Executable Modeling of Deployment Decisions for Resource-Aware Distributed Applications

Silvia Lizeth Tapia Tarifa

Precise Modeling and Analysis Group
Department of Informatics
Faculty of Mathematics and Natural Sciences
University of Oslo

sltarifa@ifi.uio.no

07.05.2014
Overview

Deployment

Various activities that place a software application on computing devices

- **Problem:** Performance during the operational phase is below expectations
- **Why?** Characteristics of the deployment infrastructures influence performance
- **Possible actions to improve performance:** Redeploy, redesign applications
- **Hypothesis:** It is advantageous to include deployment decisions in the design phase
Research Goal

*Develop a methodology to express and compare different deployment decisions for resource-aware distributed applications during the design phase.*

Requirements:
- Concurrent, distributed and object-oriented framework
- Model based approach
- Formality

Integrate in a modeling language:
- Deployment architectures
- Deployment decisions

In order to:
- Estimate and quantify QoS
- Compare deployment architectures
**Overall Contributions of this Thesis**

- **Starting point is Core ABS:**
  - Executable and object-oriented modeling language with a formal semantics and a Java-like syntax
  - Clear and simple concurrency model (using COGs) with synchronous and asynchronous communication

- **Formally defined extensions of Core ABS:**
  - Explicit and implicit modeling of dense time (Real-Time ABS)
  - User-defined cost annotations
  - Deadlines to method calls
  - Modeling of deployment architectures
  - Resource management: load-balancing and user-defined schedulers

- **Validation of methodology and formalization**
  - Modeling examples and case studies
  - Analysis results: Simulations

- **Publications:**
  - This thesis collects four paper (two journal papers, two conference papers)
  - Eight additional papers (not included in this thesis)
Outline

- Motivating example
- Research goal and research questions
- Summary and future work
## Outline

- Motivating example
- Research goal and research questions
- Summary and future work
Example definition: Phone Services
**Example: Phone Services - Real-Time ABS**

### Telephone Service

```java
.interface TelephoneService {
    Unit call(Int calltime);
}

class TelephoneServer implements TelephoneServer {
    Int callcount = 0;
    Unit call(Int calltime) {
        while (calltime > 0) {
            calltime = calltime - 1;
            await duration(1, 1);
        }
        callcount = callcount + 1;
    }
}
```

### SMS Service

```java
.interface SMSService {
    Unit sendSMS();
}

class SMSServer implements SMSServer {
    Int smscount = 0;
    Unit sendSMS() {
        smscount = smscount + 1;
    }
}
```
Example: Phone Services
Client Behavior on New Year’s Eve

Simulate the operation of the system outside standard usage.
Client can flood the system and create congestion
Example 2: Phone Services - Client Handsets

```java
class Handset { Int cyclelength, TelephoneService ts, SMSService smss) {
    Bool call = false;

    Unit normalBehavior() {
        if (timeValue(now()) > 50 && timeValue(now()) < 70) {
            this!midnightWindow();
        } else {
            if (call) { ts.call(1); } else { smss!sendSMS(); }
            call = ~call; await duration(cyclelength, cyclelength);
            this!normalBehavior(); }
    }

    Unit midnightWindow() { Time t = now(); Int i = 0;
        if (timeValue(now()) >= 70) {
            this!normalBehavior();
        } else { Int i = 0;
            while (i < 10) {
                smss!sendSMS(); i = i + 1;
            }
            await duration(1, 1); this!midnightWindow();
        }
    }

    Unit run() { this!normalBehavior(); }
}
```
How a functional model will perform in a given deployment architecture
Deployment Components (new constructor)

- **Locations** for concurrent objects in ABS
- Each deployment component has a given resource capacity
- Objects execute in the context of a deployment component
- The *resources are shared* between the component’s objects
- Object execution *uses* resources in a deployment component (via Cost annotation)

- *How* resources are assigned and consumed depends on the kind of resource
- **Phone services** example: computing capacity per time interval
Example: Phone Services - Deployment Modeling

// Main block:
DC smscomp = new cog DeploymentComponent("smscomp", CPU(50));
DC telcomp = new cog DeploymentComponent("telcomp", CPU(50));
[DC: smscomp] SMSService sms = new cog SMSServer();
[DC: telcomp] TelephoneService tel = new cog TelephoneServer();
c = new cog Handset(1, tel, sms); ... // Clients handsets
Example 1: Phone Services - Simulation Results

Can the performance of the phone service model be improved?

Can the computing resources of a deployment architecture be better used?
Example: Phone Services - Resource Management

Resource management using a simple load balancing strategy

Example strategy: Reallocate $\frac{1}{3} \times$ total resources upon request from partner
Example: Phone Services

A Simple Load Balancing Strategy

```java
class Balancer implements Balancer {

    Balancer partner = null;

    Unit run() {
        await partner != null;
        while (True) {
            await duration(1, 1);
            Int tl = thisDC().total(); Int ld = thisDC().load(1);
            if (ld > 90) {
                Fut<Unit> r = partner!requestdc(thisDC()); await r?;
            }
        }
    }

    Unit requestdc(DC comp) {
        Int tl = thisDC().total(); Int ld = thisDC().load(1);
        if (ld < 50) {
            thisDC()!transfer(comp, capacity(tl) / 3);
        }
    }

    Unit setPartner(Balancer p) { partner = p; }
}
```
Example: Phone Services
Deployment Scenario with Balancers

Main block of the Real-Time ABS model with load balancing

```cog
{ // Main block
    DC smscomp = new cog DeploymentComponent("smscomp", CPU(20));
    DC telcomp = new cog DeploymentComponent("telcomp", CPU(20));

    [DC: smscomp] SMSServer sms = new cog SMSServer();
    [DC: telcomp] TelephoneServer tel = new cog TelephoneServer();
    [DC: smscomp] Balancer smsb = new cog Balancer();
    [DC: telcomp] Balancer telb = new cog Balancer();
    smsb!setPartner(telb);
    telb!setPartner(smsb);
    c = new cog Handset(1,tel,sms); ... // Clients handsets
}
```

Simply by commenting out the Balancer, we get the same functional behavior model without load balancing.
Example 1: Phone Services - Simulation Results

---

Graphs showing the comparison between SMS load and available SMS with and without load balancing. The graphs illustrate the impact of load balancing on SMS availability during peak times.
Outline

- Motivating example
- Research goal and research questions
- Summary and future work
Resource-Aware Applications

Applications that can **administrate their resource usage** (e.g., to **automatically adapt to changes in client traffic**).
**Research Goal**

*Develop a methodology to express and compare different deployment decisions for resource-aware distributed applications during the design phase.*

RQ1: Model deployment architectures  
RQ2: Express deployment decisions  
RQ3: Estimate and quantify QoS  
RQ4: Compare deployment architectures  

Model distributed applications that can adapt to changes in e.g., client traffic
RQ1: How can we model deployment architectures for resource-aware distributed applications?

Deployment components to model physical or virtual deployment infrastructures
RQ2: How can we express deployment decisions in models of resource-aware distributed applications?

Design decisions related to QoS

1. User-defined schedulers
2. Resource management primitives
3. Object migration

Modeling of load-balancing strategies to improve performance
RQ3: How can we estimate and quantify QoS in models of resource-aware distributed applications?

1. Time
2. Deployment components
3. Deadlines to method calls
4. Cost Annotations
RQ4: How can we compare deployment architectures in models of resource-aware distributed applications?

How performance of a model improves with resource-awareness.
RQ4: How can we compare deployment architectures in models of resource-aware distributed applications?

How we can compare execution time and execution cost.

Partly ordered workflow and highly parallelizable tasks.

How we can project images

Background modeling
Calculations for smoothing out the background levels between overlapping images

Background matching
Smooth out the original background between overlapping images

Final mosaic
Co-add all the reprojected, smoothed images into a final mosaic

Re-project images
Redistribution of the images needed to create the mosaic into a common coordinate system
Outline

- Motivating example
- Research goal and research questions
- **Summary and future work**
Summary

- **Syntactic and semantic extensions of Core ABS**
  - to express resource-awareness and compare deployment decisions:
    - Real-Time ABS: explicit and implicit modeling of time
    - Deployment components: modeling of deployment architectures
    - User-defined cost annotations
    - Deadlines to method calls
    - Resource management primitives
    - User-defined schedulers

- About the proposed methodology (validated with examples and case studies), the approach of this thesis allows:
  - to observe how the same functional model can behave under different deployment choices
  - to take better design decisions for resource-aware distributed applications during the design phase
  - to acquire better understanding of the trade-offs from the different deployment choices
  - to predict performance, scalability and resource costs
Papers in this Thesis

1. **User-defined Schedulers for Real-time Concurrent Objects.**
   Joakim Bjørk, Frank S. de Boer, Einar Broch Johnsen, Rudolf Schlatte and Silvia Lizeth Tapia Tarifa.

2. **Integrating Deployment Architectures and Resource Consumption in Timed Object-Oriented Models.**
   Einar Broch Johnsen, Rudolf Schlatte and Silvia Lizeth Tapia Tarifa.
   **Publication:** Research Report 438, Department of Informatics, University of Oslo, February 2014. Journal of Logic and Algebraic Programming, accepted.

3. **Modeling Resource-Aware Virtualized Applications for the Cloud in Real-Time ABS.**
   Einar Broch Johnsen, Rudolf Schlatte and Silvia Lizeth Tapia Tarifa.

4. **Simulating Concurrent Behaviors with Worst-Case Cost Bounds.**
   Elvira Albert, Samir Genaim, Miguel Gómez-Zamalloa, Einar Broch Johnsen, Rudolf Schlatte and Silvia Lizeth Tapia Tarifa.
   **Publication:** Proceedings of the 17th International Symposium on Formal Methods, FM 2011.
Future Work

- **Resources:**
  - Other kind of resources (e.g., bandwidth, electric power)
  - User-defined resources
  - Multiple kinds of resources in the same deployment architecture

- **Stronger analysis techniques**, such as deductive verification

- Guarantee of **contractual obligations** related to QoS (e.g., integration of SLAs)

- **Methodology extensions** (e.g., stepwise development, code generation, model extraction, etc.)

- Model and analyze **other non-functional requirements** related to deployment (e.g., security, fault-tolerance)
<table>
<thead>
<tr>
<th>Motivation</th>
<th>Motivating Example</th>
<th>Goal and Research Questions</th>
<th>Summary</th>
<th>Papers</th>
<th>Future Work</th>
</tr>
</thead>
</table>

THANK YOU