Evergreen Software Development

—

Or, Why It's Better To Set It All On Fire And Start Over

Who's This Bloke, Then?

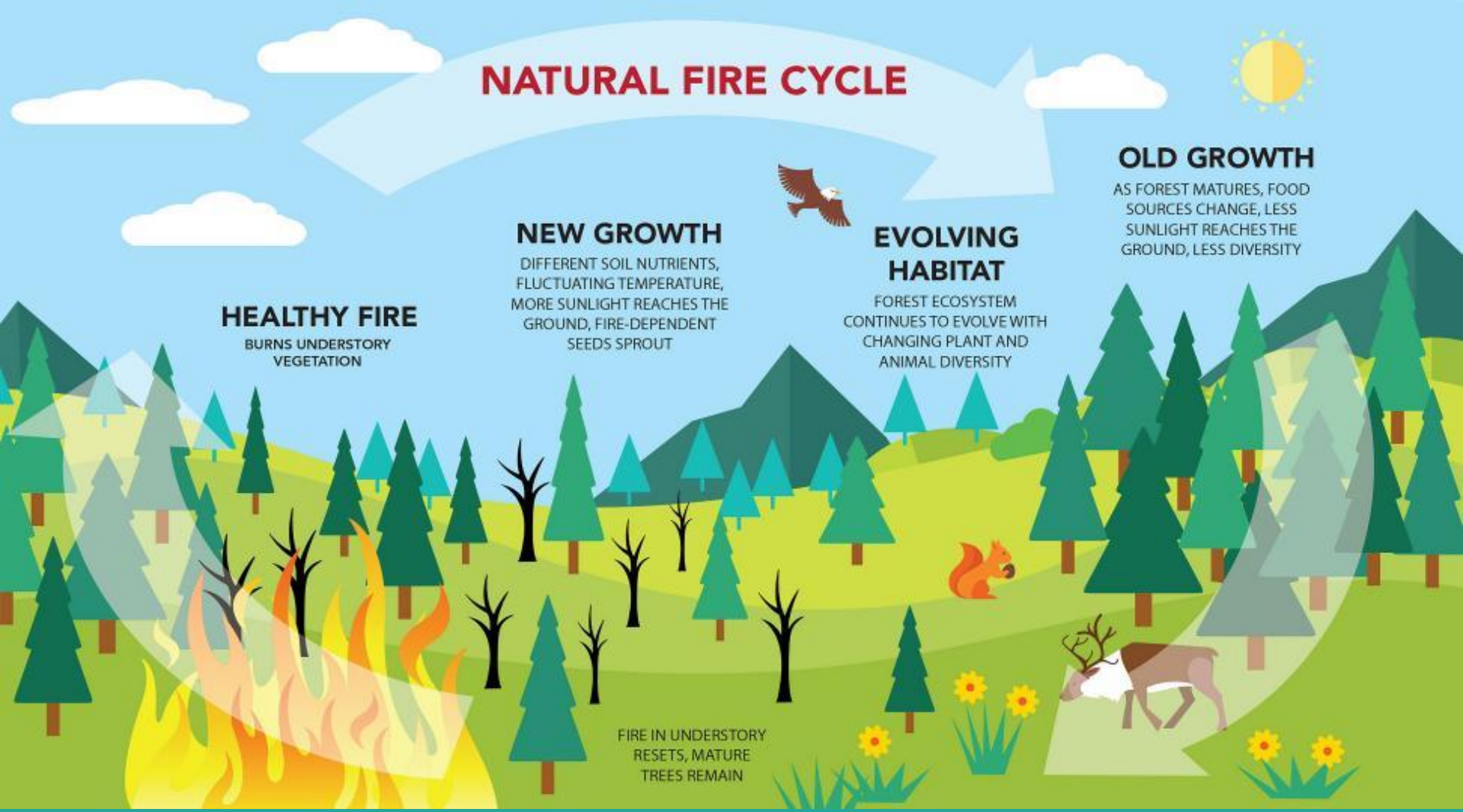
- Nathaniel Eliot, CEO of The Greenfield Guild
- Two decade veteran of the software industry
- A guy who collects and invents metaphors
- An oddly Anglophilic Texan

And What's This All About, Anyway?

- A philosophical no-code presentation
- Set all your architecture on fire
- Evergreen Development
- Recent trends in operational tools and attitudes
- Recent trends in security threat model
- Suggested implementation

Not Seeing the Forest-Fire for the Trees

- Traditional operations, like traditional forest management, focuses on fighting fires
- Growth-only focus in development produces a lot of dead wood that doesn't get cleared out
- When the breakout wildfire happens, even heroic efforts are unlikely to save things


NATURAL FIRE CYCLE
HEALTHY FIRE
BURNS UNDERSTORY VEGETATION
NEW GROWTH
DIFFERENT SOIL NUTRIENTS, FLUCTUATING TEMPERATURE, MORE SUNLIGHT REACHES THE GROUND, FIRE-DEPENDENT SEEDS SPROUT
EVOLVING HABITAT
FOREST ECOSYSTEM CONTINUES TO EVOLVE WITH CHANGING PLANT AND ANIMAL DIVERSITY
OLD GROWTH
AS FOREST MATURES, FOOD SOURCES CHANGE, LESS SUNLIGHT REACHES THE GROUND, LESS DIVERSITY
FIRE IN UNDERSTORY RESETS, MATURE TREES REMAIN

What is Evergreen Software?

- Automated upgrades and push-on-green to patch without manual intervention
- Regular recapitulation of all systems, including your infrastructure systems
- By patching and redeploying regularly, you segment the problem space for release errors


FENDER

Evolution in Operations

- Deployment tooling
  - Manual (i.e. “runbooks”)
  - Configuration management (e.g. Puppet, Chef)
  - Container orchestration (e.g. Mesos, Kubernetes)
- Organizational attitude
  - BOFH - antagonistic relationship with developers
  - DevOps - dev and ops should work together
  - SRE - best practices for working together at scale


Hello, IT, have you tried
turning it off and on again?

“Have you tried turning it off and on again?”

- Repeatable infrastructure
- Immutable infrastructure
- Crash-only software
- Chaos engineering

“If it hurts, do it more often”

- Upgrading dependencies hurts
- Rebuilding infrastructure hurts
- Security patching hurts

Evolution in Threat Model

- Attack surface is expanding
- Perimeter-defense model of security is dead
- Principle of least authority
- Bad actors are getting more sophisticated
- Vulnerabilities aren’t getting any younger
- Rapid defense in depth is now a necessity

How-To?




* Your Mileage May Vary

How-to?

- Make recapitulation more automatic, via container and infrastructure orchestration tools
- Regular push-on-green rebuilds of your core
- Automate the upgrades of dependencies
- Automatically open tickets on upgrade and rebuild failures, and fix them quickly

Questions?

Nathaniel P. Eliot
temujin9@greenfieldguild.com
@temujin9

—

https://greenfieldguild.com

References

- http://bofh.bjash.com/
- https://www.usenix.org/conference/lisa14/conference-program/presentation/klein
- https://landing.google.com/sre/books/
- https://www.synopsys.com/content/dam/synopsys/sig-assets/reports/2018-ossra.pdf

Suggested Tooling

- https://kubernetes.io
- https://www.terraform.io
- https://about.gitlab.com
- https://github.com/renovatebot/