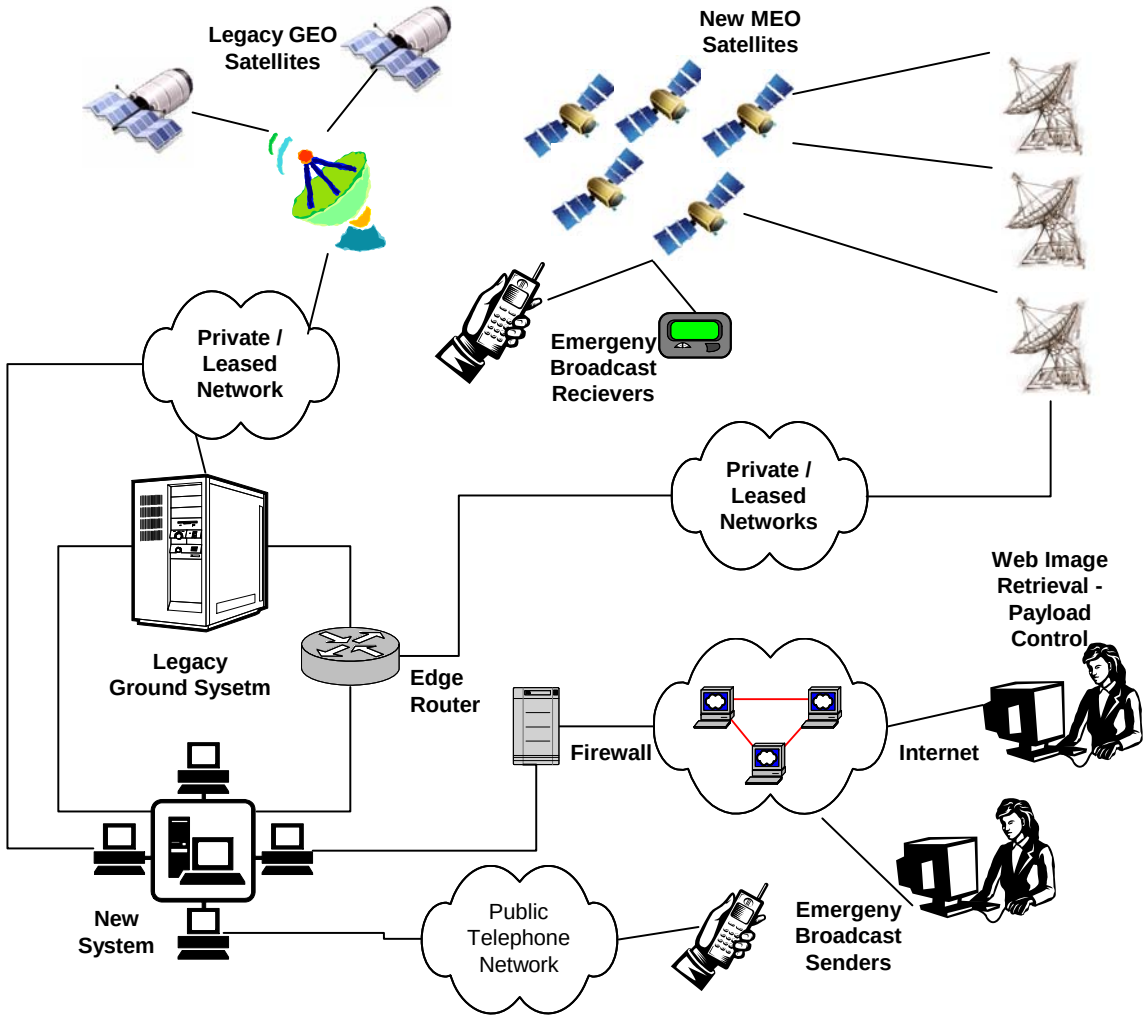
Architecture Centric Evolution

Jeff Garland
CrystalClear Software, Inc
jeff@crystalclearsoftware.com

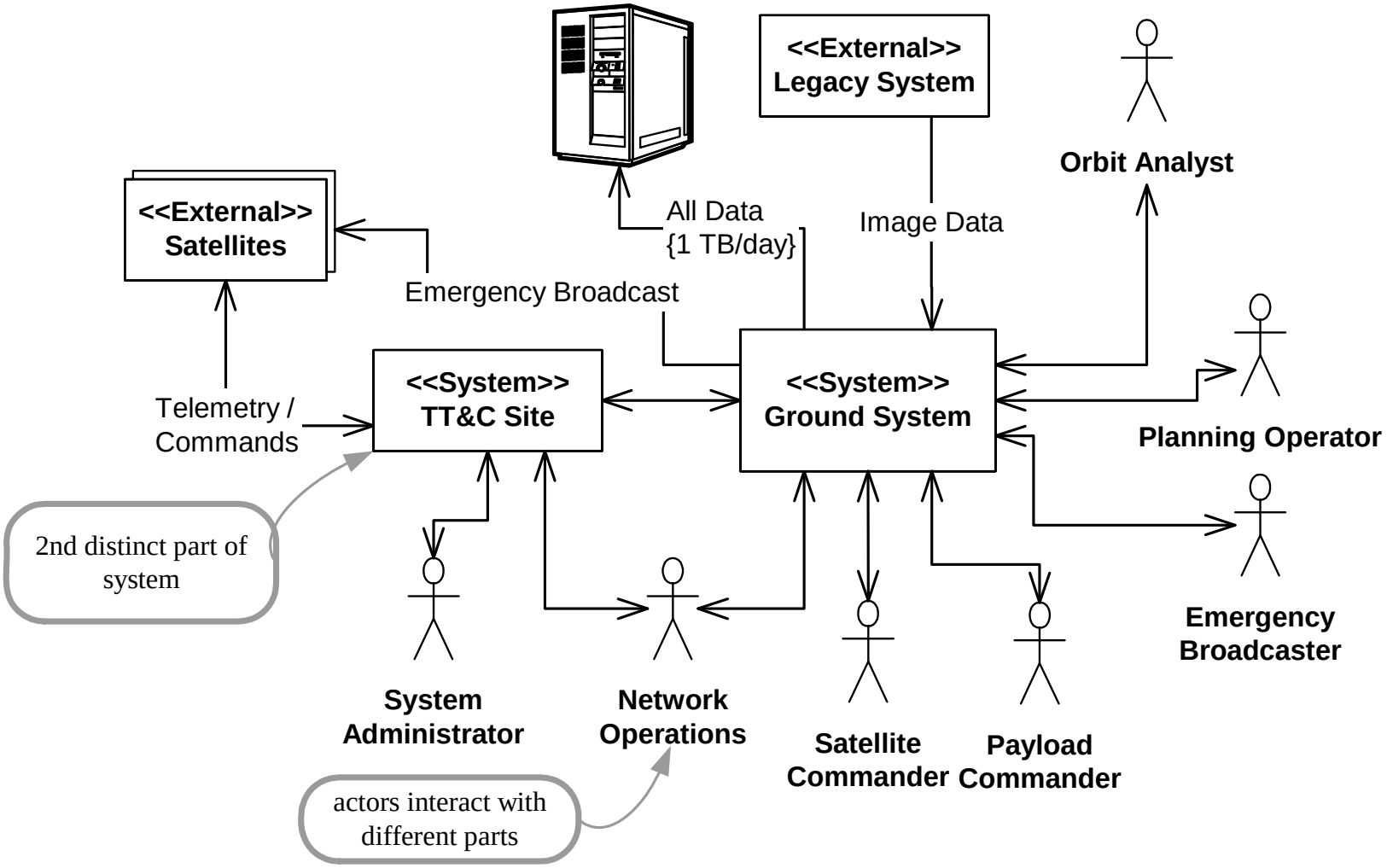
Changeability – Three Themes

- Key architecture attribute for highly modifiable systems
  - Testability
  - Buildability
- Themes:
  - Software architecture must first be realized to be relevant to evolution
  - Using standard architectures and tools is essential or mechanisms matter
  - Changeablity – at what risk and cost?

Conceptual Diagram Example



Context View Multi-Central Example



Essential Requirements for System/Software Architecture Representations

- Must provide multi-dimensional leverage in understanding very large software systems
- Must have well defined mapping to the implementation
  - Implementers need to believe it and trust it
  - Allows us to reason about design decisions
  - Allow for consistent forms of abstraction
- Must allow multiple implementation languages/technologies
- Must be tractable to create and maintain – even without sophisticated tool support
- Must be easy to learn and use

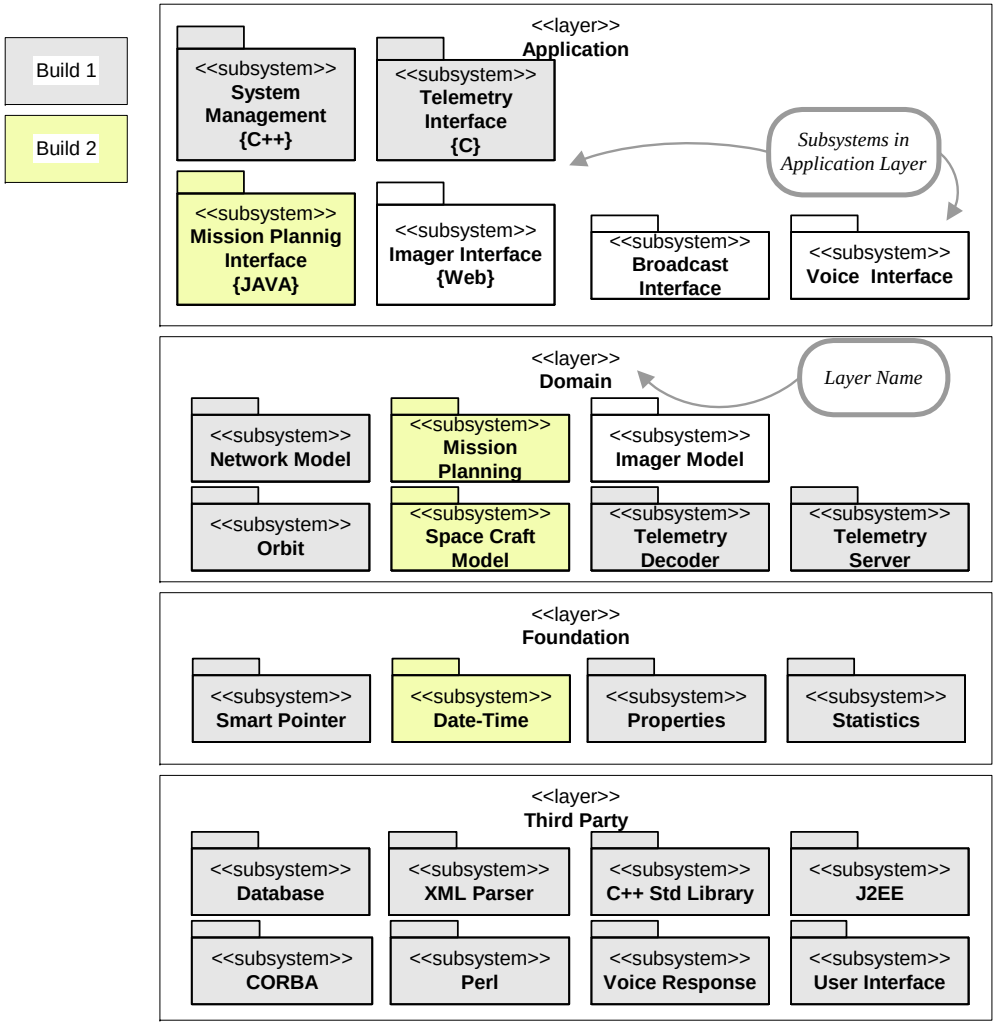
Mechanisms Matter – Prefer Open Tools and Standards

- Thought experiment – need to change the ground system to perform a complex series of command actions on one or more satellites…
- Solution 1: Domain specific satellite commend scripting language
  - Limited tutorials and examples
  - No software developers come to project ‘pre-trained’
  - Likely limited base libraries (eg: collections, regular expression matching, etc)
  - Difficult to extend
- Solution 2: Build a commanding library in an ‘Open Language’
  - Build it with SOA and all you worries are over…
- ***Nothing about satellite commanding requires a domain specific language***

Architecture Buildability / Testability

- Cycle Time of Software Change
  - Problems:
    - Ground systems complex and distributed
    - Software is complex to build / setup
    - Limited lab/hardware resource
  - Software Developer: Most time isn't spent on the needed software changes
- Architecture can be built to be more testable
  - Testing of the parts
  - 'Unit Tests'
  - Layered Architecture

Layered Subsystem View Example



Final Thoughts on Changeability

- Only Will Achieve in Dimensions Set Forth In Requirements
- Some Types of Ground System Adaptation
  - Changes to satellite(s) -- upload new software
  - Add new satellites to the system
- These Constitute “Change Cases” in Requirements
  - YAGNI
  - YNGTGI
- Cost of Adaptability
  - Need to design and test the adaptability of system
  - Lot’s of ‘general’ systems that have failed
    - Most systems complex to start
    - ‘Meta-processing’ is particularly hard