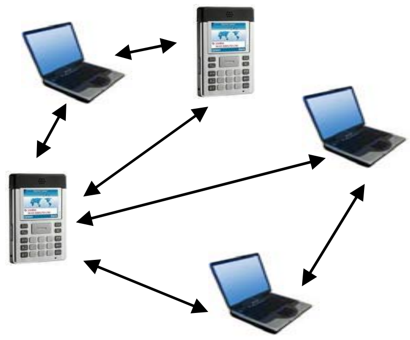
Sixth International Workshop on Dynamic Analysis (WODA), 21 July 2008

Testing mobile computing applications: toward a scenario language and tools

Minh Duc Nguyen, Hélène Waeselynck, Nicolas Rivière

HIDENETS
Highly DEpendable IP-based NETworks and Services

Mobile computing systems



- Dynamicity of system structure
- Communication with unknown partners in a local vicinity
- Context awareness

➡ Solutions for the testing of applications and services in mobile settings

Outline

- State of the art in testing traditional/mobile distributed systems
- Case study: a Group Membership Protocol (GMP) in the ad hoc domain
- Toward a scenario language & automated support for mobile computing applications
- Conclusion and perspectives

Testing: state of the art

- Traditional distributed systems
  - Platforms with dedicated test interfaces, dedicated test languages (e.g. TTCN-3)
  - Use of graphical scenario languages (MSC, UML SD) to support design & validation activities
  - Formal approaches in the protocol community

    SDL model $\times$ test purposes $\rightarrow$ test cases
  - Passive testing approaches

- Mobile computing systems
  - Experimental platforms with simulation facilities (mainly for evaluation purposes)
  - Testing issues have been little explored so far
  - Pioneering work based on SDL models (but SDL is not well-suited to mobile settings)
  - No established modeling framework for mobile computing systems

Outline

- State of the art in testing traditional/mobile distributed systems
- **Case study: a Group Membership Protocol (GMP) in the ad hoc domain**
- Toward a scenario language & automated support for mobile computing applications
- Conclusion and perspectives

A Group Membership Protocol (GMP)

- References
  - "Relying on Safe Distance to Achieve Strong Group Membership in Ad Hoc Mobile Environments " , *IEEE Transactions on Mobile Computing, Washington University*
  - Open-source implementation in the mobility-oriented LIME middleware (http://lime.sourceforge.net)
- Goal
  - Consistent view of the group members while groups merge and split according to location information
- The GMP case study:
  - Exemplifies a classical problem in distributed computing
  - Still, the mobile settings particularizes the problem
  - High dynamicity, high dependency on location & movement patterns
  - Example in the ad hoc domain
  - Not trivial (each node = 4KLOC of Java code, 6 concurrent threads)

GMP principle

- Notion of safe distance
  - Nodes are “close enough” to prevent motion-induced disconnection for some time (assuming an upper bound V_{max} on speed)

1 —safe— 2 —safe— 3 —safe— 4

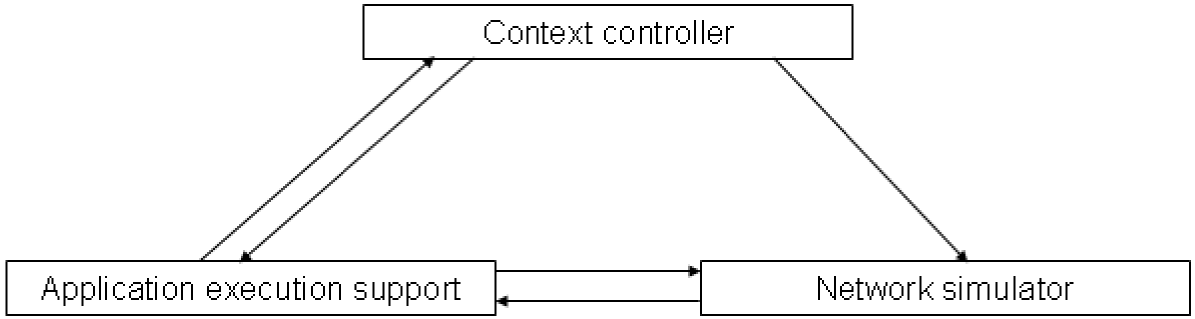
Accomodates multi-hop communication

- Requirements: 8 properties (local & global)

- GMP analysis and testing
  - Review of the paper specification
  - Reverse engineering of the source code to produce UML models
  - Test experiments using a synthetic workload (random movement of nodes)

Insights gained from the case study

- Test platforms need network and context simulators



[Ricardo Morla et al.] & [Christoph Schroth et al.]

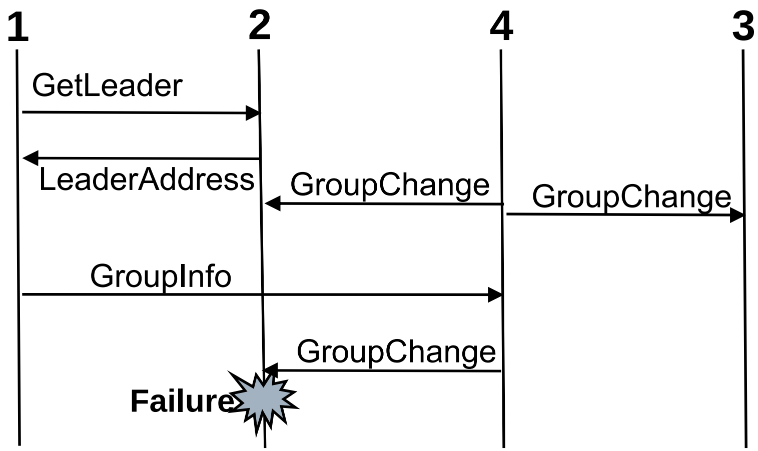
- Adequate formalisms to support design & validation activities?
  - Standard UML: OK for modeling one GMP node
  - System-level behavior and structure?

- Graphical scenario descriptions appear a useful support but:
  - Usual scenario languages need extensions to account for mobile settings
  - Production of concrete contextual data (e.g., location coordinates) to instantiate an abstract scenario?

Outline

- State of the art in testing traditional/mobile distributed systems
- Case study: a Group Membership Protocol (GMP) in the ad hoc domain
- **Toward a scenario language & automated support for mobile computing applications**
- Conclusion and perspectives

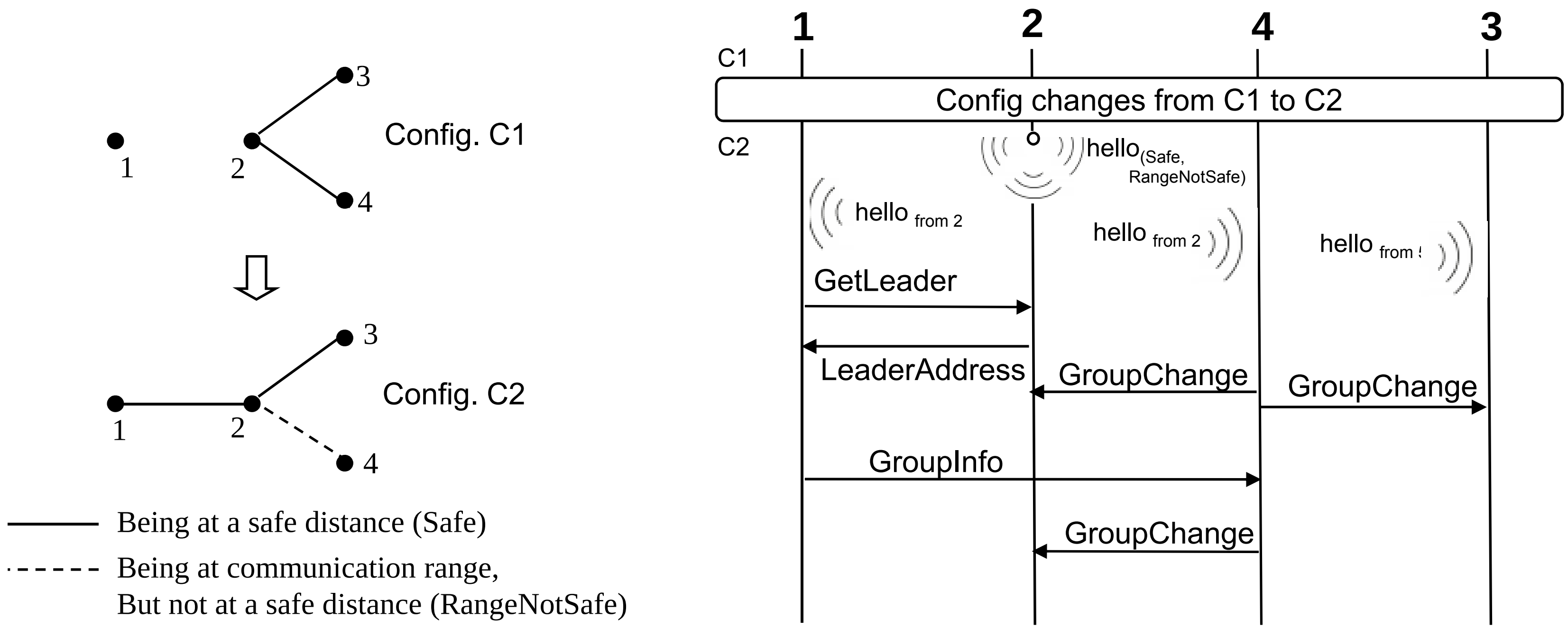
Describing scenarios from the case study



Example of a split & merge fail scenario

- MSC-like languages: focus on the partial order of communication events
- But the underlying spatial configuration is equally important to characterize the split & merge scenario
- Absence of broadcast constructs
- How to represent broadcast in local vicinity (here, « hello » message from 2)?

Scenario language for mobile settings



(a) Spatial view

(b) Event view

- ✓ Labeled graphs for the spatial configurations
- ✓ Configuration changes as global events, causally related to communication events
- ✓ Topology-aware broadcast primitives

Automated support

To check whether an execution trace satisfies a requirement
To check whether an execution trace covers a test purpose

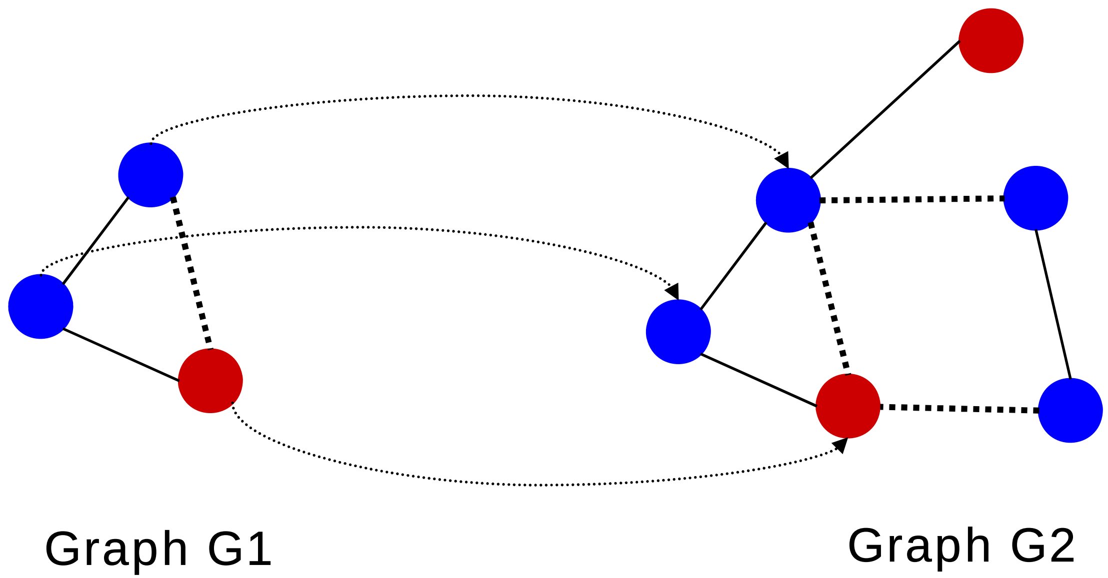
1. Determine which physical nodes match the nodes specified in the spatial view
2. Analyze the order of events in the identified configurations

To assist in the production of contextual data for implementing a test case (principle: extract data from random simulation runs)

1. Run the context controller and record contextual data at each simulation step
2. Abstract the simulation trace by series of graphs
3. Search whether subgraphs can match the desired evolution pattern
4. List of matches = baseline configurations for the implementation of the scenario

⇨ Importance of graph matching problems

Automated support (2)



Does G1 appear as a subgraph of G2?

⇩

Build a graph homomorphism from G1 to G2

A graph homomorphism from $G_1 = (V_1, E_1, \lambda_1, \mu_1)$ to $G_2 = (V_2, E_2, \lambda_2, \mu_2)$ is an injective function $f : V_1 \rightarrow V_2$ such that:

- $\lambda_1(v_1) = \lambda_2(f(v_1))$ for all $v_1 \in V_1$
- For any edge $e_1 = (v_{1s}, v_{1e}) \in E_1$, there exists an edge $e_2 = (f(v_{1s}), f(v_{1e}))$ such that $\mu_1(e_1) = \mu_2(e_2)$

Automated support (3)

- Some convenient extensions to the basic definition:
  - Allow for tuple of labels, e.g. node can be characterized by <id, type>
  - Allow for label variables, e.g. nodes <*x*,*"Mobile"*> and <*1*,*"Mobile"*> can match using substitution *x:=1*

Graph matching

=

Mapping of nodes

+

Valuation that unifies the labels

Implementation

- Based on an existing graph tool (developed at LAAS)
  - ✓ Input: a graph G1, a graph G2
  - ✓ Ouput: all homomorphisms from G1 to G2

- Our work: search for **sequences of configuration patterns** in a concrete trace

  Patterns: P1 → P2 → ... → Pm

  Trace : C1 → C2 → ... → Cn

  (Note: a configuration pattern Pi may occur in several consecutive Cj before the configuration changes to Pi+1)

- Fixed number of nodes in patterns

- Nodes may appear and disappear
  - ✓ This introduces some additional concerns...

Outline

- State of the art in testing traditional/mobile distributed systems
- Case study: a Group Membership Protocol (GMP) in the ad hoc domain
- Toward a scenario language & automated support for mobile computing applications
- **Conclusion and perspectives**

Conclusion and perspectives

- Proposition of extensions to better represent scenario descriptions in mobile computing settings
- Processing of scenario descriptions, based on graph matching algorithms
- On-going work
  - Scenario language for mobile settings
    - ✓ Extensions of UML 2.0 Sequence Diagram
    - ✓ Compromise: expressiveness / well-defined semantics
  - Support for automated comparison of scenarios and traces
    - ✓ Spatial view: Optimizations required to handle large simulation traces, consideration of min-max duration constraints
    - ✓ Event view: Comparison of the order of events: will be implemented once the language is stabilized