Dynamic Architectures: Change Notification Languages

David Wile, Teknowledge
Dwile@teknowledge

Outline

- What does "self healing" mean to you?
- What part of the self-healing problem are you dealing with?
- What part are you not dealing with?
- What applications are you targeting?
- What are the top two/three new technical ideas/approaches that you are pursuing in this work?

What does "self healing" mean to you?

- Perturbation tolerance
- Dynamic adaptation to new situation
- System “understands” its health status
  - Performance improvement as a byproduct
- Successive layers of response:
  - Autonomic: *instinctive*, immediate response to trouble
  - Guided: *planned* or dynamically-adaptive activities to repair or improve
  - Cooperative: *negotiation* process required to resolve problems
- The role of models
  - None in autonomic – models that work are compiled in
  - Planning requires abstractions to characterize “health” status
  - Reflection useful for both guided and cooperative

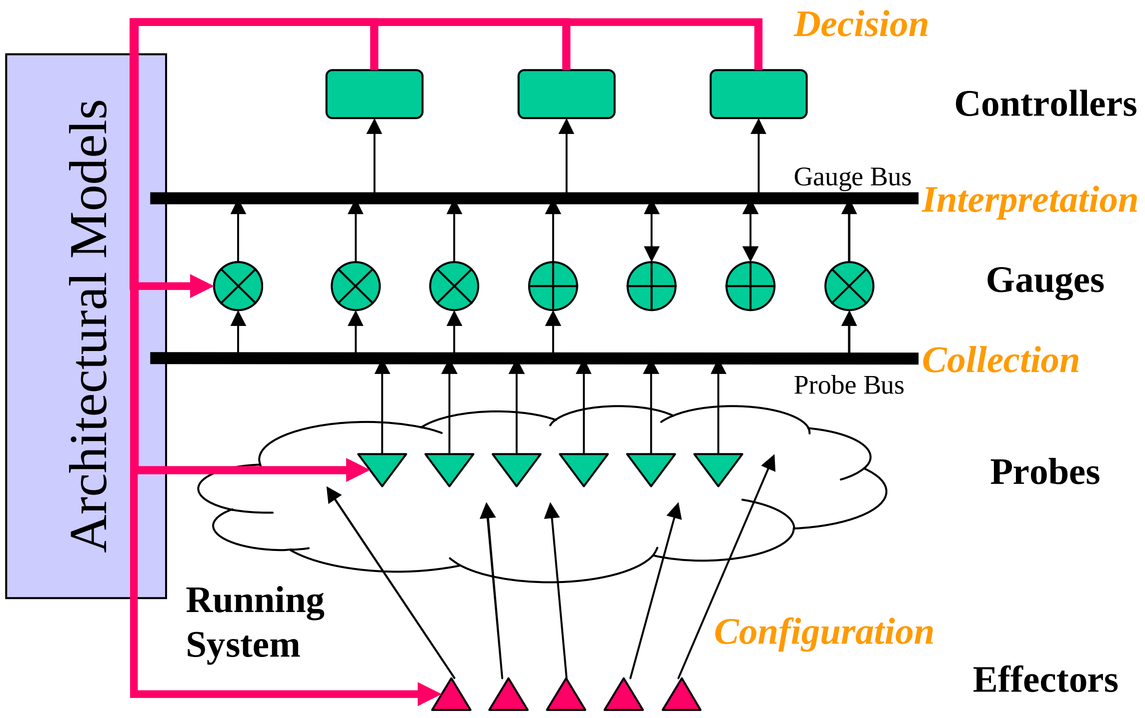
What part of the self-healing problem are you dealing with?

- Guided / planned responses
  - System structure models
  - Application structure models
  - Both covered by architecture modeling
    - Probe system
    - Map to application

DASADA Common Infrastructure

- Develop reusable probe/gauge/repair framework for DoD software systems developers
- Challenges for developers of systems where application and adaptation are interwoven (autonomic!):
  - Internal adaptation makes it difficult to change adaptation policy and mechanism
  - No reuse of adaptation mechanisms between applications
  - Hard to reason about adaptation mechanism or application itself independent of knowledge of the other
- Mission: Provide developers with an *Externalized Infrastructure* for
  - Monitoring their system (automatic probe placement)
  - Interpreting measurements (architectural models and gauges)
  - Adapting their system (automatic adaptation mechanisms)

Infrastructure Architecture



Static Architecture Scenario

- Probes and gauges are placed via the control layer.
- Probes emit implementation-level events (ILEs)
  - “process D006 opened file ‘C:\Program Files\log.txt’ for write”
  - “process E001 used 2021.”
- Gauges provide interpretations of these events
  - determine logical architectural entities are referred to
    - “Radar Tracker” (D006)
    - “Radar Analysis” (E001), for example.
  - This mapping determined by the processes that originally set up the system and probes.
  - Gauges additionally interpret implicit information from the probes
    - perhaps 2021 means 2021 microseconds.

Static Scenario continued

- Gauges are “read” by the control layer to determine action to take
  - If ILE for E001 is interpreted as “Radar Analysis took 2021 microseconds to process the last scan.”
  - And the analysis module is a function of the parameter, ScanGrain.
  - The control layer communicates to effector layer
    - Coarsen ScanGrain for Radar Analysis to 5 degrees / scan.
- Effector layer determines what physical process needs to be adapted (E001)
  - Determine what process variable of E001 corresponds to ScanGrain
  - Reset to reflect the 5 degrees / scan modification.

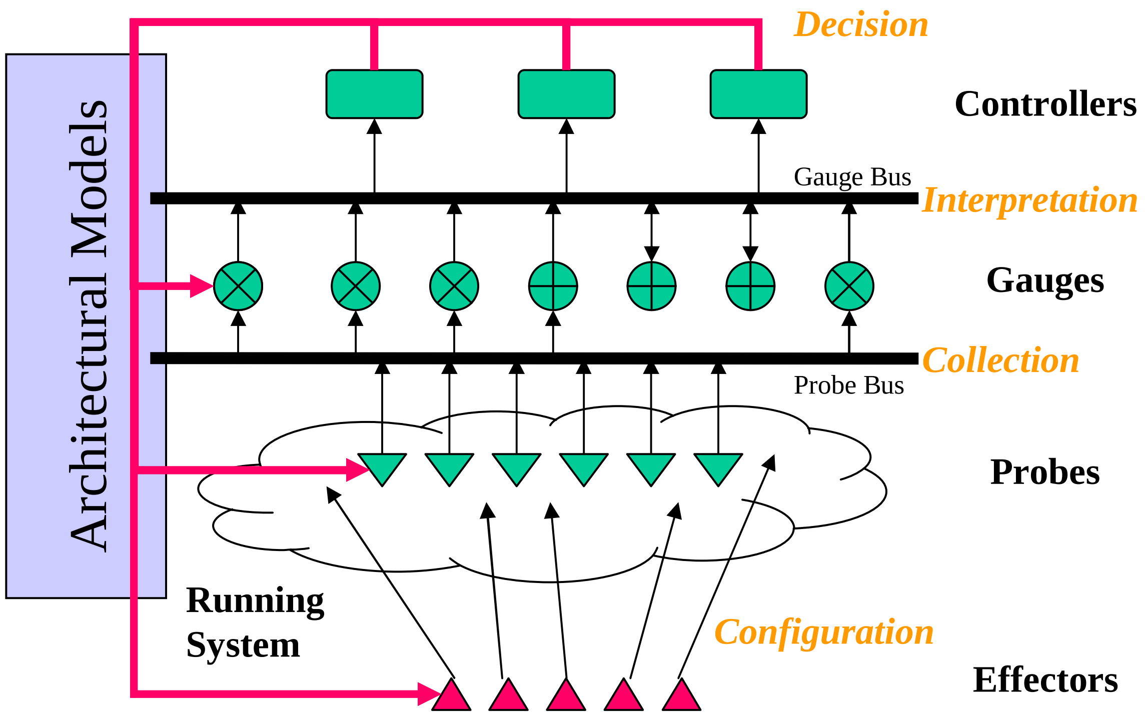
Dynamic Architecture Scenario

- Probes and gauges are placed via the control layer.
- Probes emit architecturally significant implementation-level events (ASILEs)
  - “process D006 *spawned new process* E001 of type RAN”
  - “process E001 *requested socket* 239.”
- Gauges modify corresponding physical and logical models.
  - E001 of type RAN => identify the E001 process with (previously unidentified) logical process, “Radar Analysis.”
  - I call this process *identification* of physical models logical architecture models
  - “proto-architecture” -only identified modules and connectors constitute actual logical architecture.

Dynamic Scenario continued

- Same scenario as above, “process E001 used 2021,”
  - the control layer at this point may want to change the system’s running architecture by issuing a *reconfiguration* event to the effector layer
  - “replace Radar Analysis type RAN with RAAN” (another radar analyzer type, perhaps with a coarser scan rate).
- Effector layer again maps logical Radar Analysis component onto E001
  - also has to understand how to remove that component
  - substitute a new one of type “RAAN.”

Infrastructure Architecture



xAcme Protocol

```
created (creations::
       [newComponent |
         newConnector |
         newProperty
            property::<properties:Property> …]
       context:: <instance:XMLLink >)
deleted( elementType::
                     (deletedComponent |
                     deletedConnector |
                     deletedProperty ),
       deletedElement:: <instance:XMLLink>)
attachedConnector(pairs::
       [(roleName:: <instance:XMLLink>,
         portName:: <instance:XMLLink>)]
detachedConnector(pairs::
        [(roleName:: <instance:XMLLink>,
          portName:: <instance:XMLLink>)]
```

xADL Protocol

```
Diff (changes::[(add(Add) | remove (Remove))])
Add ((component(<types:Component>) |
     connector(<types:Component>)|
     link(<types:Link>) |
     group(<archinstance:Group>) |
     componentType(< types:ComponentType >) |
     connectorType(< types:ConnectorType >) |
     interfaceType (<types:InterfaceType>))
Remove(removeID::<archinstance:Identifier>)
```

Consolidation Issues

- Hidden (xADL) vs Explicit (xAcme) structure
  - Former allows complex structures to be altered, but requires everyone receiving the events to understand the implicit structure
  - Latter allows coarse models to be formed by anyone receiving the events (want to refine as much as possible*)
- API vs Event model vs (Single-source) Broadcast
  - API = single consumer event model implementation
  - Event model requires a transaction model; otherwise it is just an API
  - Single-source broadcast allows multiple listeners without synchronization issues

More Discussion Issues

- Goals for the protocol.
  What belongs in the protocol?
  - Core (Syntactic)
  - Constrained (Type checked)
  - Completed (Analyzed) *
  - Reflective (2nd Order Representation)
- Nomenclature issues.
  *Can we agree on a nomenclature * or is a Rosetta Stone appropriate?*
- How many different representations of the events are needed?
  *Is XML sufficient? (Probe Protocol *)*

Discussion Issues continued

- How rich should the event language be?
  Union? Extensible core?
- What transaction model should be used?
  Explicit begin-end, nested transactions, set of changes, sequence of changes, higher-level operators encapsulating sequences - such as “change” for “remove and then add.”
- How does one identify an architectural element uniquely?

What part are you not dealing with?

- Approaches
  - Autonomic
  - Cooperative
- Layers
  - Control
  - Repair
  - Probing (here)

What applications are you targeting?

- COTS-based
- Air Force Heads Up display – “Master Caution Panel”

Top 2/3 new technical ideas/approaches?

- Externalization
- Reflection