Design of Self-Adaptive Monitoring Systems  
Ensuring Quality of Measurements:  
A Novel Approach

Antoine Toueir, Julien Broisin, Michelle Sibilla

University Toulouse III - IRIT (Institut de Recherche en Informatique de Toulouse)

October 14, 2013
Agenda

1. Introduction
   - Context
   - State-of-the-Art
   - Monitoring Issues

2. A Novel Approach: RE for Managing Monitoring
   - Adaptable Monitoring
   - Requirements Engineering as Adaptation Methodology

3. Use Case Implementation
   - Description
   - Demonstration

4. Conclusion
   - Added Value & Conclusion
Introduction
Context

Systems are conceived based on:

- **Functional Requirements (FR):**

- **Non-Functional Requirements (NFR):**
Systems are conceived based on:

- **Functional Requirements (FR):** Providing *Services*...

- **Non-Functional Requirements (NFR):**
Systems are conceived based on:

- **Functional Requirements (FR):** Providing *Services*...
  - System interaction with hosting environment.
  - System behavior (internal states).
- **Non-Functional Requirements (NFR):**
Systems are conceived based on:

- **Functional Requirements (FR):** Providing *Services*...
  - System interaction with hosting environment.
  - System behavior (internal states).
- **Non-Functional Requirements (NFR):** Providing *Quality*...
Systems are conceived based on:

- **Functional Requirements (FR):** Providing *Services*...
  - System interaction with hosting environment.
  - System behavior (internal states).

- **Non-Functional Requirements (NFR):** Providing *Quality*...
  - Domain-Independent attributes.
  - Domain-Dependent attributes.
Systems are conceived based on:

- **Functional Requirements (FR):** Providing **Services**...
  - System interaction with hosting environment.
  - System behavior (internal states).

- **Non-Functional Requirements (NFR):** Providing **Quality**...
  - Domain-Independent attributes.
  - Domain-Dependent attributes.

Usually, **QoS-Oriented Systems** are built as **Autonomous Systems** implementing the MAPE Loop (Monitoring, Analyzing, Planning, Executing).
What About Dynamicity in Autonomous Systems!!!

**Dynamicity** is a key issue in **Autonomous Systems**:
What About Dynamicity in Autonomous Systems!!!

Dynamicity is a key issue in Autonomous Systems:

- Functional Requirements:

- Non-Functional Requirements:
What About Dynamicity in Autonomous Systems!!!

Dynamics is a key issue in Autonomous Systems:

- **Functional Requirements:** Functional System states evolve during runtime

- **Non-Functional Requirements:**

```

```

```

```
Dynamicity is a key issue in Autonomous Systems:

- **Functional Requirements:** Functional System states evolve during runtime
  - → Providing new measurements.

- **Non-Functional Requirements:**
What About Dynamicity in Autonomous Systems!!!

**Dynamicity** is a key issue in **Autonomous Systems**:

- **Functional Requirements**: Functional System states evolve during runtime
  - → Providing new measurements.

- **Non-Functional Requirements**: Management decisions are strongly related to the agreed QoS & the infrastructure performance
What About Dynamicity in Autonomous Systems!!!

**Dynamicity** is a key issue in **Autonomous Systems**:

- **Functional Requirements**: Functional System states evolve during runtime
  - → Providing new *measurements*.

- **Non-Functional Requirements**: Management decisions are strongly related to the agreed QoS & the infrastructure performance
  - → Recognizing performance by providing *measurements*.
What About Dynamicity in Autonomous Systems!!!

**Dynamicity** is a key issue in **Autonomous Systems**:

- **Functional Requirements**: Functional System states evolve during runtime
  - → Providing new *measurements*.

- **Non-Functional Requirements**: Management decisions are strongly related to the agreed QoS & the infrastructure performance
  - → Recognizing performance by providing *measurements*.
What About Dynamicity in Autonomous Systems!!!

Dynamicity is a key issue in Autonomous Systems:

- **Functional Requirements:** Functional System states evolve during runtime
  - → Providing new measurements.

- **Non-Functional Requirements:** Management decisions are strongly related to the agreed QoS & the infrastructure performance
  - → Recognizing performance by providing measurements.

Preliminary Assertion

Monitoring System plays an important role → Thinking about QoM (Quality of Measurements)
State-of-the-Art

Criteria & Evaluation

Criteria...

- Monitoring Reconfiguration: changing behavior & architecture.
- Expressive Monitoring Adaptation: some sort of "semantic" behind adaptation.
Criteria & Evaluation

**Criteria...**

- Monitoring Reconfiguration: changing behavior & architecture.
- Expressive Monitoring Adaptation: some sort of "semantic" behind adaptation.

**Evaluation...**

- In most cases, the Functional System is directly concerned about reconfiguration.
- Lack of clear motivations for monitoring adaptation.
- No idea about the monitoring execution.
Criteria & Evaluation

Criteria...

- Monitoring Reconfiguration: changing behavior & architecture.
- Expressive Monitoring Adaptation: some sort of "semantic" behind adaptation.

Evaluation...

- In most cases, the Functional System is directly concerned about reconfiguration.
- Lack of clear motivations for monitoring adaptation.
- No idea about the monitoring execution.

In one phrase...

It is not sufficient to merely configure monitoring and its adaptation, starting from QoS specifications.
Our Aim...

Designing a **Self-Managed** Monitoring System...
Our Aim...

Designing a **Self-Managed** Monitoring System...
Our Aim...

Designing a Self-Managed Monitoring System...

1. **Adaptive**: With objective of
   - Responding to the *evolving* needs of the Functional & Management Systems.

![Diagram of System States and Relations]
Our Aim...

Designing a **Self-Managed** Monitoring System...

**Adaptive:** With objective of
- Responding to the **evolving** needs of the Functional & Management Systems.
- **Increasing** the quality of measurements.
Our Aim...

Designing a Self-Managed Monitoring System...

1. **Adaptive**: With objective of
   - Responding to the evolving needs of the Functional & Management Systems.
   - Increasing the quality of measurements.

2. **Integrated**: Hiding heterogeneity of management protocols & resources.
Our Aim...

Designing a **Self-Managed** Monitoring System...

1. **Adaptive**: With objective of
   - Responding to the *evolving* needs of the Functional & Management Systems.
   - Increasing the quality of measurements.

2. **Integrated**: Hiding heterogeneity of management protocols & resources.

3. **Generic**: Being independent of both the monitored Functional System & Management Systems.
Our Aim...

Designing a **Self-Managed** Monitoring System...

1. **Adaptive**: With objective of
   - Responding to the **evolving** needs of the Functional & Management Systems.
   - **Increasing** the quality of measurements.

2. **Integrated**: Hiding heterogeneity of management protocols & resources.

3. **Generic**: Being **independent** of both the monitored Functional System & Management Systems.

Our approach...

Leveraging on a **methodology** for twisting Functional, Management, & Monitoring expertise.
A Novel Approach: RE for Managing Monitoring
Adaptable Monitoring

Adaptable Monitoring Framework

Q: Where to start?
Adaptable Monitoring

Q: Where to start?
Q: Where to start?
Adaptable Monitoring

Adaptable Monitoring Framework

Q: Where to start?
  - Model-Driven Framework.
Adaptable Monitoring Framework

Q: Where to start?
  - Model-Driven Framework.
Q: Where to start?
   - Model-Driven Framework.
Adaptable Monitoring

Q: Where to start?
   Model-Driven Framework.
Q: Where to start?
  
  - Model-Driven Framework.
  - Distributed Architecture.
Q: Where to start?
- Model-Driven Framework.
- Distributed Architecture.
- Variety of implementations.
Q: Where to start?
- Model-Driven Framework.
- Distributed Architecture.
- Variety of implementations.

Q: How to express policies related to both FR & NFR?
A: Using a methodology for specifying requirements...
Adaptable Monitoring Framework

Q: Where to start?
  - Model-Driven Framework.
  - Distributed Architecture.
  - Variety of implementations.

Q: How to express policies related to both FR & NFR?
A: Using a methodology for specifying requirements...
Q: **Requirements Engineering??**
A: RE is a methodology for building systems solving real-world problems. It is about *"eliciting, evaluating, documenting, consolidating and changing the:*

- objectives,
- functionalities,
- assumptions qualities, and
- constraints...

... that the system-to-be should meet”\(^1\).

---

Q: **Requirements Engineering???
A: RE is a *methodology* for building systems solving real-world problems. It is about "eliciting, evaluating, documenting, consolidating and changing the:

- objectives,
- functionalities,
- assumptions qualities, and constraints...

... that the system-to-be should meet”\(^1\).

Q: **What is the adopted method?**
A: **KAOS.**

Q: *Requirements Engineering??*
A: RE is a methodology for building systems solving real-world problems. It is about "eliciting, evaluating, documenting, consolidating and changing the:

- objectives,
- functionalities,
- assumptions qualities, and
- constraints...

... that the system-to-be should meet"\(^1\).

Q: What is the adopted method?
A: KAOS.

Q: *Requirements Engineering***???
A: RE is a *methodology* for building systems solving real-world problems. It is about "elicit[ing], evalu[ating], document[ing], consoli[dating] and chang[ing the]:

- objectives,
- functionalities,
- assumptions qualities, and
- constraints...

... that the system-to-be should meet"¹.

Q: What is the adopted method?
A: KAOS.

Use Case Implementation
About Monitoring VMs in IaaS Provider

- **Objectives:** Continuous delivering of VMs measurements
About Monitoring VMs in IaaS Provider

**Objectives:** Continuous delivering of VMs measurements *with "some" respect* to the promised freshness in SLAs, by using the *minimum resources* for Monitoring.
About Monitoring VMs in IaaS Provider

- **Objectives:** Continuous delivering of VMs measurements *with "some" respect* to the promised freshness in SLAs, by using the minimum resources for Monitoring.

- **Problem:** ↑ VMs → ↑ Freshness violations.
Objectives: Continuous delivering of VMs measurements with "some" respect to the promised freshness in SLAs, by using the minimum resources for Monitoring.

Problem: ↑ VMs → ↑ Freshness violations.

Goals Elicitation:
About Monitoring VMs in IaaS Provider

- **Objectives**: Continuous delivering of VMs measurements with "some" respect to the promised freshness in SLAs, by using the minimum resources for Monitoring.

- **Problem**: ↑ VMs → ↑ Freshness violations.

- **Goals Elicitation**:
  - If freshness violations > *Max_Threshold* → Shrink monitoring perimeter...
About Monitoring VMs in IaaS Provider

- **Objectives:** Continuous delivering of VMs measurements with "some" respect to the promised freshness in SLAs, by using the minimum resources for Monitoring.

- **Problem:** \(\uparrow\) VMs \(\rightarrow\) \(\uparrow\) Freshness violations.

- **Goals Elicitation:**
  - If freshness violations > Max\_Threshold \(\rightarrow\) Shrink monitoring perimeter...
  - If monitored resources < Min\_Threshold \(\rightarrow\) Expand "someone else" monitoring perimeter...
Scenario Demonstration
Scenario Demonstration
Scenario Demonstration
Scenario Demonstration
Scenario Demonstration
Scenario Demonstration

Legend

- Model
- Class
- Instance
- VM
- **PM** Primary Manager
- **AM** Assistant Manager
- Configurability
- Adaptability
- Governance

Functional System

Adaptive Monitoring System
Scenario Demonstration

- **Functional System**
  - IaaS Client
  - Startup/Shutdown VM
  - Managing VMs

- **Adaptive Monitoring System**
  - Physical Host
  - Knowledge Base
  - PM: Primary Manager
  - AM: Assistant Manager

- **Legend**
  - Model
  - Class
  - Instance
  - VM

- **Use Case Implementation**
  - A Novel Approach: RE for Managing Monitoring

- **Conclusion**
  - Use Case Implementation
  - A Novel Approach: RE for Managing Monitoring
Scenario Demonstration

- Physical Host
- Physical Host
- Physical Host
- Expanding Agent
  - PM+AM Perimeter Size
- Knowledge Base
  - Provider
  - Configurability
  - Adaptability
  - Governance

Legend:
- Model
- Class
- Instance
- VM
- PM Primary Manager
- AM Assistant Manager

Functional System

Adaptive Monitoring System

- iaaS Client
- Startup/Shutdown VM
- Managing VMs
- SLA

Use Case Implementation

Conclusion
Scenario Demonstration

Functional System

Adaptive Monitoring System

Legend
- Model
- Class
- Instance
- VM
- PM Primary Manager
- AM Assistant Manager
- Configurability
- Adaptability
- Governance

IaaS Client

Startup/Shutdown VM

Managing VMs

Physical Host

Physical Host

Physical Host

Physical Host

Expanding Agent PM+AM Perimeter Size

Knowledge Base

SLA

Shrinking Agent Freshness Violation

PM Provider

AM Provider

Managing VMs
Scenario Demonstration
Scenario Demonstration

Legend
- Model
- Class
- Instance
- VM
- PM Primary Manager
- AM Assistant Manager
- Configurability
- Adaptability
- Governance

Functional System

Adaptive Monitoring System

Introduction
A Novel Approach: RE for Managing Monitoring
Use Case Implementation
Conclusion

Demonstration

Scenario Demonstration

- iaaS Client
  - Startup/Shutdown VM
  - Managing VMs

- Physical Host

- Expansion Agent
  - PM+AM Perimeter Size
  - Knowledge Base
  - Provider

- SLA

- AM Assistant Manager
  - Knowledge Base
  - Provider
  - Configurability
  - Adaptability
  - Governance
Conclusion
About Shrinking/Expanding... What to do in other scenarios....

Exploiting RE methodology to instrument **Governance Level**.

- **Determining the adaptation triggers**: Instantiating metrics & constraints instances (→ instrumenting & evaluating).
- **Determining the adaptation policy**: Instantiating adaptation policy (→ ”Subscribing” to constraints violations).
What to keep in mind...

About Shrinking/Expanding... What to do in other scenarios....

Exploiting RE methodology to instrument Governance Level.

- **Determining the adaptation triggers**: Instantiating metrics & constraints instances (→ instrumenting & evaluating).
- **Determining the adaptation policy**: Instantiating adaptation policy (→ ”Subscribing” to constraints violations).

The advantages of this approach....

- Proposing reusable models.
- Minimizing development cost.
- Being independent of the technological platforms.
- Providing Monitoring ”as a Service” for Functional & Management Systems.
Questions

Thanks for your attention...