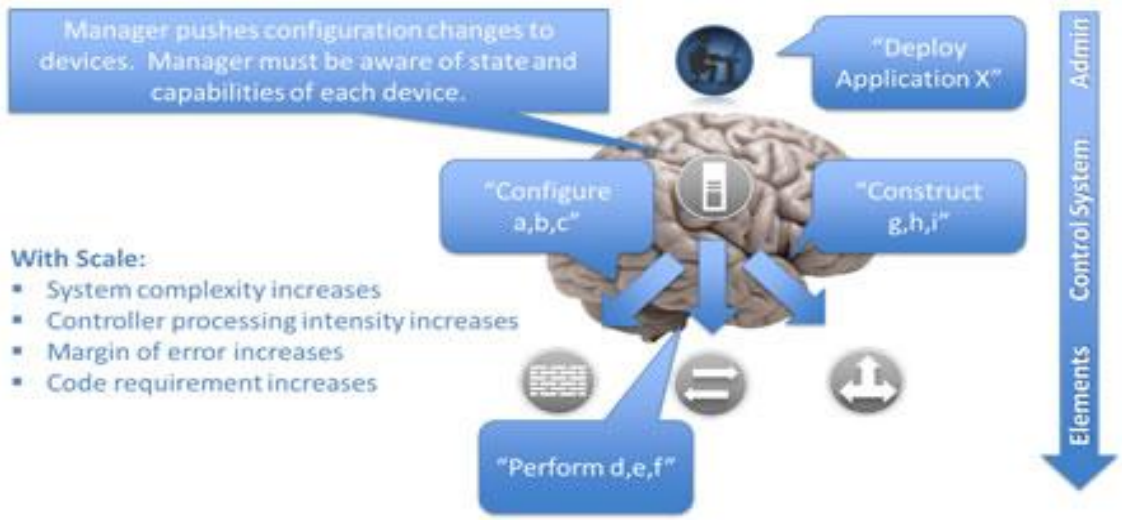
NETWORK MANAGEMENT NEEDS NEW IDEAS

Traditional network management systems haven’t adapted to the scale demanded by virtualization and cloud architectures. Distributed system designs may show the way.

As networks have grown, the industry has sought better ways in which to manage them at scale. Traditional network management systems are typically device-centric, particularly for network infrastructure. These systems take a top-down management approach and use a central server to push configuration into devices and to manage device state. With few exceptions, this approach provides no additional abstraction or functionally and fundamentally becomes a GUI representation of CLI configuration.

This top-down management model runs into problems when networks begin to scale. The problem is twofold: The management server (or a cluster of servers) must support additional elements that get added to the network, and also be able to handle the increasing complexity that comes from managing the state of numerous devices and other minute details.

We can conceptualize this approach as a "big brain" system, which is illustrated below. Unfortunately, the big brain doesn't scale well.



Big Brain System

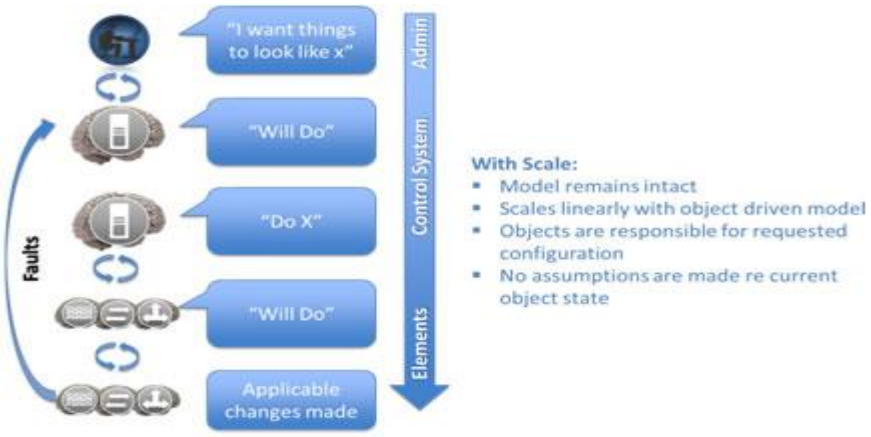
As shown in the diagram, the top-level manager must have knowledge of the state of each device and its components as well as the configuration options available for that device. As the overall system scales these systems expand in code complexity and CPU intensity. This model not only creates finite scalability limits but also inherent system fragility.

The fragility comes from the code requirements for precise management of numerous objects, as well as from the structure of the management itself. For example, a centralized management system assumes that the known state of the devices under management are the actual state of those devices. In the real world, however, changes occur and faults happen outside the control of central manager.

Inconsistency between actual state and intended state causes complications with normalizing the system. The linear processing of top-down instructions provides no ability to self-reconverge, or adopt dynamic changes.

For systems to scale past legacy enterprise environments into densely virtualized or cloud infrastructures, a new management paradigm is needed. We can take concepts from the design of distributed systems.

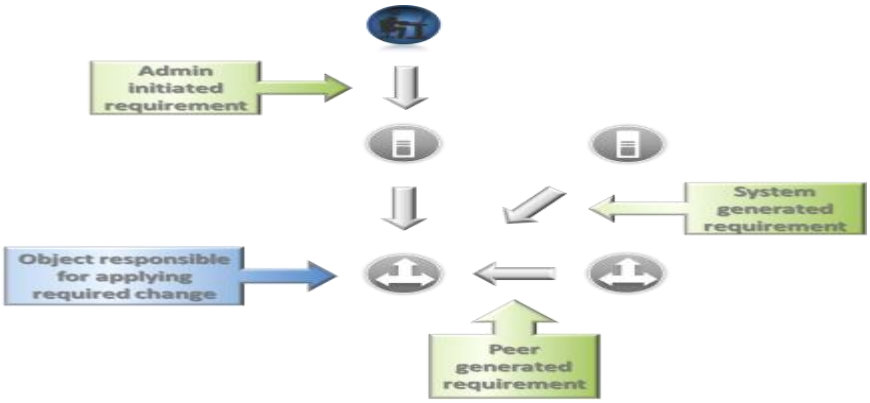
The first concept is the promise theory. At a high level, the promise theory provides a framework of autonomous agents that assist one another through voluntary cooperation. Rather than have a system of slave objects that rely on orders from a central management system, each object maintains responsibility for itself, and issues declarative state requirements to objects further down the hierarchy (which are in turn autonomous). The graphic below shows this relationship.



Promise Theory

Each object below the control system is fully autonomous and responsible for accepting change requests. Objects are additionally responsible for translating declarative state requirements into actual configuration changes and reporting faults or exceptions upward while maintaining implicit retries. This becomes a constant enforcement loop: observe > interpret > apply. In this model, the intelligence (the brain) is distributed throughout the system.

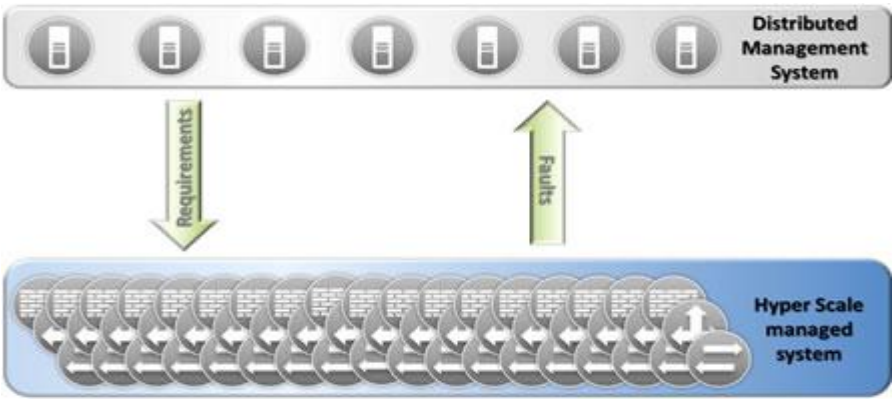
The promise theory model eliminates the serial nature of issuing and executing commands inherent in top-down models. This allows objects in the model to receive declarative state requirements from several other objects or control systems simultaneously and take responsibility for applying them. Declarative state requirements can also come from peers within the system, which can be thought of as requirements spreading like ripples through the system. This provides for better performance, faster convergence, implicit reconvergence, self-healing and distributed management. The diagram below shows this relationship.



Relationships in Promise Theory Model

The second concept that can be taken from distributed systems is the distribution of management. Rather than relying on scale-up, single controller models, or on scale-out models where state replication induces complexity and uncertainty, management can instead be distributed across multiple elements.

This distributed model of management provides far greater scale and resiliency than centralized management. As elements are added to the system, managers can be added as required. This model provides a linear scale between managed objects and management objects. This relationship is shown in the following diagram.



Linear Scale

Applying concepts from distributed systems and promise theory may provide a resilient, scalable system for highly virtualized or cloud environments far beyond what is capable with traditional top-down management systems.

Source: http://www.networkcomputing.com/networking/network-management-needs-new-ideas/a/d-id/1234258?