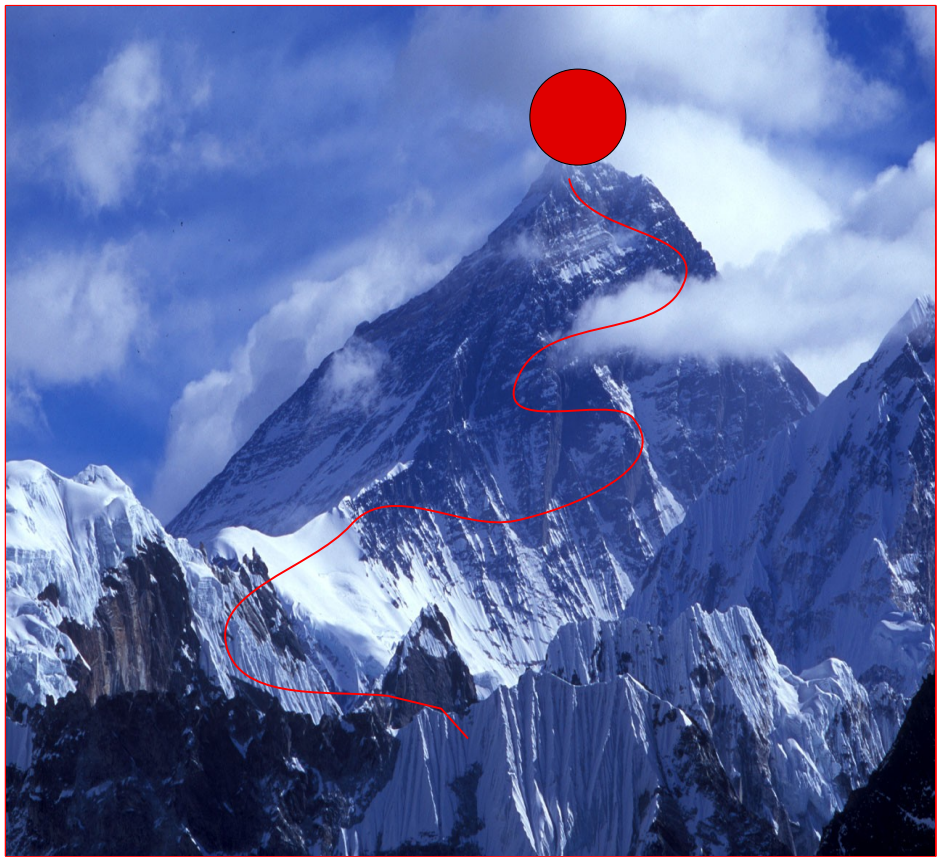
Mark Burgess

Cfengine AS
Oslo University College

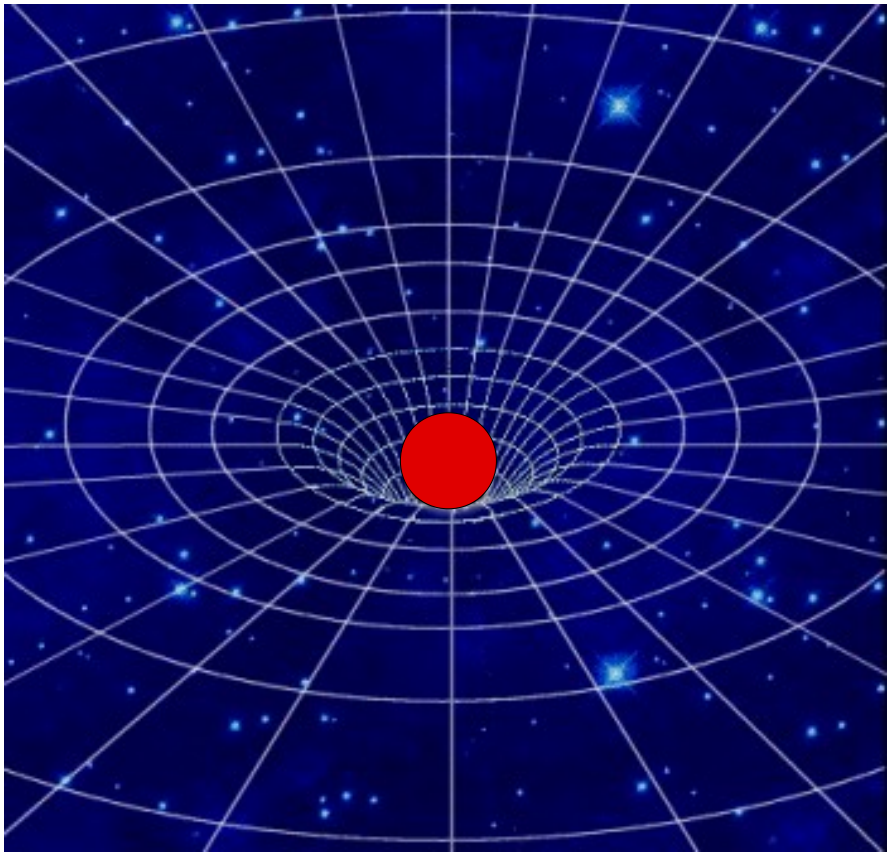
What is cfengine?

- An agent-based change management system, with “convergent” or “self-healing” semantics
- A largely declarative language for describing desired (or “promised”) states
- A self-learning monitoring framework
- A knowledge management framework
- Cfengine is written in C, with few dependencies

Convergence = self-healing



Baseline and recipe



Convergence to end state

History

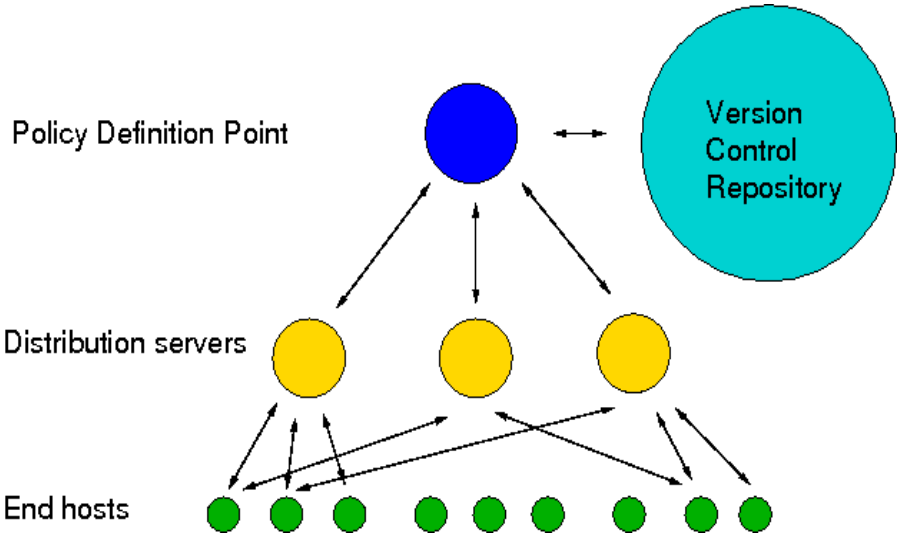
- **1993** Introduced in at Oslo University as a cross platform interface that would document desired state, rather than change algorithms
- **1998** Re-interpreted in as a `computer immune system' (self-healing). Laid out a research programme for system administration.
- **1999-2002** Formalized concept of “convergence” and limits for system correctness (more than idempotence).
- **2003-2007** Developed Promise Theory to fully understand the issues of distributed compliance
- **2008** Rewrote Cfengine completely as a “promise engine”, preserving core technology and principles.

Uncompromised Principles

- Autonomy of control – not allowed to send Cfengine instructions from outside. Local system always has last word.
- Avoid unnecessary network use
- Pull not push (voluntary cooperation not attack)
- Use network opportunistically, not required
- Run many times – “system never gets worse” (convergence + idempotence)

Architecture

- Autonomous agents can work independently or cooperate by information-sharing on a peer-to-peer basis. Decentralized.
- Commonly used to implement “single point of control” while avoiding “single point of failure”.



Comment: scalability

- Scalability is limited by network bottlenecks in
  - Inter-host communication
  - Dependencies and other relationships
  - Comprehenisbility of knowledge
- Research shows that hierarchies and inheritance are a fragile model for scaling so Cfengine avoids hierarchy, basing all its models on sets and freely associating networks.
- This is one reason for rejecting Object Oriented representations of configuration

Cfengine 3 - redesigned

- Promise Theory satisfies the principles and models constellations of autonomous agents
- PT explains the syntax and grammar needed for distributed configuration – gives a simple regular model and best-effort semantics
- Model: **configuration = promises + patterns**
- Cfengine “promises” these patterns.
- Offers generic idioms that reduce the information required for system description
- e.g “ACL” idiom – list what you want and what you don't want, or search and filter.

Solution Areas

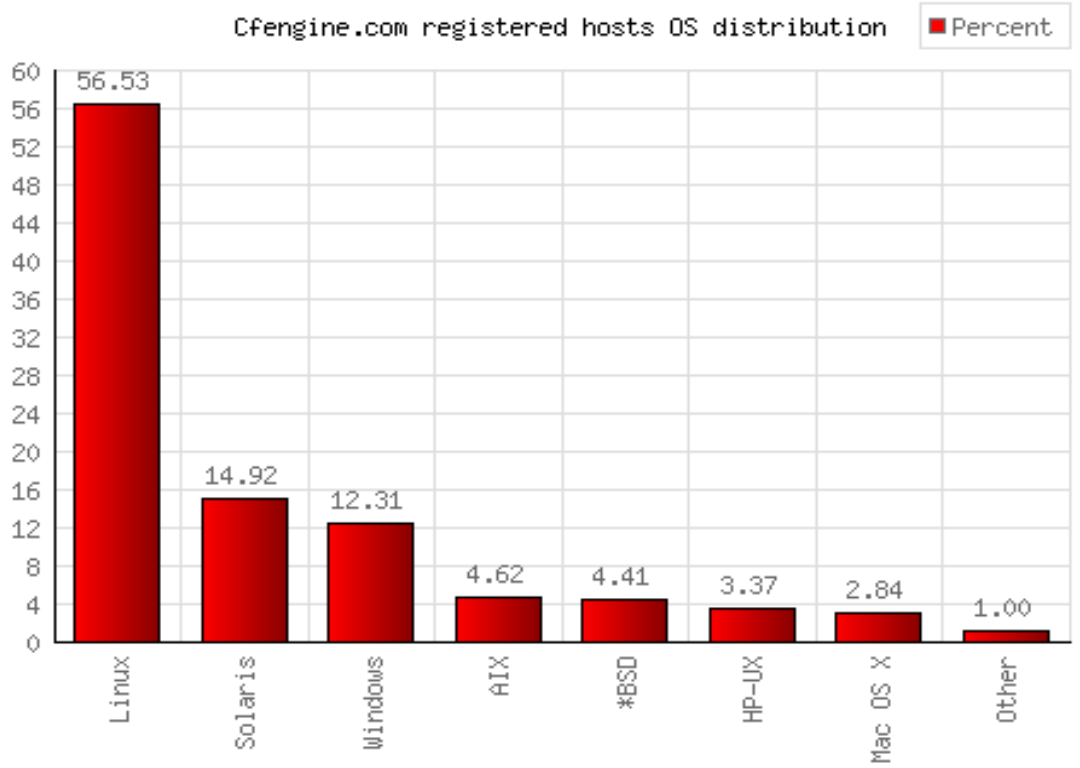
- Cover the full system lifecycle:
  - Build, Deploy, Manage, Audit
- Cfengine's capabilities currently include
  - Promising file attributes and contents, running processes, software packages, services, virtual entities, distributed scheduling, nfs devices.
  - Reporting of compliance, security monitoring.
- Future areas (no impediment in the model)
  - Routing, network management, advanced scheduling.
  - Mechanical promises: physical robotic and structural configurations

Knowledge Management

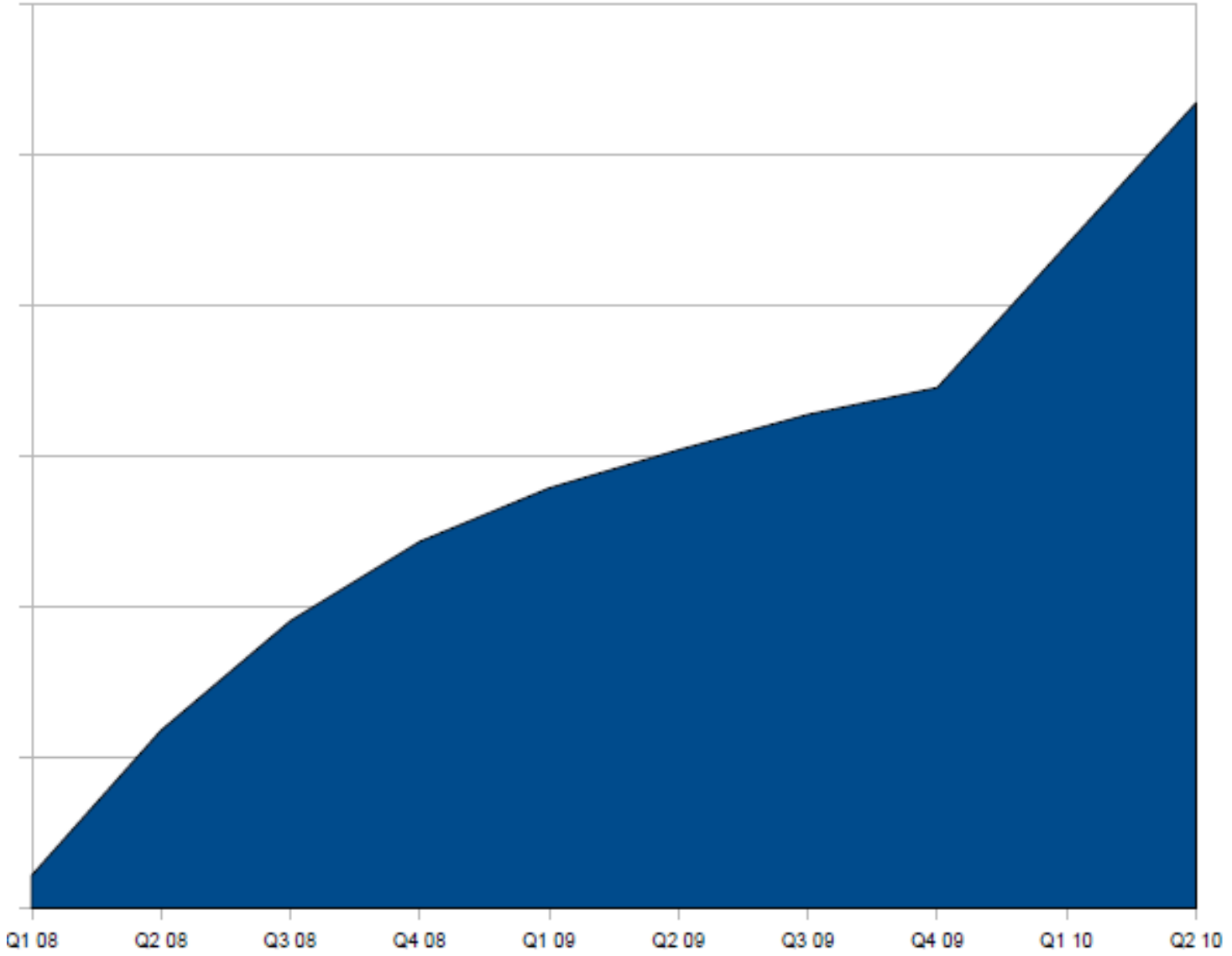
- The main unsolved problems lie in KM
- KM includes
  - Documentation (Write)
  - Assimilation / Comprehension (Read)
  - Generalizing (abstracting)
  - Modelling and X-associating (story-telling)
- Cfengine includes automated documentation and modelling based on Promise Theory. Generic mechanisms for abstracting through sets called “bundles”. Ties in to ISO13250 Topic Maps

Who is using Cfengine?

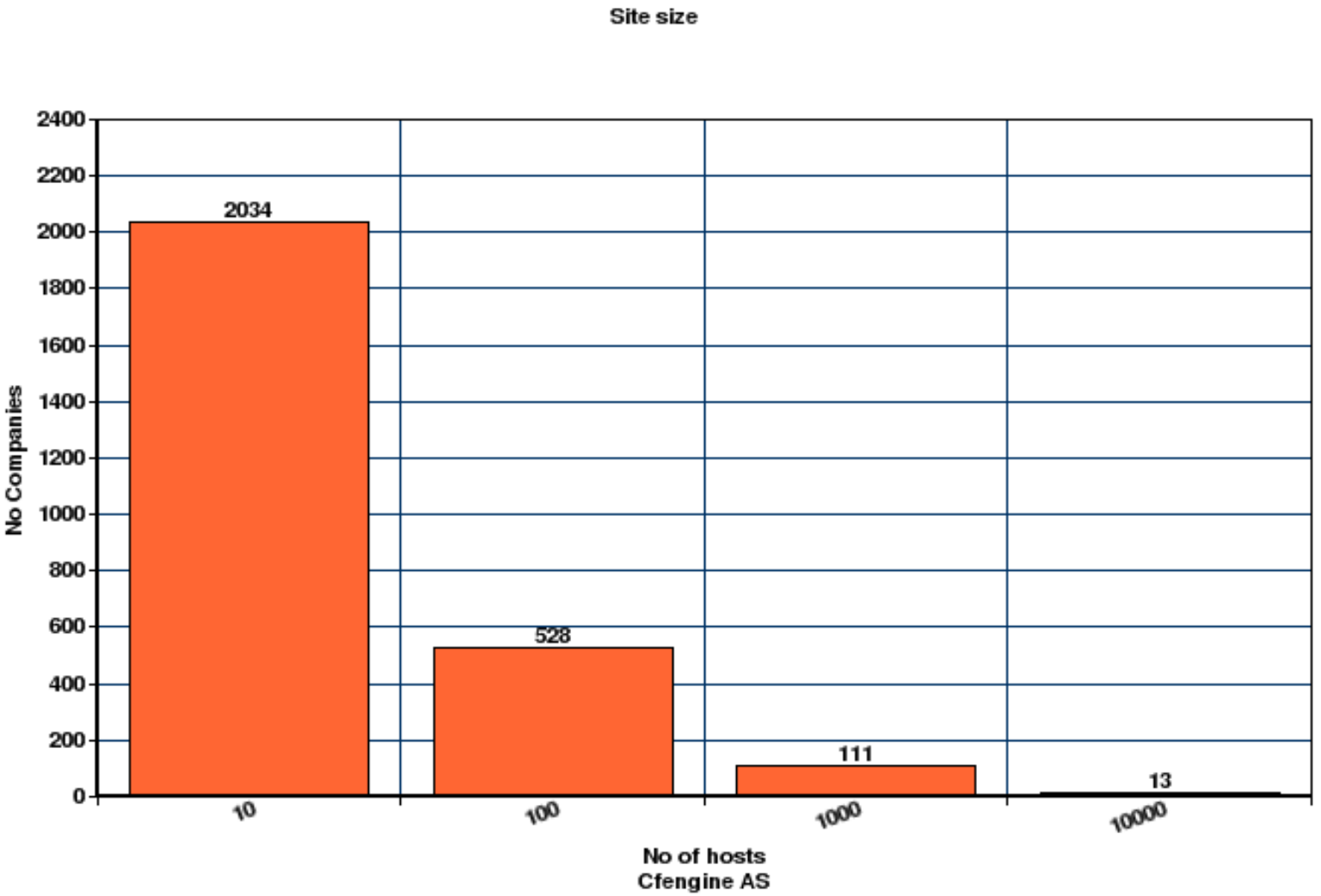
- More than 2000 registered organizations
- Estimate a million+ machines (mostly Cf2)
- All platforms:



Registered users 08-10



Machine park size



Comment: Assessment

- Many users only use cfengine for its advanced change capabilities during installation, not for self-healing or repair
- Many users use the framework but don't use the tools as intended, embedding shell commands because they don't see a better way
- Industry sophistication is only slowly catching up with the tools – old habits dye hard.
- Some “star users” who “get it” … AMD, eBay/PayPal, JP Morgen etc

Future Development

- We have research we could implement for the next 10 years.
- Unifying network management with server management
- Embedded devices
- etc

Conclusions

- How has the problem changed?
  - Cfengine originally addressed heterogeneity and consistency
  - Still problems for most organizations
  - We know how to do convergent self-healing now
  - Main problem is one of Knowledge Management
    - Tracking state
    - Understanding intentions
    - Aligning with business goals