Finnish ballot comments for ISO/IEC JTC1/SC7/WG17 W17N0127 ODP Enterprise language Final Committee Draft

Finland votes NO for the progression of the current draft document, with the following comments.

FIN1 G Summary of concerns Inconcistent use of terminology

Problems

- Part 2 terminology is not explained enough for readers to follow the discussion
- Cobehaviour missing
- Implementation objects vs. modeling world at different levels of abstraction still mixed especially when trying to match processes with role
- Audiences and usage examples not clear
- Force concepts form not a ful set, other similar groups of concepts have not been spelled out this way – language is extensible.

Proposal

Suggestions in FIN9, FIN11, FIN14, FIN23, FIN68

FIN2 G Summary of concerns Relationship of communities

Rationale

- Federation is not dealt with. Federation means creation of a new community with a new set of governing rules and the federating communities need to be able to provide this specification.
- Cluttered otherwise, same conceptual things reappear several times in different terms.
- C-object and S-community have a special position, however, it does not clarify the situation as at the same time, it is acknowledged that several c-objects could be simultaneously be interacting within the same community.
- Relationships between c-object, role, s-community, refinement via population process are not clear enough.
- Relationships between enterprise specifications and organizations are undefined; likewise ownership and domain communities and other community types have relationships that could be clarified.
- Community population process is not described.

Proposal

See FIN21, FIN22, FIN24, FIN45, FIN57

FIN3 G Summary of concerns Reationship of roles and processes

Rationale

- Relationship between roles and processes is unclear.

Proposal

See FIN19, FIN20, FIN51-FIN62

FIN4 G Summary of concerns Community life-cycle

Rationale

- The specified community could exist (be established and have responsibilities) without fixed identified objects populating it – late binding can be used for populating it
- Rules for changing the community structure are unclear and unnecessarily strict (no federaition allowed)
- How to decide whether an enterprise object can join a community?
  - Rules for community to accept a new member as being capable of performing its duties (assignment rules)
  - Rules for an object to inherit the policies of the community it joins (inconsistent)
  - Rules for an object to check the acceptability of the community policies when joining (missing) or as a preforming member (missing or inconsistent)

Proposal

See FIN39, FIN41, FIN42, FIN46-FIN49

FIN5 G Summary of concerns Behaviour and policies

Rationale

- Situations may arise in which an object is allowed to ask for a service which is prohibited from use. No failure or violation management concepts are offered.
- Relationship between policy and behaviour is unclear.
- Relationship between processes and policy are unclear.

Proposal

See FIN14-FIN18, FIN27-FIN31, FIN40, FIN44, FIN50, FIN63-FIN67

FIN6 G Summary of concerns Viewpoint correspondencies

Proposal

See FIN70-FIN76

FIN7 E 0, line 6, page 1 Spelling

Rationale

Missing word.

Proposal

Replace “Open Distributed” with “Open Distributed Processing”.

FIN8 E 2, page 3, line25 Spelling

Proposal

Replace “con tain” with “contain”.

FIN9 TH clause 5 induced rewrite of clause 5

Rationale

Collected together the comments on the audience of this standard, on the exploitation scenarios for this standard, and the clarity and spelling of clause 5 (FIN9-FIN14) induce the following rewrite of clause 5.

Proposal

5. Motivation and overview

The purpose of this Recommendation | International Standard is to provide a common language (set of terms and structuring rules) to be used in the preparation of an enterprise specification capturing the purpose, scope and policies for an ODP system. Such an enterprise specification forms a part of the specification of and ODP system in terms of the set of viewpoints defined by ITU-T Recommendation X.903 | ISO/IEC 10746-3.

The primary audience for the document consists of those who prepare and use such specifications. The motivation for a standard enterprise language is to support standardised techniques for specification. This improves communication and helps create consistent overall specifications.

The preparation of specifications often falls into the category referred to as analysis of requirement specification. There are many approaches used for understanding, agreeing and specifying systems in the context of the organisations of which they form a part. The approaches can provide useful insights into both the organisation under consideration and the equirements for systems to support it, but they generally lack the rigour, consistency and completeness needed for thorough specification. The audiences of the specifications also vary. For agreement between the potential clients of an ODP system and the provider of that system, there may be needed to have different presentations of the same system - one in terms understood by clients, and one in terms directly related to system realization.

The use of enterprise specifications can be wider than the early phases of software engineering process. Current trend is to integrate existing systems into global networks, where the functionality of interest spans multiple organisations. The enterprise language provides means to specify the joint agreement of common behaviour of the ODP systems within and between these organisations. The enterprise specification can also be used at other phases of the system life-cycle. The specification can for example be used at system run-time to control agreements between the system and its users, and to establish new agreements according to the same contact structure. Enterprise viewpoint specifications may contain rules for inter-organisational behaviour.

This standard also provides a framework for development of software engineering methodologies and tools exploiting ODP viewpoint languages, and a set of concepts for development of denotational languages for ODP enterprise language itself. For these purposes this standard provides rules for the information contents of the specification and the grouping of that information. Further requirements on the relationships between enterprise language concepts and their mapping to concepts in other viewpoints are specific to the methodologies, tools or denotational languages to be developed.

An enterprise specification defines the scope, purpose and policy framework of an ODP system. The specification gives a statement of conformance for system implementations. The system purpose becomes defined by the specified behaviour of the system; policies capture further restriction of the behaviour between the system and its environment or within the system itself. Policy is a set of rules related to a particular purpose. The expressed obligations, permissions and prohibitions are in the context of communities naturally related to the business decisions of the system owners.

An ODP system may span multiple domains and need not to be owned by a single party. The system can also be divided to independently specified and independently working subsystems. In such a case, an essential part of the enterprise specification is to capture the collective behaviour of the full system.

The enterprise specification comprises of a set of community specifications. The division to communities can be made based on functional grouping, control domains or ownership reasons. For each enterprise specification there should be a root community specification in which the scope of the system becomes explicated. This community may represent the ODP system as a single object or if the division for example to multiple owners is important to a set of ODP system objects, each of which become further refined by other community specifications.

A community specification captures the requirements for objects in terms of roles and interactions between these roles. The community specification also captures policy statements further stating what rules guide the system in situations where multiple behaviour patters are possible. Policies can be separately set for each case in which the same specification is otherwise used.

The community that conforms to the community specification is comprised of enterprise objects. The enterprise specification may include the definition of these objects or trust on supportive mechanisms to introduce those objects into the communities during the life-time of the communities.

The community specification contains rules for assigning enterprise objects to roles. These rules reflect the way in which enterprise objects are introduced and the way in which binding of objects to roles can be implemented in the factual system environment.

Some basic concepts used in this standard are defined in RM ODP Part 2 or 3. The enterprise language definitions use them in the following contexts.

A community specification captures the contract between objects within the specified community. For a community contract the essential aspects of the contracts are

- the behaviour made possible by the roles involved;
- policies governing the behaviour of the community;
- rules for changing the structure of the community (roles, policies); and
- rules for deciding when the community contract is violated and what actions are taken then.

A role is an identifier for a behaviour. A role is used to project a behaviour that needs to be assigned to a single enterprise object. A behaviour is generally defined to be a collection of actions with constraints on their occurrences. A behaviour is always associated with an object or with a composition of objects. For communities, the interesting behaviour is formed by the interactions between the community and its environment and between the objects fulfilling roles in the community. A role thus identifies a projection of the community's collective behaviour specification.

A process is an alternative projection of the community's collective behaviour; an action belongs to a process if it supports a shared goal independent on the performer of that action.

Role, contract and policy concepts are used as directly used as defined in ITU-T Recommendation X.902 | ISO/IEC 10746-2, other concepts are defined in this Recommendation | Standard.

FIN10 TH 1,5, 8 Audience of the document

Rationale

In clause 5, lines 26-33 of page 5, Canada expects to hear about the intended audience of this document. The Enterprise language should address both audiences, and this statement is already present in Clause 1. In addition, clause 5 should contain clearer statements on how these audiences should use the language in various situations. The list of situations should be expanded. The use as a basis for tool and language construction is also reflected to clause 8, where requirements should be set for tool providers to nail down some of the correspondence statements now left free for the specifiers.

Proposal

Proposed changed text for clause 5, lines 38-39: Replace the sentence “It also provides a rigorous framework for the development of supporting methods and tools” with a new paragraph:

> This standard also provides a framework for development of software engineering methodologies and tools exploiting ODP viewpoint languages, and a set of concepts for development of denotational languages for ODP enterprise language itself. For these purposes this standard provides rules for the information contents of the specification and the grouping of that information. Further requirements on the relationships between enterprise language concepts and their mappings to concepts in other viewpoints are specific to the methodologies, tools or denotational languages to be developed.

Proposed text for clause 8, add between lines 10 and 11 at page 24:

> When this standard is taken as a basis for a standard methodology or a tool supporting a detailed software engineering methodology, the tool or methodology designer can specify a list of correspondences between viewpoint concepts or between enterprise language and the modeled system concepts.

FIN11 TL 5 Usage examples

Rationale

Needs more concrete understanding on the situations in which enterprise language can be applied.

Proposal

Proposed changed text at line 5, page 6, first sentence after note:

> When preparing a specification, there are many approaces used for understanding, …

Proposed new paragraph between lines 34 and 35, page 6:

> Current trend is to integrate existing systems into global networks, where the functionality of interest spans multiple organizations. The enterprise language provides means to specify the joing agreement of common behaviour of the ODP systems within and between these organizations.

FIN12 E 5, page 6, line 30, lines 33-34 Clarity

Proposal

Replace “Thus, two” with “Several”.

Remove last sentence of the paragraph.

FIN13 E 5, page 6, lines 35-39 Clarity

Proposal

Replace the last paragraph with:

> The enterprise specifications can also be used at other phases of the system life-cycle. The specification can for example be used at system run-time to control agreements between system and its users, and to establish new agreements according to the same contract structure. Enterprise viewpoint specifications may contain rules for inter-organizational behaviour.

FIN14 TH G, 5 Relationship between Part 2 and EL

Rationale

Part 2 concepts are general and thus difficult to generally apply to enterprise language. The application of the concepts need to be refined for each specific context, thus the readers should be given some guidance for interpretation.

Proposal

Add role, behaviour, action, activity, contract, policy, etc introductions and interpretations to clause 5, which currently works only as a motivation section. See FIN9 for proposed text.

FIN15 TL 6.1.2 Objective

Rationale

The system objective is not a syntactical entity. Therefore, the word sentences is extra. The representations of the objective become visible in behaviour, contracts and policies as described in FIN9.

Proposal

Replace the definition by

> "Preferences about possible future states, which influence the choises within some behaviour".

Add note:

> 3 – Statements of objective may be an explicit part of the specification depending on the denotation and methodology selected. The objective of the system may also be indirectly present for example in policies governing the system and its behaviour.

Remove note 2.

FIN16 A 6.1.3 Scope, answer to question raised

Rationale

We feel no need for additional concepts here. When there is a need to express the actual capabilities of a system, the terms capability and system behaviour are available.

FIN17 TH 6.1.x, 9.3 Scoping statement

Rationale

See clause 8 and capture that rationale and semantics into a definition.

Proposal

Add definition

> A scoping statement specifies the preconditions for the use of the community. The scoping statement expresses either informally or formally what the community is intended to do and what properties the environment must have for the community to operate.

Remove 9.3.

FIN18 TH 6.1.4 Purposeful selection

Rationale

Policies as means of controlling alternative behaviours is equally powerful, already existing mechanism.

Proposal

Remove 6.1.4.

FIN19 TH 6.1.5 Process

Rationale

- Continuous processes are denied.
- We don't understand the need of multiple starting points. How is starting point defined?
- What is a result?
- Is there really difference between activity and process?
- Is a process a partially ordered sequence of parallel activitites of enterprise objects.
- Could we use the term workflow instead?
- Clause 7.8 is more clear in this respect and material from there should be moved here as a definition.

Proposal

Replace the current definition with the following.

> **Process:** Composite behaviour of the community leading to a preferred state in the community. Process can be expressed as a partially ordered sequence of steps.
>
> 1 - Process is an activity with objective.
>
> 2 - Activities can be refined and thus repetitive behaviour patterns can be modelled.
>
> Note – The process specification is independent of the identity of performing enterprise object.
>
> Note - The process specification is abstract, and clearly differs from a computational description of the cobehaviour between actors. The goal of process specification is to give a partial order for actions within the community, not to fully specify the behaviour.
>
> Note – A process specification can represent a workflow description.
>
> Note – Role is thus an identifier for a part of a composite behaviour. Role can be expressed as a collection of steps it participates and its internal behaviour.

FIN20 TH 6.1.6 Step

Rationale

Misreading of part 2…. 8.3 does not require an identified object to be associated.

Proposal

> **Step:** An action associated with one or more roles.
>
> 1 – An action is a modeling concept that is not reserved to a single occurrence of an action. However, the term action is for short also used for action occurrences.
>
> 2 – As the granularity of actions is a design choise, the actions can be further refined as more detailed behaviour.

FIN21 TH 6.1.7 S-community

Rationale

The need for a community with a special role is not properly revealed by the text, thus also the intended use of the concept is left unclear. The term has been under debate quite a while.

Proposal

Proposed replacement text:

> **Root community specification.** A community specification in which an ODP system can fullfill a role. The role specification thus expresses how the ODP system is required to interact with its environment.

Add notes:

> 1 - The root community is a representation of the scope of a system.
>
> 2 - For example, in cases where federation needs to be modeled, there may be more than one ODP systems simultaneously fulfilling roles in the root community.
>
> 3 - The ODP system is represented as a single enterprise object, possibly as a community object.
>
> 4 – For expressing an environment contract (for example for reuse purposes and runtime binding across ODP systems) it is preferred that the root community specification includes a scoping statement.

FIN22 E 6.1.8 Community object

Rationale

Term under debate. The suggested term is one of the best alternative in the first rounds of discussion. Then it was not appropriate as there was a dangerously close counterpart with it, which now has totally dissappeared.

Repetition that causes confusion.

Proposal

Rename with “community object”.

Remove the latter sentence.

FIN23 TL 6.1.10 Machine

Rationale

All examples given are already concepts that can readily be used in any ODP system description as entities or can be described as ODP systems.

Proposal

Delete 6.1.10.

FIN24 E 6.2 Title

Rationale

The contents is mostly about objects instead of roles.

Proposal

Retitle as Role and enterprise object related concepts

FIN25 TH 6.2.x Assignment rule

Rationale

The population process of communities is not described properly, partly because the basic concepts are not yet defined.

Proposal

Add

> Assignment rule: denotes the criteria on which an enterprise object is selected to perform a role. The criteria defines what the object to fulfil a role must be capable of doing, not restricted from doing by earlier commitments, and what relationships to other objects are required or denied.

FIN26 TH 6.2.7 Answer to editor question

Proposal

Should use interactions instead of actions.

FIN27 TL 6.3.1 Authorization

Rationale

Authorization is one of the force concepts, and belongs to a different kettle than permission, prohibition and obligation.

Proposal

Remove authorisation from line 4.

Remove note 1.

FIN28 TL 6.3.2 Authorization

Rationale

In distributed environment, there are no mechanisms that can be used to guarantee this in a general case. Although the concept is useful in special cases, introducing it here gives faulty expectations. When specifiers need to define their own authorization concepts they also need to consider whether that is actually possible within the circumstances they have chosen, or not.

Proposal

Remove 6.3.2.

FIN29 TH 6.3.x Policy framework

Rationale

The concepts of objective, cobehaviour, policy and nesting policy frameworks are poorly understood, mostly because an essential structuring concept is missing.

Proposal

Add

> Policy framework: A structuring rule in a community specification that specifies for which purposes policies are set in these communities.
>
> > Notes -
> >
> > 1 - When communities are nested, the enterprise objects must conform to all policies in all communities it participates. As the participated communities may have different policy frameworks, the effective set of policies becomes larger.
> >
> > 2 - When a community plays a role in another community, the policies may not be contradictory, but the policy frameworks may differ.
> >
> > 3 - Establishing a federation means creation of a new community, thus, a new policy framework is created that is consistent with the policy frameworks of the establishing parties. For example, the members of the federated community must conform to both the communities within their organizations and the federated community spanning across the organizatonal boundary. The federated community policy framework may leave some aspects of common behaviour without policies. Therefore, enterprise objets may run into situations where their policies contradict.

FIN30 E 6.3.3 Violation

Rationale

There are violations against explicitly stated policies, i.e., failures (violations against a contract). These should be considered as errorousness behaviour implementation or specification. There are also violations caused by inconsistent assumptions between communicating parties about the permissions, obligations and prohibitions. These may arise for example in federation situations where there is no full control to the interacting objects environment or in other situations where an action is not considered to be essential enough to be specified with policies in detail for all possible participants of an interaction. Only some of these situations can be considered as design or implementation errors. Mechanisms are needed for catching violations and for using appropriate recovery or sanction mechanisms.

Proposal

Rephrase

> An event causes a violation situation to arise when their expectations on permissions, prohibitions or obligations on an action do not match.
>
> Notes –
>
> 1. Violations occur while an enterprise object tries to follow requirements associated to its role within either a single community or within multiple communities.
> 2. Failure is a violation against a consistent contract.
> 3. Violations can also occur because of inconsistencies between the contracts involved.
> 4. There can be policies or rules expressed for a community or for a federation defining recovery or sanction mechanisms for failures and other violations.

FIN31 TH 6.4.1 Purposeful selection

Rationale

Remove references to purposeful selection.

Proposal

Rewrite as:

> A behaviour of parties or agents related to community objective.

Remove both notes.

FIN32 TH 6.4.3 declaration

Rationale

The concept of truth is subjective. In a distributed environment, there are no mechanism to force a given state of affairs to take effect on a domain where the statement maker has no authority.

Proposal

Rewrite as:

> An act by which a party publishes to other parties a statement about the state of affairs the party is committed to support.

> Note – State of affairs may be related to the ODP system state, environment state, or future behaviour of the system or environment. The statement may be about a community or an enterprise object in the system that is controllable by the party or outside the control of this party.

FIN33 E 6.4.4 delegation

Proposal

Replace note 1-2 with

> Note – Delegation may be from a party to a system and the system may also be delegated an authority to make further delegations.

FIN34 E 6.4.5 Evaluation

Rationale

Ambiguity.

Proposal

Rewrite as:

> An act that assesses the value of something.
>
> Note – value can be considered in terms of usefulness, importance, preference, acceptabilty etc; the evaluated target can be a system state, a potential behaviour, etc.

FIN35 E 7, page 10, line 17 contradictory statements

Rationale

C-community does not exist; note on lines 19-20 contradictory with lines 17-18.

Proposal

Line 10: The ODP system may participate more than one communities.

Line 17: Delete sentence.

Line 17: replace c-community with "root community specification".

Line 17: roles, processes -> roles or processes (too strong requirement otherwise)

FIN36 E 7.1 Repeating text

Rationale

Repeated in 7.2

Proposal

Delete final note (lines 40-43) from 7.1

FIN37 E 7.1, between lines 5 and 6 Nature of ODP systems

Rationale

The nature of ODP systems is not clear.

Proposal

Add:

> The ODP system under specification may span multiple organizations.

FIN38 E 7.1, between lines 9-10 Levels of abstraction

Proposal

Add:

> Note – The set of community specifications selected may be induced by using various levels of abstraction, functional decomposition of the system and ownership of the system or parts of the system.

FIN39 TH 7.2 Assignment rules

Rationale

Answer to editors question: Yes.

FIN40 TH 7.3.1 Policy framework

Rationale

Use the definition of policy framework in FIN29 to clarify the structure of community specifications.

Proposal

Add to the end of 7.3.1:

> Each community must be associated with a single policy framework.
>
> Note - The policy framework may form part of a nested hiearchy of policy frameworks. This may position the community within a larger environment, for example, in respect of some organizations.
>
> Note – The community may be federated, i.e., be composed of domains. The policy framework and policies within it are established when the community is specified or when established according to the specified establishing behaviour. The establishing behaviour may involve domain controllers or already established other communities.

FIN41 TH 7.3.1, page 11, line 26 Population process

Rationale

Hopefully just editorial.

Proposal

Replace “States the assignment of enterprise objects to role” with “States the assignment rules of enterprise objects to role”.

FIN42 E 7.3.1, page 11, lines 27-36 Establishing a community

Rationale

Text is ambiguous and difficult to interpret.

Proposal

Replace with

> The contract is formed by a defined process carried out by a community that has the objective of forming a new contract. The members of this establishing community include parties, agents representing these parties, or members of a new community exploiting the contract-to-be. This establishing community appears in an earlier epoch than any community based on the new contract. This earlier epoch may represent a defined design process, defined negotiation process between agents, or commitment of agents to a new community.
>
> Note - The first epoch involved is a design process.

FIN43 E 7.3.1, page 12, line 1 Use of EL

Rationale

Defines methodology.

Proposal

Combined -> related

FIN44 E 7.3.1, page 12, line 4-6 Combinations of policies

Rationale

Unclear text

Proposal

Replace by:

> The collective behaviour composed either from role specifications or from process specifications is constrained by the joint policies associated to roles and by the community contract.
>
> Note – there is a substructure in role related policies that can be matched with step conststraints.

FIN45 TH 7.3.2 Relationship between communities

Rationale

The text still gives a number of overlapping examples and fails to reveal all interesting cases. It has been found extremely difficult to interprete.

Proposal

Replacement text:

An enterprise specification can include several communities. The scope of the enterprise specification becomes determined by the choise of one or more root community specifications.

Other communities can further refine the behaviour of these root communities. The communities can be used as a structuring concept, for example, for splitting the specification into subsystems with separate functional responsibilities, into domains for describing ownership and authorization, or into domains for describing controlling and contract making facilities.

The community specifications may overlap each other. In this way it is possible to show how for example functional and administrative communities must interact with each other. The enterprise specification can enforce community interaction at least in the following ways

- communities are nested so that a community becomes a refinement of a role in an outer community and thus inherits all contractual and policy related aspects;

- named roles in one or more communities are required to be fullfilled by the same enterprise object (which can represent a community specified separately, see community object definition in x.x);

- some roles in the community specification are selected and defined so that they explicitly describe how interaction takes place between the communities; more precicely in each community an interface role is described to interact with the community environment and the specifier has to take care that all those environment interactions become captured with interface role actions in appropriate community specifications; and

- one or more community specifications may include behaviour for creating new communities; for example, federation establishment means creation of a new community involving the definition of appropriate policy framework, structure for the community and the community contract.

The actual communities are related to each other following the possibilities created by their specifications. In addition, an enterprise object may fulfil a role in multiple communities as it has not been prevented from doing so. In this case, the amount of information flow or actions involving multiple communities is mainly accidental, and it is the responsibility of the specifier to restrict unwanted interactions.

Enterprise, community and role specifications can be considered as textual items managed by some specification tool and support environment. Reuse of these items is acceptable and the specifications may use textual references to items available in the support environment instead of forcing them to be embedded in the same specification.

FIN46 E 7.4, lines 26-28, 31-33 community life-cycle

Rationale

Unclear.

Proposal

Replace lines 26-28 with

> An enterprise specification may include specifications of enterprise objects; the group of such objects is a free choice of the system specifier.

Replace lines 31-33 with

> An enterprise object may be refined as a community at a greater level of detail.
>
> For the purposes of modeling enterprise viewpoint behaviour, the enterprise object behaviour is restricted by the roles it is assigned to.

FIN47 E 7.4, last bullet community life-cycle

Rationale

Not precise enough.

Proposal

Replace by

> The object joins the community during the life-time of the community.
>
> > Note – The community specification includes assignment rules for populating the community, thus, the configuration of identified enterprise objects is not necessary for establishing a community. A role can be empty at times.
> >
> > Note – The community specification can include rules that change the community structure (for example, number of roles).

FIN48 TH 7.6 community life-cycle

Rationale

The relationship between the community specification which determines the structure, configuration, rules and policy framework of the community and the community itself is not clear. The text has an underlying assumption that a community is born by instantiating all of its enterprise objects, which in all cases is not wanted. The changes made try to bring out aspects of communtity structures necessary to maintain independently form the enterprise object assignments for the community.

Proposal

> **7.6.1 Establishing and maintaining a community**

A community specification include establishing behaviour for a community. The establishing behaviour may be implicit or explicit, but it establishes the required structures and responsiblities to maintain and control the community, e.g., the policy framework, the community contract and the community membership need to be known.

7.6.2 Populating a community

The community specification includes assignment rules for choosing enterprise objects to fulfill the specified roles. The rules can directly identify the objects wanted, or trust on a supporting mechanism for which more complex rules can be given for population to take place. Such rules may be based on object identities, relationships between objects, object capabilities, technologies, preceding commitments, object behaviour, etc.

The community specified may be a previously existing one, thus the population process only captures the relationships between the objects and the community.

The community specified may need to be created, even, the members of the community may need to be instantiated as the community comes established.

There is no requirement for a community to be fully populated at any time, although the responsiblities taken by the community are present during the life-time of the community. Members to the community can be selected by demand according to the assignment rules for that community. If an enterprise object ceases to fulfill the assignment rule associated to it, it violates the community contract.

7.6.3 Changes in a community

Changes in the structure or behaviour of a community can occur only if the enterprise specification includes behaviour that can cause such changes.

The changes to be considered here include

- introduction of new policy rules within the existing policy framework; or changing the existing policy rule; and

- introduction of new roles into the community.

7.6.4 Terminating a community

An enterprise specification or a community specification may include a terminating behaviour for a community.

Notes

1 – Some communities are permanent and never terminate.

2 – Termination may be triggered by reaching the community objective or failure.

3 - The specification need not to be symmetric in respect to establishing and terminating behaviour. Either of them can be present or missing.

FIN49 TL 7.7 purposeful selection, objective, policy

Rationale

Clarification for the relationship of concepts objective and policy, without the concept of purposeful selection.

Proposal

Line 17, add:

> In this case, the sub-objective defines the state in which the process terminates.

Between lines 17-18, add:

> The policies of a community restrict the community behaviour in such a way that it is possible to reach the objective. Such policies result in the selection of behaviour that suits the objective of the community.

Lines 20-22, replace with:

> An enterprise object can only be assigned to a role when the objectives of the object and the community and the role are consistent.
>
> > Note - This also applies to community objects and the corresponding more detailed community.
> >
> > Note – The enterprise specification may provide for detection of conflicts in objectives and for resolution of those conflicts.

FIN50 E 7.8.1, 5 inappropriate text style

Rationale

Descriptive text, not normative rules. Not suitable for clause 7.

Proposal

Move 7.8.1 to 5.

Add to 5 the idea that behaviour refers to collective behaviour unless otherwise stated: an object can always have individual behaviour but the object can also be a composite object or represent a community.

The behaviour can be split into specifiable segments by projecting it into roles or processes. The roles and processes both can be subdivided into steps, which bridges between these two projections together. Policies need to have the granularity of step to be usable for both specification styles.

Possibly add notes for 6.1.5, 6.1.6 for the relationship.

FIN51 E 7.8.1, 7.8.5.2 process

Rationale

After FIN19 clause 7.8.1 is empty and the text does not really need to be in a separate clause.

Proposal

Move 7.8.5.2 to form a new version of 7.8.1

FIN52 E 7.8.2, lines 21-23, line30 role rules

Proposal

Move lines 21-12 just before line 30.

FIN53 E 7.8.2, lines 17-20 and 24-29 role rules

Rationale

Contradictory text.

Proposal

Replacement text:

> An enterprise object may fulfill several roles in one community, and may fulfil roles in several communities. An object associated to several roles becomes constrained simultaneously by all the behaviours named by the roles and the policies associated to these roles.
>
> At any location in time at most one enterprise object is associated with each role. The constraints of the behaviour named by the role become constraints on the object fulfilling the role. A role may be filled by different objects at different times, even, a role may be unfilled, provided that the specification of the community so permit.

FIN54 E 7.8.2, line 30 role rules

Rationale

Unclear.

Proposal

Replace by

> Enterprise objects of different types may be assigned to a role, given that the object type not contradicts with the role type.
>
> > Note - The community supporting system environment or the community specification may include mechanisms to determine and resolve inconsistencies between object types and requirements set by roles, thus enlargening the group of acceptable objects for a given role.

FIN55 E 7.8.2, page 16, lines 16, 39-41 role rules

Proposal

Add on line 16:

> Integrally related to the specification of a role there is an assignment rule that sets requirements for the potential objects to fulfil that role.

Replece 39-41:

> Assignment rules for a role include prescription concerning the assignment of enterprise objects to roles.
>
> Note - The existence of these rules in a community specification is compulsory, but the form in which they appear is dependent on the support mechanisms for that community. Some execution environments or specification tools support only assignment of identified objects, others support type based matching and late binding within the communities.

FIN56 E 7.8.2, page 16, line 39 – page 17, line 15 role rules

Rationale

Repetitive, and at least partially incorrect.

Proposal

Remove notes; except note 2 that may be kept and moved under line 29 on page 16.

FIN57 E 7.8.3 Interactions between communities

Rationale

Contents already revealed by 6.2.7 more clearly (second case), first case has nothing to add for relationships between communities (not actually talking about interface roles but interactions between roles within an additional community).

Proposal

Delete clause.

FIN58 E 7.8.4 Enterprise objects

Rationale

Clause 7.8.4 gives only trivial rephrasing of clause 6 definitions. All rules related to enterprise objects are given through roles in the specifications, and thus become stated in 7.8.2.

Proposal

Remove 7.8.4. Rename 7.8.2 as Role and enterprise object rules.

FIN59 TH 7.8.5 Process rules

Rationale

Current text causes more questions than it is able to give answers, mainly because of inconsistencies.

Proposal

Replace by

In an enterprise specification, a process is an abstraction of the behaviour of some configuration of objects in which the identities of objects have been hidden as a result of the abstraction. Each step need not be performed by the same enterprise object. A step can even be performed jointly by multiple roles.

If processes are used as part of the community specification, each step must be associated to an actor role.

The process specification must include specification of the state which initiates the process, and the state that terminates the process.

The collective behaviour of a community is a set of processes. This set can be seen as a more abstract process performed by a single role performed by a c-object. Also, a step of a process can be further refined as a more detailed process.

FIN60 E 7.9, page 19, lines 13-16 policy rules

Rationale

Answer to editor question on line 14: No.

Proposal

Change line 13:

behaviour for changing the policy

FIN61 TH 7.9.1 policy rules

Proposal

Replace by:

In an enterprise specification, policies can be associated with

- communities, including the root community representing whole ODP systems
- roles
- steps as parts of processes
- enterprise objects.

Include 7.9.5 here

Include 7.9.3 here

FIN62 E 7.9.2 policy rules

Proposal

Delete lines 33-35, as they cause more confusion than clarification.

FIN63 TH 7.9.2.2 permission and successful operation

Proposal

Replace editor note by

> Note – There is, however, no quarantee that the action succeeds. The action may have participants, for example, at other domains on which the action is prohibitied.

FIN64 E 7.9.2.3 prohibition

Rationale

Clarification.

Proposal

Line 20: delete "like a permission".

Line 26: Cannot -> is not allowed.

FIN65 TL 7.9.2.3, editor question authorization

Rationale

Proposal

No need for authorization subclause. Check spelling of authorization/autorisation.

FIN66 TL 7.9.4 Policy violations

Rationale

See FIN29

Proposal

Add to the beginning of 7.9.4:

> There are violations against explicitly stated policies, i.e., failures (violations against a contract). These should be considered as erroreousness behaviour implementation or specification. There are also violations caused by inconsistent assumptions between communicating parties about the permissions, obligations and prohibitions. These may arise for example in federation situations where there is no full control to the interacting objects environment or in other situations where an action is not considered to be essential enough to be specified with policies in detail for all possible participants of an interaction. Only some of these situations can be considered as design or implementation errors. Mechanisms are needed for catching violations and for using appropriate recovery or sanction mechanisms.

FIN67 TH 7.9.5 organization of policy

Rationale

Use of the policy framework for organizing and structuring policy rules.

The concepts of policy type, policy template and policy instance are not a useful classification for this clause. Policies are instances by definition, and they are organised by a policy framework. The implementation of the policy framework may cause restrictions for policy types and even to policy templates when entering actual policies is discussed. However, those discussions are not of suitable abstraction level here.

Proposal

Replacement text:

> An enterprise specification and a community specification include a policy framework that defines what kind of policy rules are needed to guide the behaviour of enterprise objects playing various roles in relation to the ODP system in question. The policy rules form a hierarchy and controlled domain in the manner determined by the enterprise specification structure.
>
> Policy rule covers for example
>
> - limits on the system, community or object behaviour in terms of possible interaction sequences with other corresponding entities;
> - limits on the quality of service obtained from the peer system, community or object;
> - limits on the identity of peer systems, communities, or objects interacting with an entity;
> - limits on the technology by which such interactions can be performed;
> -

FIN68 E 7.10 force rules

Rationale

Looks more like definitions than rules for the usage. Only 7.10.3 and 7.10.4 gives further information than the definitions. Questions to ask here: What is the specifier required to say? What would cause contradictory rules? What are the implications of the specification for the community and its environment?

How policies and force rules are related?

An enterprise specification contains policies and rules for changing these policies. The policy changing rules need to specify who are allowed to change the policies. Force rules can be used to express how the chain from parties to the policy changing agents is formed and modified, and how the responsibilities of actions of the system gets distributed to the parties according to this chain. Thus the access rights and responsibilities become inseparable.

Proposal

FIN69 E 8 clause title

Rationale

Readability.

Proposal

Remove subtitle 8.1. and use the title Viewpoint correspondences for clause 8.

FIN70 TL 8, lines 10-11 categorization expansion

(part of FIN10)

Rationale

The situations of interest

- level of requirements stated in RM ODP
- requirements and interpretations build in into tools and methods
- relationships stated in an individual specification

Proposal

Proposed text for clause 8, add between lines 10 and 11 at page 24:

> When this standard is taken as a basis for a standard methodology or a tool supporting a detailed software engineering methodology, the tool or methodology designer can specify a list of correspondences between viewpoint concepts or between enterprise language and the modeled system concepts.

FIN71 TH 8.2.1 correspondences / EL and info

Proposal

Delete bullets actor and artefact; replace process by action.

FIN72 TH 8.2.3 correspondences /EL and info

Proposal

a) Replace temporary not on lines 7-8 with

> Note – An information object may appear in lists of multiple enterprise objects.

b) Replace "with the roles it fills" on line 10 with "filling this role".

c) Replace bullet on lines 14-15 with

- for each action in the enterprise specification, the information objects that have a dynamic schema further contraining the realization fo that action as the action does not become defined by these contstraints.

FIN73 TH 8.3.1 correspondences / EL and computational

Proposal

a) Remove actor role and on line 23.

b) Add on line 23

> Note: actor and interface roles are related to enterprise objects within the enterprise specification.

c) Replace line 24 "enterprise interaction" with

- interaction between roles

> Note – action between roles are essentially captured by interactions.

FIN74 TH 8.3.3 correspondences / EL and computational

Proposal

a) Add in the beginning of the list

- for each enterprise object, a configuration of computational objects for realizing the required behaviour

> Note – the identification and introduction of enterprise objects fulfilling an actor role can be indirect, basaed on elaborate assignment rules and supporting system functionality
>
> Note – the configuration can be expressed either direcly or indirectly as a set of rules for constructing the configuration.

b) Page 26, lines 1-2 remove "or is qualified by a predicate" and add note

> Note – if the same operation or stream can be used for multiple enterprise viewpoint actions, the computational specification has to include predicates for qualifying each case.

c) Page 26, line 6, replace "that are constrained by the enterprise interaction" by "required by the enterprise interaction or a rule for identifying such bonding types".

d) Page 26, line8, add to the end "or a rule for identifying such behaviour type."

FIN75 TH 8.4.1 correspondences / EL and engineering

Proposal

Remove bullets principal role, actor role and basic engineering object.

FIN76 TH 8.4.3 correspondences / EL and engineering

Proposal

a) Replace bullets on lines 31-34 by

- for each enterprise object in the enterprise specification, rules about assigning some or all of its behaviour to engineering nodes; these rules may capture policies from enterprise specification;
- for each interaction between roles in the enterprise specification, a list of engineering channel types and stub, binder or protocol objects, and interceptors that are required by the enterprise interaction; these rules are constrained by enterprise policies.

FIN77 TL 9, 9.1 Title, scope of discussion

Proposal

Remove subtitle 9.1 and use the title "Compliance, conformance and consistency" for clause 9.

FIN78 E 9.2, page 27, line 40 Spelling

Proposal

Remove words "of the".

FIN79 E 9.3 consistency with other comments made

Rationale

See FIN17.

Proposal

Remove clause 9.3.

FIN80 E 8 and 9 readability

Proposal

Swap clauses 8 and 9.

FIN81 TH Annex A consistency with the text

Rationale

Diagram needs redrawing. Problems include

- Relationships of objective, state, and role

- Party and owner being on their own
- C-object exporting something
- Scoping statement being not connected to anything
- Relationships of a system restricted to c-object, purpose and scope
- ….