Networked Organizations Viewpoint for Architectures

Damian A. Tamburri
VU University Amsterdam
d.a.tamburri@vu.nl

Patricia Lago
VU University Amsterdam
p.lago@vu.nl

Christoph Dorn
Vienna University of Technology
Information Systems Institute
c.dorn@infosys.tuwien.ac.at

Rich Hilliard
consulting software systems architect
r.hilliard@computer.org

version 0.9, October 12, 2013

Abstract

Networked Organizations Viewpoint for Architectures (or NOVA) is intended to capture the influence of dynamic organizations upon a software architecture. It is described here using the viewpoint template defined on the ISO/IEC/IEEE 42010 website [3].

Overview

The purpose of the Networked Organizations Viewpoint for Architectures (or NOVA) is to capture the influence of dynamic organizations upon a software architecture.

Keywords: networked software engineering, software architecture, architecture knowledge, networked organizations, software architecture viewpoints

Framed concerns and typical stakeholders

Concerns

NOVA frames these specific concerns:

- responsibility: who is responsible for each architecture element?
- ownership: who owns each architecture element?
- usage: who uses each architecture element?
- collaboration: who is collaborating with whom? and how?

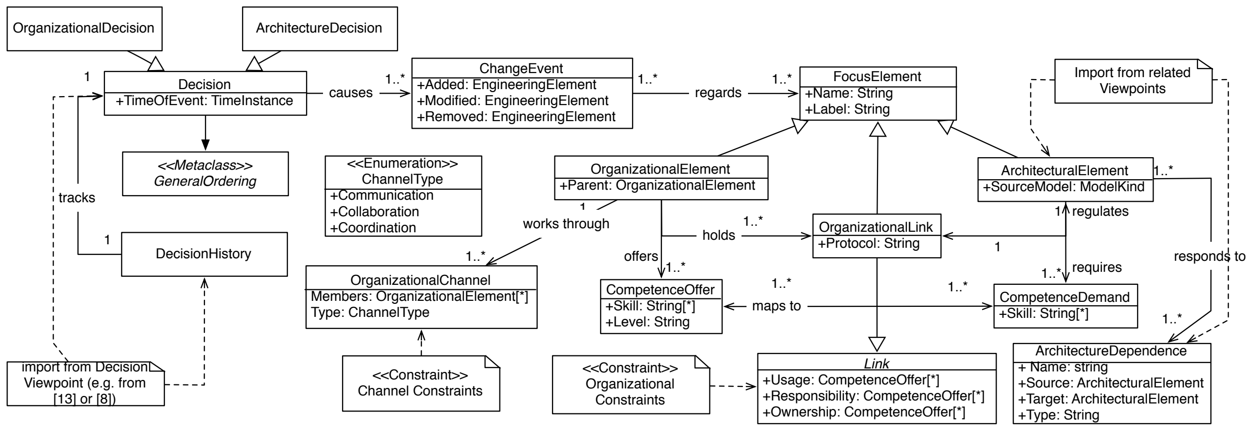
OrganizationalDecision
ArchitectureDecision
Decision
+TimeOfEvent: TimeInstance
causes
ChangeEvent
+Added: EngineeringElement
+Modified: EngineeringElement
+Removed: EngineeringElement
regards
FocusElement
+Name: String
+Label: String
Import from related Viewpoints
<<Metaclass>> GeneralOrdering
<<Enumeration>> ChannelType
+Communication
+Collaboration
+Coordination
OrganizationalElement
+Parent: OrganizationalElement
ArchitecturalElement
+SourceModel: ModelKind
tracks
DecisionHistory
works through
holds
OrganizationalLink
+Protocol: String
regulates
requires
responds to
offers
OrganizationalChannel
Members: OrganizationalElement[*]
Type: ChannelType
CompetenceOffer
+Skill: String[*]
+Level: String
maps to
CompetenceDemand
+Skill: String[*]
Import from Decision Viewpoint (e.g. from [13] or [8])
<<Constraint>> Channel Constraints
<<Constraint>> Organizational Constraints
Link
+Usage: CompetenceOffer[*]
+Responsibility: CompetenceOffer[*]
+Ownership: CompetenceOffer[*]
ArchitectureDependence
+ Name: string
+Source: ArchitecturalElement
+Target: ArchitecturalElement
+Type: String
1
1..*

Figure 1: NOVA meta-model.

Typical stakeholders

NOVA addresses the following typical stakeholders:

- owners of the system
- developers of the system
- builders of the system
- maintainers of the system

Model kinds+

A viewpoint may comprise one or more model kinds. Currently, only a single model kind is used for describing a NOVA architecture view: the NO model kind (described immediately below).

NO model kind

The NO model kind brings together elements from different existing models; specifically, it imports constructs for architecture decisions, architecture elements and their dependencies from existing viewpoints (or model kinds) and it is built upon hADL4nova [6] for modelling the organizations' collaboration aspects.

Figure 1 depicts the meta-model. Note that the NO meta-model and NOVA instance example below (see figure 2) are simplified depictions for sake of clarity and readability. Most elements are refined in the text of the following subsections.

Major entities and their attributes

All NO model elements exhibit the attributes *identifier* and *label*.

- *ArchitectureElement*s (AE) have attributes: *name*, *isCompleted* flag, *sourceModel*, *sourceModelElementId*. An AE is reified as one of: *ArchStructure*, *Component*, *Connector*, or *Interface* (from hADL4nova). The *isCompleted* flag indicates whether the element is an asset ready for use or still subject to manipulation.
- An *OrganizationElement* (OE) represents one of the Actor subtypes from the hADL4nova model: *OrgStructure*, *HumanComponent*, or *CollabConnector*.
- *OrganizationChannel* is implemented as (i.e., is identical to) a *CollabChannel* from the hADL4nova model.
- *Skill*s are first-class entities that exist independently of AEs or OEs. They become offered, respectively demanded, competencies once associated to OEs and AEs (see Relationships below).
- *OrganizationLink*s are first-class entities (implemented as *Mapping*s) for defining expressive relations between *OrganizationElement*s and *ArchitectureElement*s. NO distinguishes *Ownership*, *ProductionOf*, *AccessTo*, and *DependencyOn* mappings. These distinct m-to-n mapping types allow for fine-grained constraint checking as outlined in Constraints below.
- *Decision*s are meant to be imported from an independent decision viewpoint (or model kind) such as [2], [4]. These viewpoints also manage the decision history. hADL4nova, however, provides a mechanism to highlight a decision's immediate and cascading effects.
- *DecisionChange*s are sets that contain all hADL4nova model elements that were changed (added, deleted, updated) as an immediate result of a decision of from subsequent (cascading) changes.

Relationships

NO splits relationships into four coarse-grained groups: (i) architecture dependencies (for defining relations among architecture elements), (ii) organizational relations (for defining collaboration channels and constraints among organizational structures), (iii) skill associations (for defining skill demand and supply), and (iv) architecture–to–organization mappings (specifying input/output/task dependencies), the core contribution of this model kind.

Architecture dependencies are imported from other viewpoints (e.g., components–and-connectors, logical viewpoints). These dependencies are relevant here because they may propagate certain kinds of architecture-to-organization relations. For example, two linked architecture elements may demand that their respective responsible organization elements exhibit a joint collaboration channel.

$ArchitecturalDependence \subseteq ArchitectureElement \times ArchitectureElement$

In NO, an *ArchitectureDependence* has attributes: *name*, *type*, and *source* and *target*, which are themselves AEs. *ArchitectureDependenc*ies are instantiated either as a *Link* between two *Interface*s, or as a *Dependency* between two *ArchStructure* elements.

Organizational relations describe the collaboration channels and mechanisms available to the networked organizations. See [5] which outlines the semantics of linking organizations to collaboration channels in more detail.

$OrganizationRelation \subseteq OrganizationElement \times CollabChannel$

Skill associations specify the underlying skill demand and supply. An architecture element's development (e.g., implementation, integration, operation, maintenance) requires skills at a particular $level_{demand}(arch_i, skill_j)$ of mastery.

$CompetenceDemand \subseteq ArchitectureElement \times Skill$

In the same manner, an organization's competencies are determined by its skill and respective skill $level_{avail}(org_i, skill_j)$.

$CompetenceOffer \subseteq OrganizationElement \times Skill$

Architecture-to-Organization mappings connect the first-class *Mapping* elements to architecture elements on the one side, and organization elements on the other side.

The *Owning* relation identifies which organization is the ultimate authority over the identified architecture element(s) (via *Ownership* mapping subtype); independent of the architecture element's development status.

$Owning \subseteq OrganizationElement \times Ownership$

Complementary to the owning relationship, the *Accessing* relation identifies which *OrganizationChannel*s (e.g., an SVN) give access to what particular architecture elements (via the *AccessTo* mapping subtype); thereby placing the collaboration mechanisms and constraints among organizations into an architectural scope.

$Assigned \subseteq OrganizationChannel \times AccessTo$

The *Assigned* relations identifies all organizations that are involved in the implementation, integration, testing, or other actions affecting one or more architectural elements (via *ProductionOf* mapping subtype). The assigned organizations may get involved as *Responsible*, *Participant*, or *Consultant*. Note that the responsible organization may not necessarily be the owning organization.

$Assigned \subseteq OrganizationElement \times ProductionOf$

The *Depending* relation identifies organizations that require (i.e., use) particular architecture elements (via *DependencyOn* mapping subtype) for their assigned activities (e.g., implementation of a different architecture element).

$Depending \subseteq OrganizationElement \times DependencyOn$

In combination, *Assigned* and *Depending* relations define all of the input/output dependencies in the networked organization.

Ultimately, the *MapToArch* relations ground the various *Mapping* subtypes in the *ArchitectureElement*s.

$MapToArch \subseteq Mapping \times ArchitectureElement$

In summary, all relations between *ArchitectureElement*s and *OrganizationElement*s may be collected as follows:

$AE{\star}OE \subseteq OrganizationElement \times Mapping \times ArchitectureElement$

Constraints

NO distinguishes between hard and soft constraints.[1] *Hard constraints* are automatically enforced, i.e., the designer cannot complete a violating design decision. *Soft constraints* result in a warning.

Hard constraints. Currently, only a single hard constraint exists; aside from connecting model elements only according to the available relations and dependencies defined in the previous subsection:

H1 An *OrganizationLink* (*Mapping*) is always connected to at least one OE and one AE.

[1]Constraints expressing relations *between architecture elements* are considered correspondence rules in the terminology of ISO/IEC/IEEE 42010.

Soft constraints. At this stage, soft constraints focus on organizational skill application, skill demand-supply matches, and AE-to-OE mapping constraints.

S1 For every demanded skill (i.e., at least one *CompetenceDemand* relation), there should be a corresponding offered skill (i.e., *CompetenceOffer* relation).
Rationale: independent of the actual organization-to-architecture mappings, all required skills should be available in the networked organization.
Violation Effect: a constraint violation highlights that the set of collaborating organizations is unable to properly produce (or otherwise manipulate) the complete set of architecture elements.

S2 Every *ArchitectureElement* requires at least one *Ownership* mapping.
Rationale: Ownership clearly identifies the organization to contact on any coordinative matter on the architecture element. Joint ownership is possible but requires explicit management via dedicated *CollabChannel*s.
Violation Effect: an AE without a dedicated owner may be subject to conflicting changes from multiple organizations without the dedicated authority to resolve the conflict.

S3 Every *ArchitectureElement* that is not yet complete (i.e., not yet an asset) requires at least one *ProductionOf* mapping.
Rationale: Identifying explicitly the organizations involved in producing an architecture element enables reasoning about the resulting inter-organizational collaboration requirements.
Violation Effect: Multiple organizations may change the architecture element without notifying other participants of the effect, potentially even overwriting previous changes when mutual interest is not captured in *ProductionOf* mappings.

S4 Every *ProductionOf* mapping requires exactly one *AssignedResponsible* organization.
Rationale: Only one organization should coordinate the lifecycle of the connected architecture element(s) thereby serving as the sole responsible entity for successful manipulation.
Violation Effect: Having no responsible organization (possibly due to insufficient resources) incurs high risk of failure to produce the architecture element as planned. Involving too many responsible organizations induces risk of disagreement.

S5 Every *ProductionOf* mapping requires *Assigned* organizations that exhibit offered competencies as indicated by the architecture elements' required skills.
Rationale: While the networked organizations together may possess all of the required skills, the right organizations need to be involved in the production of an architecture element.
Violation Effect: Low quality or completely failing architecture elements will result from involving organizations that cannot offer all of the required skills.

S6 Every *OrganizationElement* should be involved in at least one *ProductionOf* mapping.
Rationale: Every involved organization should provide added value through taking responsibility for, participating in, or consulting during production of an architecture element. Exceptions are organizations providing purely management services.
Violation Effect: Lack of identifying organizations without clear involvement may result in wasted resources.

Correspondence rules

NOVA and its elements correspond to decision viewpoints as well as change-tracking and dissemination mechanisms for models (and model kinds).

Each *ArchitectureDependence* (type, really) used in NO is imported from (therefore corresponds to) another viewpoint specification and obeys any constraints from that source, in addition to the NOVA meta model above.

The hard and soft constraints above (see) are correspondence rules in the terms of the Standard.

Operations on views

NOVA is designed entirely within the Generic Modelling Environment (GME)[2]. As such, it enables architects to construct the model by dragging and dropping NOVA model elements onto a canvas, establishing relations among elements, and checking constraints all within the GME tool. At the current stage, no import capabilities are available for including architecture description elements from other views or models. Similarly, change propagation and tracking is ultimately out of the scope of the viewpoint, however, related work on model consistency and synching exists that delivers such functionalities [1]. Authoritative sets of *ArchitectureElement*s, *OrganizationElement*s, and *Skill*s are managed in *folders*. Individual canvases then host the actual NOVA view. Designers create references to these elements and subsequently link them according to the relations specified above.

On each canvas, NOVA provides two *aspects* (in GME terminology) to switch between *ViewElement*s (for creating relations among organizations, relations among architecture elements, and mappings between organization and architecture) and *SkillAssignment* (for specifying *CompetenceOffer* and *CompetenceDemand* relations). *OrganizationElement*s, *ArchitectureElement*s, and *Skill* elements, the common elements, are automatically synchronized upon switching between the two aspects.

Construction/analysis methods

The following steps provide assistance to the designer in resolving the soft-constraints outlined above.

C1 *CompetenceDemand* and *CompetenceOffer* mismatch: The *SkillAssignment* aspect in GME visualizes precisely for each violated skill which architecture elements demand this skill. Resolving activities may include (i) confirmation that the skill is actually required (potentially removing the *CompetenceDemand*, unlikely when many AE require this skill), (ii) search and identify previously unknown skills in the existing networked organizations (adding a *CompetenceOffer* link), or restructure the networked organization to include a suitable partner.

C2 *ArchitectureElement*–*Ownership* mapping: identify the most fitting organization element (e.g., the client, the lead organization), respectively, in the case of an completed AE, the organization that supplied the AE. Subsequently, create an Ownership mapping and connect it to the AE and OE.

C3 *ArchitectureElement*–*ProductionOf* mapping: a missing *ProductionOf* mapping might indicate insufficient resources among the participating organizations. Creating a *ProductionOf* mapping and assigning responsible organizations (either existing or new) resolves this constraint violation. The *SkillAssignment*

[2] http://w3.isis.vanderbilt.edu/Projects/gme/

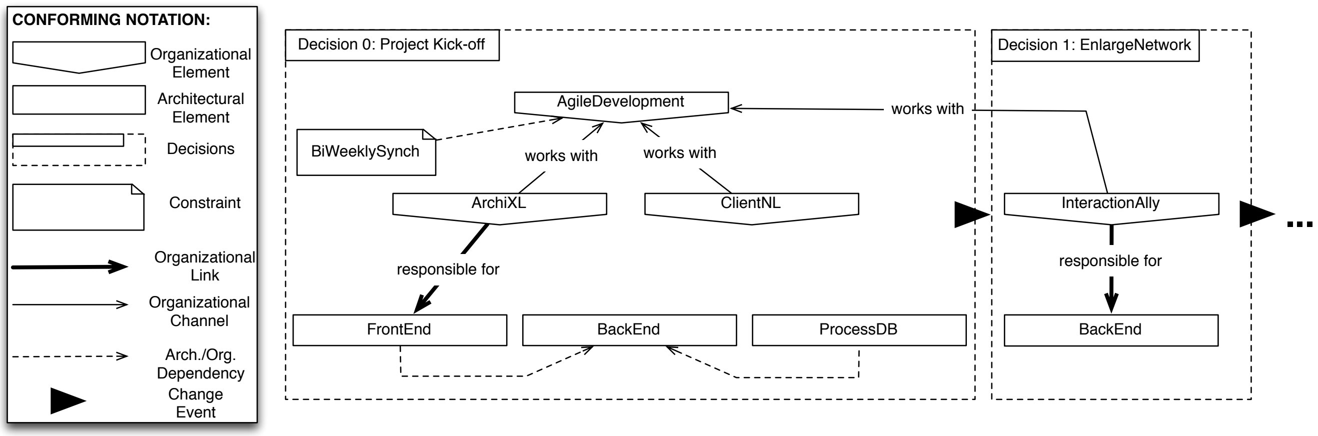

Figure 2: Instances of Networked-Organizations Architecture Viewpoint.

aspect provides insights into which organization is skill-wise best suited to be responsible, participate, or consult in the AE's implementation.

C4 *ProductionOf*–*Responsible* organization: The designer needs to check the assignment links between *ProductionOf* and organizations for the *InvolvementType* (set by default to *Participant*); only one may be set to *Responsible*. The *SkillAssignment* aspect provides insights into which organization is best suited skill-wise to be responsible; potentially adding new, suitable organizations.

C5 *ProductionOf*–*Skill* mismatch: given overall *CompetenceDemand* and *CompetenceOffer*s match (see C1), then inappropate or inadequate organizations are participating in the *ProductionOf* one or more AEs. The *SkillAssignment* aspect identifies which organization to involve. When many architecture elements are combined in one *ProductionOf* mapping, dividing the AEs among multiple *ProductionOf* mappings, each only involving the right skilled organizations, improves mapping manageability.

C6 *OrganizationElement*–*ProductionOf* mapping: given the other constraints are fulfilled, the *SkillAssignment* aspect provides insights which *ArchitectureElement*, and thus which existing *ProductionOf* mappings might benefit most from the organizations skills.

Notes

1. For further discussion of this viewpoint, see [5].
2. Figure 2 shows an example of a NOVA view.
3. The hADL4nova meta model, scenario example, and some screenshots are available on-line from [6].
4. The latest version of this viewpoint will be maintained at `http://www.iso-architecture.org/viewpoints/NOVA`.
5. Comments or questions? Contact one or the authors via email.

References

[1] Romina Eramo, Ivano Malavolta, Henry Muccini, Patrizio Pelliccione, and Alfonso Pierantonio. A model-driven approach to automate the propagation of changes among architecture description languages. *Software & Systems Modeling*, 11(1):29–53, 2012.

[2] Uwe van Heesch, Paris Avgeriou, and Rich Hilliard. A documentation framework for architecture decisions. *The Journal of Systems & Software*, 85(4):795–820, April 2012.

[3] Rich Hilliard. Architecture viewpoint template for ISO/IEC/IEEE 42010. `http://www.iso-architecture.org/42010/templates/`, June 2012.

[4] Philippe Kruchten, Rafael Capilla, and Juan Carlos Dueñas. The decision view's role in software architecture practice. *IEEE Software*, 26(2):36–42, 2009.

[5] Damian A. Tamburri, Patricia Lago, Christoph Dorn, and Rich Hilliard. Architecting in networked organizations. Submission to WICSA 2014, October 2013.

[6] Damian A. Tamburri, Patricia Lago, Christoph Dorn, and Rich Hilliard. hADL4nova model and screenshots. `http://christophdorn.wordpress.com/current-activities/hADL4nova/`, October 2013.