

Goal-Oriented Monitoring Adaptation: Methodology and Patterns

Antoine Toueir, Julien Broisin, Michelle Sibilla
IRIT, Université Paul Sabatier
Toulouse, France
Email: {toueir,broisin,sibilla}@irit.fr

July 3, 2014



Agenda

- 1 Introduction
 - Context & Problematic
 - Related-Work
- 2 Cornerstone Framework
 - Configurability, Adaptability, Governability (CAG) Framework
- 3 Goal-Oriented Methodology for Adaptive Monitoring
 - Requirements Engineering (RE)
 - Base Patterns
 - Reconfiguration Dimensions
- 4 Adaptive Monitoring Patterns
 - Exchange Dimension
 - Metric Dimension
 - Spatial Dimension
 - Temporal Dimension
- 5 Conclusion
 - Wrap up & Questions

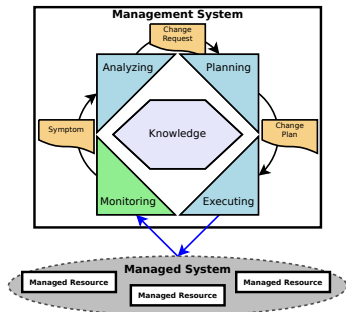
Introduction

Context

- What is network **management**?
" *The activities, methods, procedures, and tools that pertain to the operation, administration, maintenance, and provisioning of networked systems*" [1].

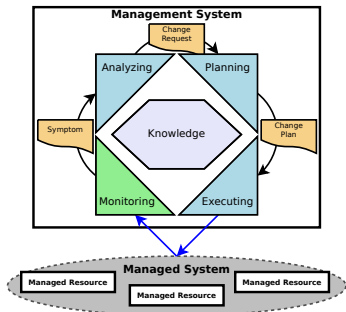
Context

- What is network **management**?
"The activities, methods, procedures, and tools that pertain to the operation, administration, maintenance, and provisioning of networked systems" [1].
- What about delegating **management** to the system **itself**?
 ⇒ **Autonomic Systems...**
 - **Properties:** Self-Configuration, Self-Optimization, Self-Healing & Self-Protection.



Context

- What is network **management**?
"The activities, methods, procedures, and tools that pertain to the operation, administration, maintenance, and provisioning of networked systems" [1].
- What about delegating **management** to the system **itself**?
 ⇒ **Autonomic Systems...**
 - **Properties:** Self-Configuration, Self-Optimization, Self-Healing & Self-Protection.
 - **MAPE-K Loop:** **Monitoring**, Analyzing, Planning, Executing & Knowledge.



Introductory questions !!!

Q: How to monitor?

A1 By using **instrumentation** operations (e.g., gathering, measuring, calculating).

A2 By applying **treatment** operations (e.g., filtering, correlating, aggregating).

A3 Ideally, **instrumentation & treatment** are configured starting from quality specifications (e.g., SLA).

Introductory questions !!!

Q: How to monitor?

- A1 By using **instrumentation** operations (e.g., gathering, measuring, calculating).
- A2 By applying **treatment** operations (e.g., filtering, correlating, aggregating).
- A3 Ideally, **instrumentation & treatment** are configured starting from quality specifications (e.g., SLA).

Q: Why to adapt monitoring?

- A1 Following the **state evolution** of sub-systems supporting *Functional Requirements* (services) & *Non-Functional Requirements* (qualities).
- A2 **Increasing** Quality of Service & Quality of Information.

Introductory questions !!!

Q: How to monitor?

- A1 By using **instrumentation** operations (e.g., gathering, measuring, calculating).
- A2 By applying **treatment** operations (e.g., filtering, correlating, aggregating).
- A3 Ideally, **instrumentation & treatment** are configured starting from quality specifications (e.g., SLA).

Q: Why to adapt monitoring?

- A1 Following the **state evolution** of sub-systems supporting *Functional Requirements* (services) & *Non-Functional Requirements* (qualities).
- A2 **Increasing** Quality of Service & Quality of Information.

Q: How to adapt monitoring?

- A1 Adaptation actions adjust the configuration of **instrumentation & treatment** operations.
- A2 There exist *ad-hoc* approaches, but no "**ready-to use recipe**"

Related-Work & Evaluation

The following statements were made based on [2, 3, 4, 5, 6, 7, 8, 9] :

- **Sometimes**, adaptation addresses the functional system but not the monitoring itself.
- When monitoring **is adapted**:
 - Scaling up/down metrics and managed resources.
 - (Un)-Deploying monitoring resources (e.g., managers, agents).
 - Modifying temporal parameters.
- Patterns approaches:
 - Dealing with the monitoring **deployment** and not **behavior**.

Related-Work & Evaluation

The following statements were made based on [2, 3, 4, 5, 6, 7, 8, 9] :

- **Sometimes**, adaptation addresses the functional system but not the monitoring itself.
- When monitoring **is adapted**:
 - Scaling up/down metrics and managed resources.
 - (Un)-Deploying monitoring resources (e.g., managers, agents).
 - Modifying temporal parameters.
- Patterns approaches:
 - Dealing with the monitoring **deployment** and not **behavior**.

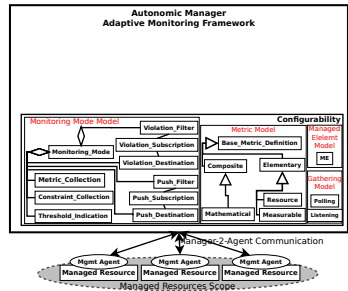
Limitations:

- "**Limited**" management systems.
- Monitoring adaptations **are not applicable** in other contexts.
- **No/Poor collaboration** among monitoring entities on the monitoring system whole scale.
- **No/Poor awareness** of monitoring system problems.

Cornerstone Framework

Zoom in !!!

- *Configurability Layer*:
 - Based on the *Common Information Model (CIM)*.
 - Modelling the managed elements, basic gathering mechanisms (*i.e.*, polling & listening), [Metrics, constraints & subscriptions].



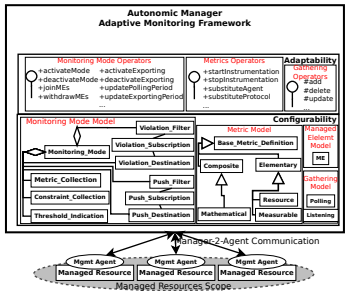
Zoom in !!!

● *Configurability Layer:*

- Based on the *Common Information Model (CIM)*.
- Modelling the managed elements, basic gathering mechanisms (*i.e.*, polling & listening), [Metrics, constraints & subscriptions].

● *Adaptability Layer:*

- Gathering-related (add, delete, etc.).
- Metric-related (startInstrumentation, stopInstrumentation, etc.).
- Constraint-related (add, delete, etc.).
- Mode-related (activateMode, deactivateMode, etc.).



Zoom in !!!

● *Configurability Layer:*

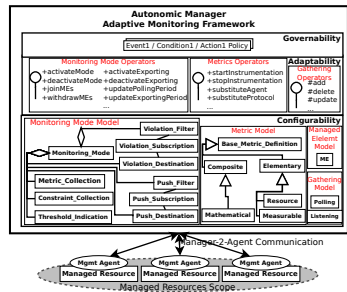
- Based on the *Common Information Model (CIM)*.
- Modelling the managed elements, basic gathering mechanisms (*i.e.*, polling & listening), [Metrics, constraints & subscriptions].

● *Adaptability Layer:*

- Gathering-related (add, delete, etc.).
- Metric-related (startInstrumentation, stopInstrumentation, etc.).
- Constraint-related (add, delete, etc.).
- Mode-related (activateMode, deactivateMode, etc.).

● *Governability Layer:*

Event/Condition/Action **Policies**.



Zoom in !!!

● *Configurability Layer:*

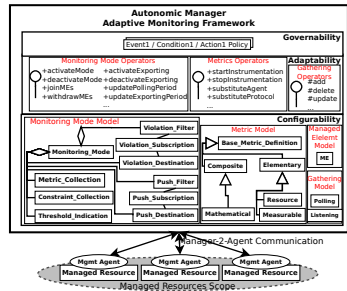
- Based on the *Common Information Model (CIM)*.
- Modelling the managed elements, basic gathering mechanisms (*i.e.*, polling & listening), [Metrics, constraints & subscriptions].

● *Adaptability Layer:*

- Gathering-related (add, delete, etc.).
- Metric-related (startInstrumentation, stopInstrumentation, etc.).
- Constraint-related (add, delete, etc.).
- Mode-related (activateMode, deactivateMode, etc.).

● *Governability Layer:*

Event/Condition/Action **Policies**.

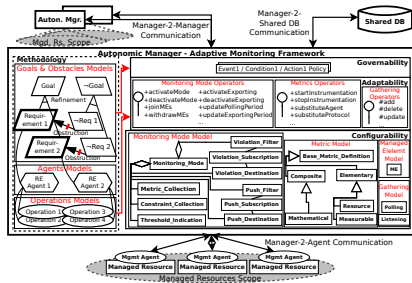
**IMPORTANT !!!**

Manipulating instances ⇒
Reconfiguring monitoring...

Goal-Oriented Methodology for Adaptive Monitoring

What & Why RE ???

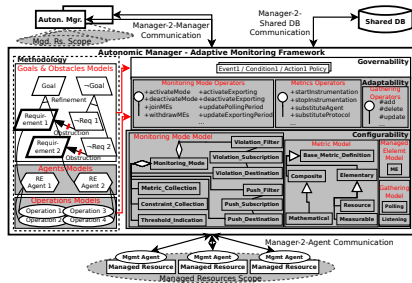
- RE iterates activities of "eliciting, evaluating, documenting, consolidating and changing the objectives, functionalities, assumptions qualities and constraints that the system-to-be should meet..." [10].
- KAOS¹ models the system-2-be through *Goal, Agent, Operation, Behavior & Object* Models.



¹Keep All Objectives Satisfied.

What & Why RE ???

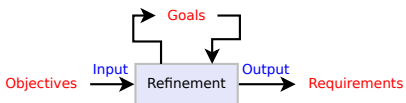
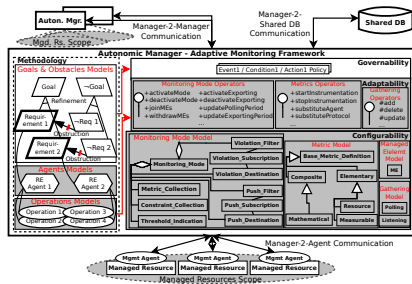
- RE iterates activities of "eliciting, evaluating, documenting, consolidating and changing the objectives, functionalities, assumptions qualities and constraints that the system-to-be should meet..." [10].
- KAOS¹ models the system-2-be through *Goal, Agent, Operation, Behavior & Object* Models.
- How to obtain the **ECA policies** to be inserted in the *Governability Layer*?



¹Keep All Objectives Satisfied.

What & Why RE ???

- RE iterates activities of "eliciting, evaluating, documenting, consolidating and changing the objectives, functionalities, assumptions qualities and constraints that the system-to-be should meet..." [10].
- KAOS¹ models the system-to-be through *Goal, Agent, Operation, Behavior & Object* Models.
- How to obtain the **ECA policies** to be inserted in the *Governability Layer*?



¹Keep All Objectives Satisfied.

How to refine goals ???

- **The main idea:** to assist refinement by using **correct & complete** formal patterns that are already checked, while **hiding mathematics**.
 - Patterns: **Milestone, Case & Guard.**

How to refine goals ???

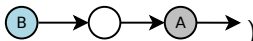
- **The main idea:** to assist refinement by using **correct & complete** formal patterns that are already checked, while **hiding mathematics**.
 - Patterns: **Milestone, Case & Guard**.
- **Formal language:** *Linear Temporal Logic* for goals modeling.
 - \diamond : *some time in the future* ($\diamond A$ )

Table: Patterns Refining ($P \Rightarrow \diamond Q$) Goal [11]

Pattern	Subgoal 1	Subgoal 2	Subgoal 3
---------	-----------	-----------	-----------

How to refine goals ???

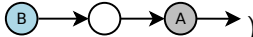
- **The main idea:** to assist refinement by using **correct & complete** formal patterns that are already checked, while **hiding mathematics**.
 - Patterns: **Milestone, Case & Guard.**
- **Formal language:** *Linear Temporal Logic* for goals modeling.
 - \diamond : *some time in the future* ($\diamond A$ )

Table: Patterns Refining ($P \Rightarrow \diamond Q$) Goal [11]

Pattern	Subgoal 1	Subgoal 2	Subgoal 3
Milestone	$P \Rightarrow \diamond R$	$R \Rightarrow \diamond Q$	

How to refine goals ???

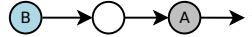
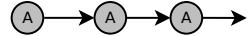
- **The main idea:** to assist refinement by using **correct & complete** formal patterns that are already checked, while **hiding mathematics**.
 - Patterns: **Milestone, Case & Guard**.
- **Formal language:** *Linear Temporal Logic* for goals modeling.
 - \diamond : *some time in the future* ($\diamond A$ )
 - \square : *always in the future* ($\square A$ )

Table: Patterns Refining ($P \Rightarrow \diamond Q$) Goal [11]

Pattern	Subgoal 1	Subgoal 2	Subgoal 3
Milestone	$P \Rightarrow \diamond R$	$R \Rightarrow \diamond Q$	
Case	$P \wedge P1 \Rightarrow \diamond Q1$	$P \wedge P2 \Rightarrow \diamond Q2$	$\square(P1 \vee P2)$ $Q1 \vee Q2 \Rightarrow Q$

How to refine goals ???

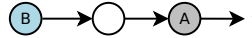
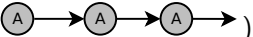
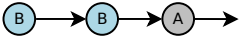
- **The main idea:** to assist refinement by using **correct & complete** formal patterns that are already checked, while **hiding mathematics**.
 - Patterns: **Milestone, Case & Guard**.
- **Formal language:** *Linear Temporal Logic* for goals modeling.
 - \diamond : **some time in the future** ($\diamond A$ )
 - \square : **always in the future** ($\square A$ )
 - \mathcal{W} : **always in the future unless** ($B \mathcal{W} A$ )

Table: Patterns Refining ($P \Rightarrow \diamond Q$) Goal [11]

Pattern	Subgoal 1	Subgoal 2	Subgoal 3
Milestone	$P \Rightarrow \diamond R$	$R \Rightarrow \diamond Q$	
Case	$P \wedge P1 \Rightarrow \diamond Q1$	$P \wedge P2 \Rightarrow \diamond Q2$	$\square(P1 \vee P2)$ $Q1 \vee Q2 \Rightarrow Q$
Guard	$P \wedge \neg R \Rightarrow \diamond R$	$P \wedge R \Rightarrow \diamond Q$	$P \Rightarrow P \mathcal{W} Q$

Idea and definition !!!

- **The main idea:** to **neutralize** patterns from any quality semantic.

Idea and definition !!!

- **The main idea:** to **neutralize** patterns from any quality semantic.
- **Dimension definition:** containers of "**consistent**" monitoring **parameters** that are subject to be **reconfigured**.

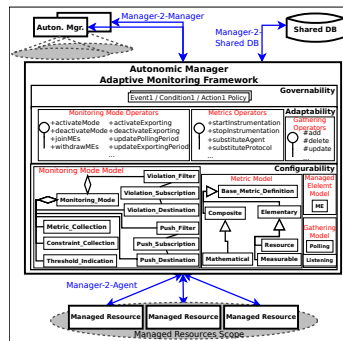
Idea and definition !!!

- **The main idea:** to **neutralize** patterns from any quality semantic.
- **Dimension definition:** containers of "**consistent**" monitoring **parameters** that are subject to be **reconfigured**.
- **Dimensions types:** based on *bottom-up* analysis, four types are identified [12].
 - Exchange.
 - Metric.
 - Spatial.
 - Temporal.

Adaptive Monitoring Patterns

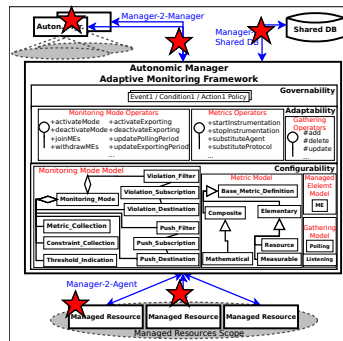
Exchange Dimension - Context

- Autonomic Managers (AMs) **collaborate** together ⇒ information exchange.
- Pattern deals with metrics **gathering/delivering** problems.



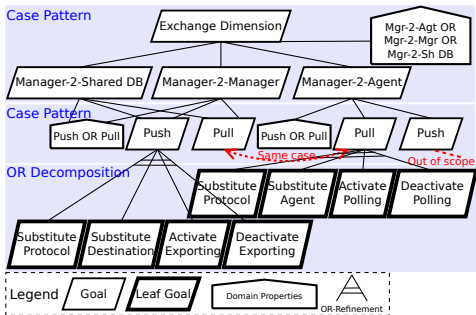
Exchange Dimension - Context

- Autonomic Managers (AMs) **collaborate** together ⇒ information exchange.
- Pattern deals with metrics **gathering/delivering** problems.
- Pattern could be applied for:
 - Increasing the quality of pulled/pushed metric values.
 - Querying more qualified agents.
 - Blocking fake agents.
 - Securing the communication between sources & destinations.



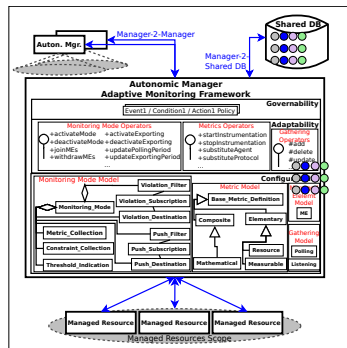
Exchange Dimension - Pattern

- **Dimension Specificities:**
 - Communication **classes:**
M-2-A, M-2-M,
M-2-Shared DB.
 - Communication **modes:**
Pull & Push.
 - Exchange triplet: *Source, Destination, Protocol.*
- **Requirements:**
 - (De)-Activate Polling
 - (De)-Activate Exporting
 - Substitute Protocol
 - Substitute Agent
 - Substitute Destination



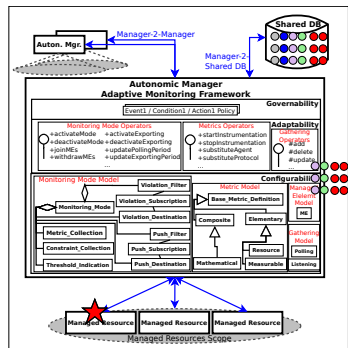
Metric Dimension - Context

- Monitoring must follow Fun. & Mgmt. systems \Rightarrow Monitoring **doesn't** instrument the same metrics all the time.
- Pattern controls the **trade-off** between constructing more knowledge and monitoring the **necessary** information.



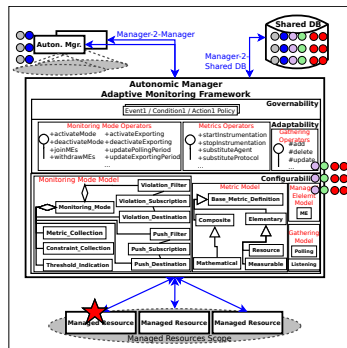
Metric Dimension - Context

- Monitoring must follow Fun. & Mgmt. systems ⇒ Monitoring **doesn't** instrument the same metrics all the time.
- Pattern controls the **trade-off** between constructing more knowledge and monitoring the **necessary** information.
- Pattern could be applied for:
 - Troubleshooting, root cause analysis.



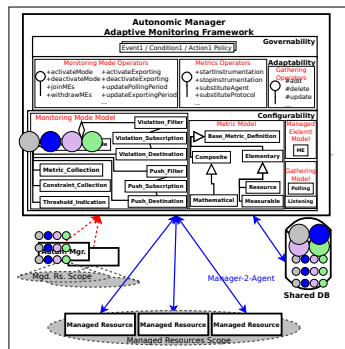
Metric Dimension - Context

- Monitoring must follow Fun. & Mgmt. systems ⇒ Monitoring **doesn't** instrument the same metrics all the time.
- Pattern controls the **trade-off** between constructing more knowledge and monitoring the **necessary** information.
- Pattern could be applied for:
 - Troubleshooting, root cause analysis.
 - "Engineering" the distribution of monitored metrics among AMs.



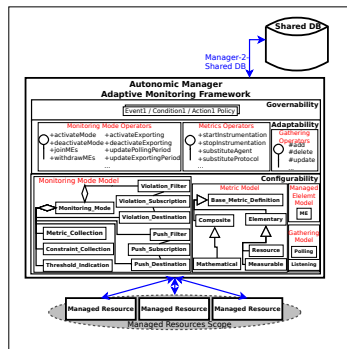
Metric Dimension - Context

- Monitoring must follow Fun. & Mgmt. systems \Rightarrow Monitoring **doesn't** instrument the same metrics all the time.
- Pattern controls the **trade-off** between constructing more knowledge and monitoring the **necessary** information.
- Pattern could be applied for:
 - Troubleshooting, root cause analysis.
 - "Engineering" the distribution of monitored metrics among AMs.
 - Modifying the hierarchical topology of AMs.



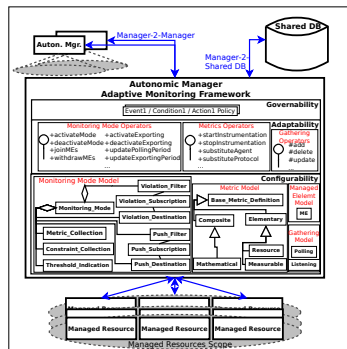
Spatial Dimension - Context

- Each Autonomic Manager (AM) has its own control perimeter.
- Pattern deals with important changes regarding the control perimeters; because a **mass of users** consuming system services could **oscillate continuously**.



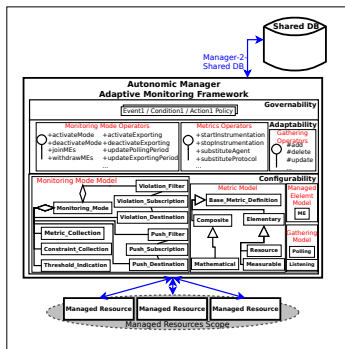
Spatial Dimension - Context

- Each Autonomic Manager (AM) has its own control perimeter.
- Pattern deals with important changes regarding the control perimeters; because a **mass of users** consuming system services could **oscillate continuously**.
- Pattern could be applied for:
 - Load balancing of monitoring overhead among AMs.
 - Supporting scalability.



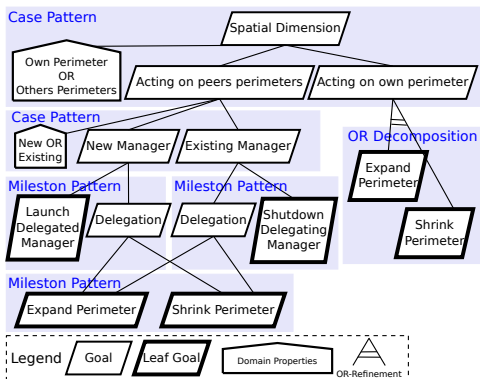
Spatial Dimension - Context

- Each Autonomic Manager (AM) has its own control perimeter.
- Pattern deals with important changes regarding the control perimeters; because a **mass of users** consuming system services could **oscillate continuously**.
- Pattern could be applied for:
 - Load balancing of monitoring overhead among AMs.
 - Supporting scalability.
 - Minimizing the cost of AMs management.



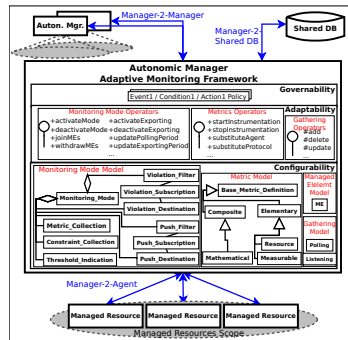
Spatial Dimension - Pattern

- **Dimension Specificities:**
 - **Local & Collaborative** treatment.
 - **New or Existing** managers.
- **Requirements:**
 - Expand perimeter.
 - Shrink perimeter.
 - Launch delegated manager.
 - Shutdown delegating manager.



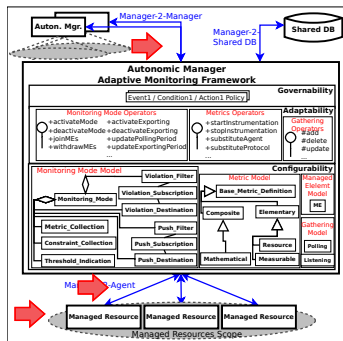
Temporal Dimension - Context

- Temporal aspects intervene in other Dimensions.
- Pattern deals with **temporal violations**, **scheduling** issues and tuning **metric analysis**.



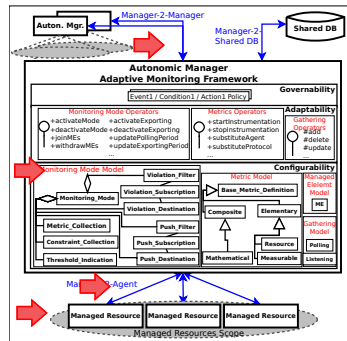
Temporal Dimension - Context

- Temporal aspects intervene in other Dimensions.
- Pattern deals with **temporal violations**, **scheduling** issues and tuning **metric analysis**.
- Pattern could be applied for:
 - Controlling (e.g., relaxing, stressing) the monitoring overhead on AMs, network paths & remote agents.



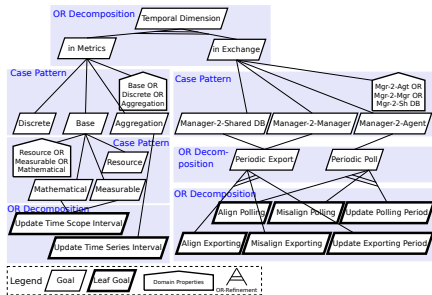
Temporal Dimension - Context

- Temporal aspects intervene in other Dimensions.
- Pattern deals with **temporal violations**, **scheduling** issues and tuning **metric analysis**.
- Pattern could be applied for:
 - Controlling (e.g., relaxing, stressing) the monitoring overhead on AMs, network paths & remote agents.
 - Tuning temporal parameters of metrics **aggregation & correlation**.



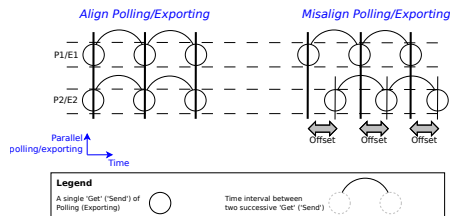
Temporal Dimension - Pattern

- Dimension Specificities:
 - Considering **individual & collectives** aspects.
- Requirements:
 - Update Time Series Period (i.)
 - Update Time Scope Period (i.)
 - Update Polling Period (i.)
 - Update Exporting Period (i.)
 - Align/Misalign Polling (c.)
 - Align/Misalign Exporting (c.)



Temporal Dimension - Pattern

- Dimension Specificities:
 - Considering **individual & collectives** aspects.
- Requirements:
 - Update Time Series Period (i.)
 - Update Time Scope Period (i.)
 - Update Polling Period (i.)
 - Update Exporting Period (i.)
 - Align/Misalign Polling (c.)
 - Align/Misalign Exporting (c.)



Conclusion

Wrap up!!!

- **Extending** the functionalities of the management system by virtue of integrating new quality objectives to be satisfied (**all patterns**).
- Each identified monitoring adaptation **Requirement** can satisfy **+1** objectives and they are **applicable in many** contexts (**all patterns**).
- **Collaborating** among AMs (**Metric & Spatial patterns**).
- Making the monitoring system **aware** about its faults (**all patrons**).

Questions

Thanks for your attention...

Wrap up & Questions

- [1] A. Clemm, *Network Management Fundamentals*. Cisco Press, 2006, ch. 5, pp. 138–141.
- [2] P. Grefen, K. Aberer, H. Ludwig, and Y. Hoffner, “Crossflow: Cross-organizational workflow management in dynamic virtual enterprises,” *International Journal of Computer Systems Science & Engineering*, vol. 15, 2000, pp. 277–290.
- [3] D. Roxburgh, D. Spaven, and C. Gallen, “Monitoring as an sla-oriented consumable service for saas assurance: A prototype,” in *2011 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2011, pp. 925–939.
- [4] P. Thongtra and F. Aagesen, “An adaptable capability monitoring system,” in *2010 Sixth International Conference on Networking and Services (ICNS)*, 2010, pp. 73–80.
- [5] M. Munawar, T. Reidemeister, M. Jiang, A. George, and P. Ward, “Adaptive monitoring with dynamic differential tracing-based diagnosis,” in *Managing Large-Scale Service Deployment*, ser. *Lecture Notes in Computer Science*, F. Turck, W. Kellerer, and G. Korkentzas, Eds. Springer Berlin Heidelberg, 2008, vol. 5273, pp. 162–175.
- [6] J. Nobre, L. Granville, A. Clemm, and A. Prieto, “Decentralized detection of sla violations using p2p technology,” in *Proceedings of the 8th International Conference on Network and Service Management*, 2012, pp. 100–107.
- [7] IBM Corp., “An architectural blueprint for autonomic computing,” *IBM White Paper*, June 2005.
- [8] D. Weyns, B. Schmerl, V. Grassi, S. Malek, R. Mirandola, C. Prehofer, J. Wuttke, J. Andersson, H. Giese, and K. Goschka, “On patterns for decentralized control in self-adaptive systems,” in *Software Engineering for Self-Adaptive Systems II*, R. Lemos, H. Giese, H. Muller, and M. Shaw, Eds. Springer Berlin Heidelberg, 2013, vol. 7475, pp. 76–107.
- [9] A. J. Ramirez and B. H. C. Cheng, “Design patterns for developing dynamically adaptive systems,” in *Proceedings of the 2010 ICSE Workshop on Software Engineering for Adaptive and Self-Managing Systems*, ser. *SEAMS '10*, 2010, pp. 49–58.
- [10] A. Van Lamsweerde, *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Wiley, 2009.

- [11] R. Darimont and A. Van Lamsweerde, "Formal refinement patterns for goal-driven requirements elaboration," in ACM SIGSOFT Software Engineering Notes, vol. 21, no. 6. ACM, 1996, pp. 179–190.
- [12] A. Toueir, J. Broisin, and M. Sibilla, "A goal-oriented approach for adaptive sla monitoring: a cloud provider case study," in LATINCLOUD 2013, Maceio, Brazil, December 2013.