Paper Presentation

Dynamic Evolution of Context and Corresponding Self Configuration in Service Oriented Cyber Physical System

Presented by: Muhammad Salman Aslam  (G-00763290)
Department of Computer Science George Mason University
(12/04/17)
SOA and Optimal Service Composition

- Composed of services
  - Distributed Design
  - Standard interfaces and communication protocols
  - Loosely coupled services
  - Functional Demarcation

- Optimal Selection of SOA architecture
  - Selection if service components
  - Selection Service providers
  - Re configuration and re architecting
  - Optimal Allocation to meet SLAs

- Self Configuration - adaptation with changing context
Layered SOA Optimization model described by Alferiz et.al. [5]
- Design Phase
  - Requirements/Traffic Model
  - Variability Model
  - Configs gen
  - Context
  - Weaving
- Adaptation Phase
  - Adapted Variability
  - Adapted Composition
- Evolution phase
  - Evolves Composition
  - Evolve Variability
Context, CAS

- “the circumstances that form the setting for an event”. Alagar et.al.
- Any information that represents a change in an entity important to a systems
  - Change in Environment
  - Change in User
  - Change in computing platform
- Context Aware Systems
  - Sense and compose context information from different sensors;
  - Autonomously detect relevant changes in the context in order to dynamically adapt their services;
  - Interoperate with third-party service providers
Where is the Context Most Important

- Context is most important for systems where
- Dynamic
- Distributed
- Users may be added. User roles may change
- Actions and activities can not be all known priori
- Decision making is non deterministic – needs intelligence
- Large systems with many interconnection
- Distributed applications and services
- Environment changes are large and unpredictable
  - High Interaction with the Physical world

- Characteristics of Cyber Physical Systems
Characteristics of CPS

- Interact with environment
- Heavily Networked
  - Components interconnected
- Distributed
  - Components spatially spread
- Integrated
  - Tightly coupled components (usually)
- Time Constrained (Real Time)
  - Hard or Soft Real time constraints
- Dynamic / Configurable
- Dependable/Reliable
- Secure
Importance of Context in CPS

- **Consider Reconfiguration Services**
  - Patients Population has gown the ward needs expansion
  - Before moving the wall consider whether the equipment is all off and ready to move + no occupancy in adjacent rooms + Authorization etc

- **Navigation Service**
  - Service routing visitors and staff
  - Consider Critical patient in the Hallway
  - In this Context Nav Service needs to advise users to stop
  - What if there is an Emergency Doctor using he Nav application?
Model to Derive Context and Evolve Context Knowledgebase
Where to model the Context

- Context Aware Service compositions
- Service Providers
- Service
- Middle ware
- Application
  - Coordinator
  - Eg. Smart car, Smart space
Using SOA Adaptation for Context Evolution

- The figure shows the Fragmented SOA Adaptation Model.
- Context model is built in a closed world assumption (Circled Red)
- We need to move the Context Model to open world assumption and move to the Dynamic Evolutionary layer.
- How to Evolve Context?

Context = Known Context $C_{(k)}$ U Unknown Context

Context $C_{(k)} = \sum$ unknown context events that can be handled

Unknown Context = $\sum$ unhandled unknown context events

$C_{(k)} = C_{(k)} U c_{(new)}$
How to model the Context

- Various Ontologies – we Used one by Hatim et. Al [6.10]
  - Model using simple constructs
    - Number of occupants >0
    - Light – Low /Off
    - Machine X - Ready

Various levels - context divided into categories
- User
- Environment
- Devices
- Application
How to Model Context

- **Captured In simples Propositions**
  - Context Properties
  - Joined together by Operator

- **Stitching**
  - Similar Context can stitch together with operators

- **Property**
  - sensedProperty
  - derivedProperty
  - storedProperty

- **Operator**
  - Logical Operator - Complex operators
How to Model Context

- **Propositions**
  - Context Properties, Joined together by Operator

- **Stitching**
  - Similar Context can stitch together with operators
  - Operator Build together the sub category into a tree like structure c-tree

- **Variability**
  - As a Sensor is added its properties are added to the leaf node
  - In all the tee structures available the properties for the other devise are tried incrementally and removed if not applicable as per the validation rule

Sub-Cat <Occupant> OP (P1, P2)
Sub-Cat <Environment> OP((P1, P2, P3), OP (P3, P4)
Sub-Cat <SC3> OP(.....)
Sub-Cat <SC4> OP(.....)
Sub-Cat <SC5> OP(.....)

OP(Sub-Cat1, Sub-Cat2, Sub-Cat3 ...... SubCatn)
Algorithm

Generate Context
- Generate (context, user) based on new context and request data
- For each variation in op in op set
  - Add node to C-Tree
  - Use the Context Knowledge base discard the nodes and connections
- End For
- For each sub tree in the C_Tree DFS
  - Validate the C_tree with the validation rules and Prune any edges
  - Accept_Set <- Null
- Do Until all Service explored in RR select service or Maximum Tries exhausted
  - If Service Request fails try next
  - Validate response against the response checker
  - If Response Accepted
    - Check Response SLA
    - If the Service response within SLA response time, Add to Context_Accept set
  - Else
    - Discard and try next service
- End Do
- If Accept_set not Null
  - Add the Accept_Set data to Context Knowledge base
- Else
  - Present the Context to Expert // Expert may add the context to Discard Knowledge Base
- Remove Context from ContextNew

END For

End
Evaluation

- **Required**
  - Context Aware services
  - CPS – Simulated architecture
  - Coordinator Designed
  - Run simulations by adding Devices/Sensors
    - Introducing properties

- **Comparison to be made**
  - Added context to Knowledge Base
  - Ability to resolve context I lower response time
    - With Knowledge Base and with out

- In Progress - TBC
References

- Ewing John, Autonomic Performance Optimization with Application To Self-architecting Software Systems , Doctoral Dissertation, Menascé, Daniel A. (advisor); Gomaa, Hassan (committee member); Malek, Sam (committee member); Neyfach, Stephen (committee member); George Mason University, 2015
- Hatim Hafidi, Hicham Baldouri, Mahmoud Nassar and Abdelaziz Kriouile, Context-Awareness for Service Oriented Systems,