A Software Architecture Framework for Quality-Aware DevOps

D. A. Tamburri, E. Di Nitto, M. Guerriero, P. Jamshidi, I. Spais
What are we up to, today?

- Architecture Frameworks, bits and pills
- Quality-Aware DevOps Concerns
- Quality-Aware DevOps Architecture Descriptions & Requirements
- What’s missing from the state of the art
  - Our research solution, SQUID!
  - SQUID implementation and usage in Data-Intensive Architectures (DIA)
The ISO/IEC/IEEE 42010 Conceptual Model of Architecture Description\cite{1} defines the term architecture framework as:

“a (set of) common practice(s) for creating, interpreting, analyzing and using architecture descriptions within a particular domain of application or stakeholder community”
The ISO/IEC/IEEE 42010 Conceptual Model of Architecture Description\[1\] defines the term architecture framework as:

“a (set of) **common practice(s) for** creating, interpreting, analyzing and using **architecture descriptions within a particular domain of application** or stakeholder community”
The ISO/IEC/IEEE 42010 Conceptual Model of Architecture Description\textsuperscript{[1]} defines the term architecture framework as:

```
“a (set of) \textbf{common practice(s)* for} creating, interpreting, analyzing and using \textbf{architecture* descriptions* within a particular domain of application* or stakeholder community”}
```

\* Continuously!
The ISO/IEC/IEEE 42010 Conceptual Model of Architecture Description\textsuperscript{[1]} defines the term architecture framework as:

"a (set of) \textit{common practice(s)}* for creating, interpreting, analyzing and using \textit{architecture* descriptions* within a particular domain of application* or stakeholder community}"

** With Quality!* * Continuously****!
Quality-Aware DevOps Concerns*

**Concerns**
- Automation
- Heterogeneous maturity (systems & orgs.)
- End-to-end Architecting
- QoS-, Business- and Technical-driven Continuous Architecting

**Practices**
- Trial-and-error!
- ...

*30+ Interviews and 7 Focus-Groups with industrial partners in the EU H2020 DICE and other industrials*
Quality-Aware DevOps Architecture Descriptions: Requirements!

1. Fine-grained architecture descriptions specific to DevOps frameworks and middleware;

1. Architecture blueprints with infrastructure, platform and application topology specs;

2. Model-based synch of all of the above;
Comparative evaluation of previous well-known/established arch. Frameworks:

- MODAF
- RM-ODP
- DODAF
- TOGAF
- 4+1-Views
- ...

Systematic mapping of previous architecture frameworks[2]
Comparative evaluation of previous well-known/established arch. Frameworks:

- MODAF
- RM-ODP
- DODAF
- TOGAF
- 4+1-Views
- ...

Systematic mapping of previous architecture frameworks\[^2\]
Quality-Aware DevOps Architecture Descriptions

The 4+1 Views\textsuperscript{[3]} model offers a basis consistent with the identified stakeholders and concerns for quality-aware DevOps...
Quality-Aware DevOps Architecture Descriptions

The 4+1 Views\(^3\) model offers a basis consistent with the identified stakeholders and concerns for quality-aware DevOps...

RQ1: Fine-grained architecture descriptions
Quality-Aware DevOps Architecture Descriptions

The 4+1 Views\(^3\) model offers a basis consistent with the identified stakeholders and concerns for quality-aware DevOps...

RQ1: Infrastructure, platform & app topology specs
The 4+1 Views\textsuperscript{[3]} model offers a basis consistent with the identified stakeholders and concerns for quality-aware DevOps...

RQ1: Model-based synching of arch. views
But...
Something is missing, for example...

1. The Logical architecture view needs sync with quality properties verification views.
2. The Dev-architecture view needs synch with the -Ops view (i.e., special Physical view)

* More on the paper
Summing it all up: SQUID Views and Transformations
Summing it all up: SQUID Views and Transformations

Dev Area Views (Development structure, business logic, quality verification)

Ops Area Views (Deployment, process, monitoring)
SQUID continuous architecting in action

1. Specify Architecture components and restrictions... UML profile!
SQUID continuous architecting in action

2. Transform architecture elements in techs. Using ad-hoc tech-packs
SQUID continuous architecting in action

3. Verify quality properties by means of ad-hoc annotations and tools
SQUID continuous architecting in action

4. Deploy improved architecture
SQUID continuous architecting in action

5. Monitor and continuously enhance/architect/(re-)-deploy
For example*...
In conclusion... Take-home messages!

- MDE and DevOps are made for each other
  - SQUID offers a complete approach for Quality-Aware MDE-based continuous architecting (e.g., of DIAs)
  - Heavy use of M2M and M2T transformations

- Needs in DevOps rotate around multi-view and continuous-architecting
  - Speedy modeling, synch and (re-)deployment are critical
Bibliography

