Exploring the Architecture of Ultra Large Scale systems

Len Bass
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pa 15213

Sponsored by the U.S. Department of Defense

Acknowledgements

Phil Boxer

Rick Kazman

Mark Klein

Michael Musheno

Linda Northrop

Kurt Wallnau

Outline

What is a ULS system?

Examples that exemplify some ULS characteristics

Quality attributes for ULS systems

Summary

Ultra Large Scale (ULS) Systems

Ultra Large Scale Systems (Ultra-Large-Scale Systems: The Software Challenge of the Future July 2006) have the following characteristics

- Decentralization. ULS systems will be decentralized in terms of data, development, evolution, and operational control
- Inherently conflicting and unknowable requirements. ULS systems will have a wide variety of stakeholders with unavoidably different and conflicting requirements
- Continuous evolution and deployment. ULS systems will be continually adding new functions and new content. Phased development or deployment is not possible
- Normal failure. Software and hardware failure will be the norm rather than the exception.

Current examples

Internet

International Telephone System

United States Electric Grid

The examples exhibit the characteristics of ULS systems

- Decentralization.
- Inherently conflicting and unknowable requirements.
- Continuous evolution and deployment.
- Normal failure.

Our expectation is that many more ULS systems will emerge in the future.

A useful analogy

Consider US Zoning process. It exhibits the characteristics of ULS systems

- Decentralization. Multiplicity of stakeholders with different and conflicting goals
- Inherently conflicting and unknowable requirements. Conflicting goals lead to conflicting requirements
- Continuous evolution and deployment. Zoning process must reflect changes in building technology, demographics, and city growth or shrinkage.
- Normal failure. Zoning code must consider fire as normal occurrence. This leads to consideration of prevention of fires and prevention of the spread of fires.

Zoning Board Stakeholders

This figure does not show the judicial system. The civil courts may be involved in appeals of the zoning board decisions and the criminal courts may be involved in cases of corruption.

Mayor, elected official or voters

selects

Member serves on

Expert advisors

Funding by government

funds

produces

Building code

constrains

Requests variance or zoning change

Approves or rejects change request

Requests variance or zoning change

Building inspector

Enforces building code

Developers

Citizen or citizen groups

What do we see from looking at the zoning board stakeholders?

Governance

- Expert advisors to board. The board sets policy, the advisor are trained in technical areas relevant to the board.
- Board generates list of rules that govern developers
- Board members are placed on the board according to some political mechanism
- Board is funded by some funding arrangements

Communities of Interest: Stakeholders or groups of stakeholders can make requests to the board for variances from or modifications to the rules.

Conformance: There is an activity that explicitly checks conformance to the rules.

Outline

What is a ULS system?

Examples that exemplify some ULS characteristics

Quality attributes for ULS systems

Summary

Examples - 1

Content Sourced Systems

- Wikipedia
  - Board of trustees acts as governing board and produces policy.
  - Board members are appointed by the current board or elected by volunteers
  - Conformance of content is performed by editors.
- Facebook
  - Privately held
  - Platform for applications. Conformance is enforced by platform API
  - Conformance of content is enforced by Facebook Corporation

Examples– 2

Platforms

- Eclipse
  - Eclipse foundation has a number of boards that provide governance
  - Board members are either volunteers or from sponsoring organizations
  - Funding comes from sponsors
  - Architecture council provides architectural guidance
  - Conformance
    - To platform enforced by API
    - Committers examine submissions
    - To architecture rules enforced by social pressure (if at all)

Eclipse architecture rules

Architecture Council of Eclipse is producing top 10 recommendations

1. Minimize plug-in dependencies.
2. Be asynchronous.
3. Don't assume your bundle is the center of the world.
4. Long-running operations should report progress and be cancelable.
5. Think API.
6. Create Unittests early.
7. Separate policy and mechanism.
8. Keep simple things simple.
9. Package coherence.
10. Be aware of the deployment context of your bundle.

Building code vs architecture rules

Building code: To Comply with Requirements of Acceptable Construction Practice Stairs Should Not Have
more than 18 risers in a flight of steps to ensure that people negotiate a limited number of steps before a landing is installed so they can rest
more than 3 winders in a ¼ landing where the going of the winders to either ¼ or ½ landings may differ from the remainder of the flight however they must be consistent within the landing and not varied individually:
Eclipse: Minimize plug-in dependencies. You will end up with easier to maintain, evolve, and easier to reuse components if you find the right granularity. One good rule of thumb is to separate UI and non-UI code into different plug-ins. Another principle is to define layers along which you can split your plug-ins. For example, put custom widgets that only depend on SWT in one plug-in, use a separate plug-in for code that additionally requires JFace, and then another plug-in that additionally requires the Workbench APIs.

Observe testability of building code and non-testability of Eclipse architecture rule.

Examples– 3

Platforms

- Apache HTTP Server
  - Governed by the Apache Foundation
  - Board of directors are elected by Apache Foundation membership
  - Funding comes from sponsors but sponsors have no particular extra rights
  - Architecture guidance comes from existing API
  - Conformance enforced by APIs and by committers

Exemplifying Examples– 4

Platforms

- Symbian OS
  - Private corporation
  - Publishes conformance document and list of test suites
  - Contributors self test and self sell
- Apple Iphone
  - Private corporation
  - Publishes conformance document
  - Apple tests for conformance. If accepted, placed in Apple store

Outline

What is a ULS system?

Examples that exemplify some ULS characteristics

Quality attributes for ULS systems

Summary

ULS systems vs normal systems

Normal Systems

ULS Systems

At an abstract level they are the same. The differences come once we begin refinement

Normal systems goals

Under control of a coherent organizational structure

Business goals*:
- Total cost of ownership
- Capability/Quality
- Market share
- Improved process
- Stakeholder satisfaction and confidence

* Taken from ATAM evaluation data

ULS Systems goals - 1

Governed by a political process.

Governing board sets policy goals (rather than business goals):
- ICANN for internet
- US Department of Energy for US energy grid
- Zoning board for our analogy

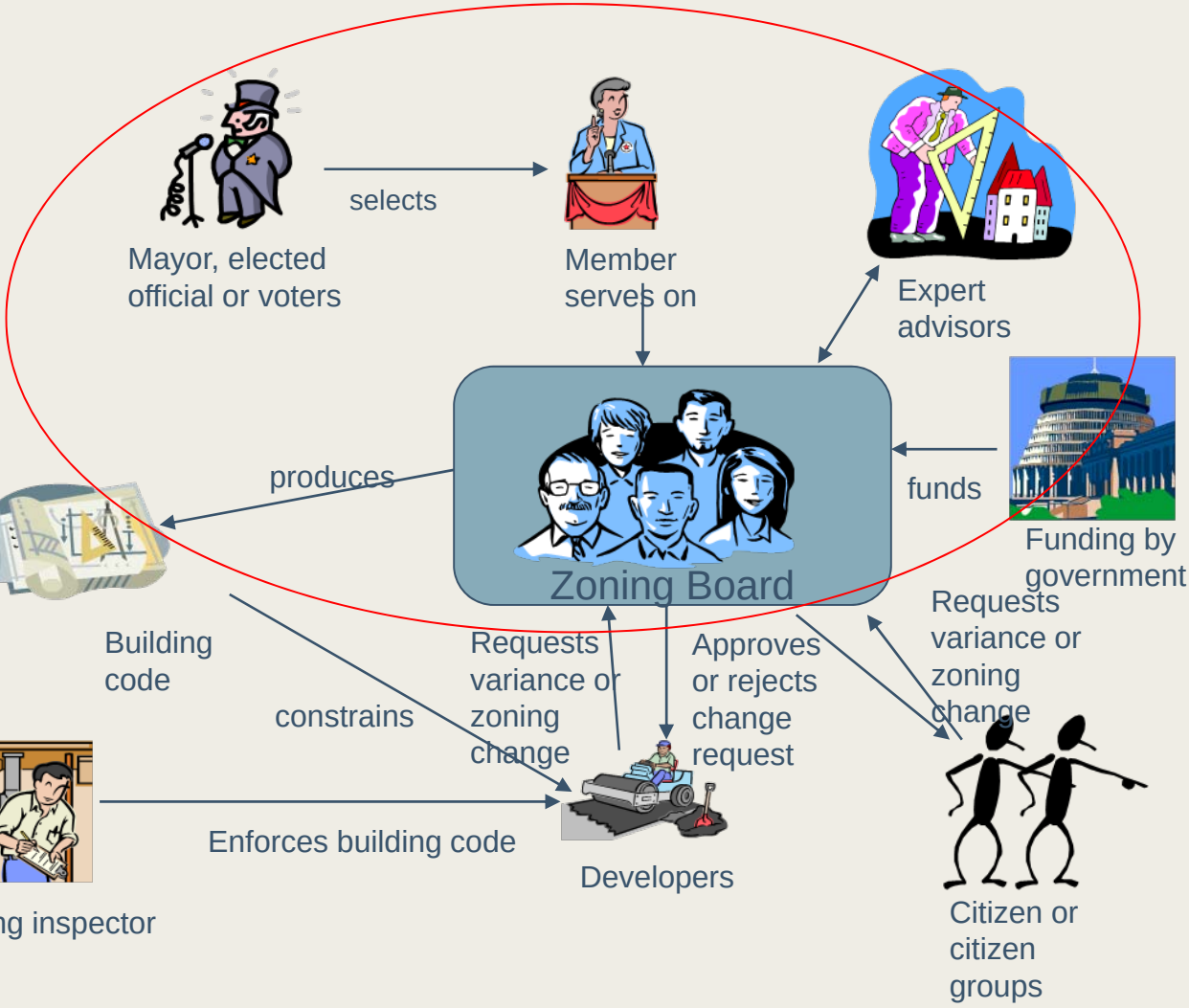
Policy board has expert advisors
- Specify detailed requirements to achieve policy
- Determine feasibility of detailed requirements

ULS systems goals - 2

Broad policy objectives

- ICANN first objective
  - Preserve and enhance the operational stability, reliability, security, and global interoperability of the Internet.
- US Department of Energy with respect to the US Energy grid
  - Make the US energy grid more secure, more reliable, safer, more economical, more environmentally friendly, more efficient.
- Zoning boards, e.g
  - maintain and enhance the quality of life for the people of <affected city>

What have we discussed?



Still to discuss from analogy

Static

- Communities of interest
- Conformance

Dynamic

- Evolution
- Emergent behavior

ULS communities of interest

Different stakeholders band together to form a *community of interest*

- Allows subgroups to interact over topics of mutual interest
- Provides a means of lobbying the policy board.
- Allows subgroups to set up policies for their portion of the ULS system (in support of or subservient to overall policies).

Communities of interest may overlap, may have sub-communities of interest, etc.

E.g. utility providers for energy grid, environmental groups for zoning board, telephone carriers for internet.

Content/conformance with respect to communities of interest

ULS systems have two types of entities for which conformance is important

- Code
- Content. E.g. wikipedia, Facebook, Youtube.

Conformance checking is done differently for these two types of entities.

Code Conformance

Conformance to ULS protocols is enforced by usage.

Code either can plug into the ULS or cannot based on whether they conform to the APIs

There may be audit based conformance testing. E.g. show me (the auditor) where your security controls are in this component.

Content Conformance

Conformance to content requirements is accomplished by some sort of censorship process, e.g. Wikipedia, Facebook, Apple Store, YouTube.

This process may be either manual or with automated tools. It may be unilateral (Apple Store) or community based (Wikipedia).

Evolution

Political processes contain a great deal of inertia

This makes them difficult to change

New versions of the protocols are introduced periodically after which stakeholders have the option of using them (typically).

This means that software that manages the ULS must support multiple protocols and that protocols must be self identifying.

Emergent Behavior

Emergent behavior is the arising of novel and coherent structures, patterns and properties during the process of self-organization in complex systems

E.g. the stock market has no single entity that sets prices but prices emerge from the components that act through an auction

In ULS systems, emergent behaviors are either

- Allowed. In this case, the policy makers may set up regulations to control allowable behavior
- Disallowed. In this case, emergent behavior is treated as a failure and controls are put in place to prevent the undesired behavior.

Quality Attributes for Normal Systems

These five quality attributes are frequently articulated as the most important for systems. Which one is more important depends on the business goals of the system*

- Availability
- Security
- Modifiability
- Performance
- Usability

* Taken from ATAM evaluation data

Quality Attributes for ULS Systems

Interoperability
Performance
Availability
Security

Interoperability

Interoperability is typically achieved by a set of protocols that participants in a ULS system must adhere to.

•Protocols support basic ULS membership.

•Communities of interest may define additional rules on top of basic protocols

•Registration/discovery process is frequently associated with protocols.

- Registry must be distributed but hierarchical. E.g. DNS servers
  - no central control
  - Allows for local response to local requests

Reliability for ULS Systems - 1

Any portion of a ULS system may fail.

The critical requirement is that the system itself does not fail.

This is similar to the telecommunications concept of failure

- any call may not go through
- the network must remain up
- the outage should be insignificant relative to the total system performance

Reliability for ULS Systems - 2

Failures must be contained locally because there is no central control.

System adjusts itself (self healing) to account for failures
- Rerouting of messages
- Alternative protocols
- Recognition of failure and reconfiguration

e.g. TCP recognizes collisions and retries messages after random amount of time.

Security

Collection of security controls in ULS systems is the same as in normal systems

Controls that depend on availability of components may not be appropriate because of the assumption of normal failure.

Trust issues must be resolved locally among peers. E.g.

- Certificates. Global issuing body but *a priori* to any particular communication.
- One time passwords and tokens.
- Failures in trust must not bring down system (availability). E.g. expired certificate may cause local failure but not global failure

Performance

Rapid response time

Scalability

Rapid response time

Use of protocols to achieve interoperability eliminates some performance tactics
- Scheduling strategy
- Shared memory

Other performance tactics are relevant
- Caching
- Expandable resources for computations (e.g. more servers)
- Stateful systems
- Compression

Scalability

Peer to peer systems

- A node locates its neighbors and determines their properties
- Nodes are self describing
- Nodes run in parallel

Well defined building blocks

Models of the environment

Self similar structure – aggregations from an external perspective are similar to the elements from an internal perspective

Outline

What is a ULS system?

Examples that exemplify some ULS characteristics

Quality attributes for ULS systems

Summary

Summary - 1

There is a set of protocols that define the base level of coordination among the ULS system elements. These protocols are governed by a stakeholder dominated group and are oriented toward providing a base level of service for all stakeholders within the ULS community.

Communities of interest within the ULS ecosystem define their own set of protocols that exist on top of the base level. These communities can be dominated by stakeholders or directed by individual stakeholders. The communities of interest dominated by individual stakeholders tend to manifest themselves as a platform. Those dominated by multiple stakeholders can either be platforms or interaction protocols.

Summary - 2

The protocols (both for the ULS and for communities of interest) emphasize local behavior and achieve global behavior through careful selection of acceptable local behavior. A portion of this local behavior is explicit understanding that failure is normal rather than exceptional. One aspect of assuming normal failure is that the emphasis is on ensuring that failures within a particular community of interest do not propagate to the whole ULS system.

Summary - 3

Once established, the protocols used in ULS systems are difficult to change and so versioned protocols and interoperability among different versions of a protocol are necessary.

Conformance is performed with a combination of semi-automated and manual means.

- Code conformance is done with test suites and check lists
- Content conformance is done with content analyzers, organizational censors, and public censors.

More information

ULS report – www.sei.cmu.edu/uls

Len Bass
lenbass@cmu.edu