# Knowledge base for an autonomic transport layer

Ernesto Expósito, Christophe Chassot, Michel Diaz

# Knowledge base for an autonomic transport layer

Ernesto Exposito[1,2], Christophe Chassot[1,2], Michel Diaz[1,2]

[1] CNRS ; LAAS ; 7 av. du Colonel Roche, F-31077 Toulouse, France
[2] Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077 Toulouse, France
{ernesto.exposito, christophe.chassot, michel.diaz}@laas.fr

**Abstract.** The accelerated development of Internet and mobile devices has lead to new QoS-demanding distributed applications and new QoS-providing communication services, particularly at the transport level. The diversity of transport services and underlying networks environments asks for a novel design of the transport layer, able to provide in a transparent and autonomous way the most adapted service to the application. With this aim in mind, this paper presents a methodology based on a model-driven architecture (MDA) approach specialized using ontologies. It provides a QoS ontology model integrating standard and QoS-oriented transport services, protocols and mechanisms aimed at characterizing the entities, concepts and relationships composing this complex domain. This semantic model is intended to facilitate the integration of future mechanisms and protocols extensions. It offers the required properties to implement the knowledge base of an autonomic manager enabling transport service discovery, selection and composition based on application requirements and network constraints.

**Keywords:** Model-driven architecture, ontology-driven architecture, transport protocols, autonomic computing.

## 1 Introduction

With the accelerated development of Internet and the diversity of mobile devices (smart-phones, tablets, netbooks and laptops), a new large family of networked distributed applications and communication services are available today. Multi-platform instances of a same application are available at home, at work, in our mobile devices or more recently within the Cloud. These applications present heterogeneous needs in terms of Quality of Service (QoS) mainly related with time, bandwidth and reliability requirements. Moreover, networked applications are constantly changing from high-speed and high-bandwidth networks (e.g. ADSL networks at home), to variable bandwidth and high delay networks (e.g. when operating over WiFi or 3G mobile wireless networks).

For traditional applications offering file transfer, web navigation or email functionalities, a fully ordered and fully reliable transport service such as the one offered by TCP over a Best-Effort network is well suited. However, time-constraint applications such as multimedia and interactive applications could prefer a partially reliable and partially ordered service able to offer a more suited lower end-to-end

delay. Likewise, current heterogeneous network environments lead to more complex scenarios from the transport layer point of view. Even if traditional protocols such as TCP perform well over classical wired IP networks, their performances are suboptimal under new network technologies. Actually, current high speed, wireless and mobile networks deeply modify the QoS characterization of the network layer by providing more complex service models in terms of bandwidth, losses, delay or jitter.

These new models of the current heterogeneous network layer, as well as the complex requirements demanded by the new generation application layer, have deeply impacted the traditional transport layer. Classical TCP [1] and UDP [2] protocols are not able to provide an optimal transport service in this new context. This explains the trends for transport protocols creation and extensions proposals during the last decades. Examples of the transport layer evolution are the specializations proposed by the IETF aimed at enhancing traditional transport protocols (e.g. TCP congestion control and error control extensions [3], TCP extensions for mobile wireless networks [4], TCP fast retransmission and recovery strategies [5], TCP satellite enhancements [6], UDP-lite [7], Reliable UDP [8], etc). Likewise, completely new transport protocols such as the Stream Control Transmission Protocol or SCTP [9] and the Datagram Congestion Control Protocol or DCCP [10] have also been developed. Recently, a new IETF group has been created in order to propose an important extension to TCP called MPTCP [11] in order to take advantage of the new network capabilities of end-terminals (i.e. multi-homing and multi-paths). We believe that the current diversity of transport services as well as the complexity resulting from the deployment of a particular transport protocol or transport mechanism over the different services provided by heterogeneous networks ask for a novel design of the transport layer. The next generation transport layer must be able to cope with the diversity and complexity of this new family of transport protocols. Moreover, current and future applications will only be able to take advantage of the most adapted and available transport service if they are able to efficiently interact with this advanced transport layer.

In this paper a methodology based on a model-driven architecture (MDA) approach aimed at guiding the design of such a new generation transport layer is proposed. The approach proposed is based on the specialization of the MDA paradigm by using ontologies (i.e. ontology-driven architecture or ODA approach). The main contribution of this paper consists in a QoS ontology model integrating standard transport services, protocols, functions and mechanisms aimed at characterizing the entities, concepts and relationships composing this complex domain. This semantic model is intended to facilitate the integration of future mechanisms and protocols extensions resulting from the transport layer evolution. This semantic model offers the required properties to implement the knowledge base of an autonomic manager enabling transport service discovery, selection and composition based on application requirements and network constraints. Furthermore, this model also integrates the required knowledge to guide self-managing strategies in order to cope with dynamic and heterogeneous environments.

This paper is structured as following. Section 2 presents the state of the art of model-driven architecture design approaches. Section 3 presents the ontology-driven architecture design of a knowledge base representing the different features of a QoS-oriented transport layer. Section 4 illustrates how this knowledge base may be used to

guide self-configuring properties of an autonomic transport layer. Finally, the conclusions and perspectives of this paper are presented.

## 2 Model-driven and Ontology-driven architecture

This section introduces the model-driven architecture approach aimed at guiding the design and development of complex and evolving systems such as the transport layer protocols. This approach guarantees the portability, the interoperability and the reusability of the final system. A methodology based on this approach and using semantic models will be proposed in order to design the architecture of a new generation QoS-oriented transport layer. In the next paragraphs, the model-driven architecture and the various models used to represent different abstract level views of the designed system will be presented. An extension to this approach named ontology-driven architecture will also be presented.

### 2.1 Model-driven architecture

The Model Driven Architecture or MDA approach is based on the separation of the specification of a system from its implementation in any specific platform [12], [13], [14]. Model-driven approach allows the use of models to understand, design, develop and maintain a system architecture. The primary goals of MDA are portability, interoperability and reusability.

The MDA approach follows a process based on abstractions by viewpoints. It means that a system can be modeled by abstracting a set of selected architectural concepts and structuring rules. In this way, simplified viewpoints of the system can be constructed. MDA specifies three viewpoints on a system, a computation independent viewpoint, a platform independent viewpoint and a platform specific viewpoint. Each one of these viewpoints is represented or specified using models.

- Computation Independent Model: a computation independent viewpoint focuses on the requirements of the system and its environment, hiding the details related to its structure and internal behavior. The Computation Independent Model or CIM represents this viewpoint. The CIM is also known as the domain model and is built using the semantic associated to the system domain.
- Platform Independent Model: a platform independent viewpoint focuses on the operation of the system, hiding the details related to specific platforms. This viewpoint should be the same for different platforms. The Platform Independent Model or PIM represents this viewpoint. Platform independence can be obtained by representing the system as operating in a technology neutral virtual machine.
- Platform Specific Model: a platform specific viewpoint results from adapting the platform independent viewpoint to specific details of platform. The Platform Specific Model or PSM represents a system at this viewpoint. A PSM combines the specifications in the PIM with the details of using a particular platform.

### 2.2 Ontology-driven architecture

The Ontology Driven Architecture or ODA has been proposed by the W3C in order to promote the use of semantic models or ontologies in the framework of the MDA methodology [15]. The use of Semantic Web technologies is intended to naturally extend the MDA framework by defining unambiguous domain vocabularies and by providing model consistency checking and validation capabilities [16]. Semantic web technologies are mainly based on implementations following the RDF and OWL languages specifications.

Ontologies can be used to represent services allowing declarative functionalities to be deployed, discovered and reused. The use of ontologies can facilitate software development by enabling discovering and composition of existing functions to provide a new functionality rather than construct a completely new solution.

ODA may also facilitate dynamic service composition by enabling the definition of semantic models integrating the agreements that software components expose via their interfaces. These semantic models should include preconditions, post-conditions, and invariant rules aimed at specifying the behavior of components and composition of components. ODA can be used to build semantic models aimed at supporting specification and design phases. These models can also be integrated within the system implementation by including components identification and descriptions and thus enabling discovering and reuse then during design-time and runtime. These ontology models capture semantic related to properties, relationships, and behavior of components. As an ontology is an explicit conceptual model with formal logic-based semantics, the descriptions of components may be queried or may be checked to avoid inconsistent compositions.

The large set of mechanisms and services available at the transport layer, make hard or even impossible to applications' programmers to select the most adequate service (or composition of mechanisms) during design-time, based on application requirements and all the possible network environments while integrating current and future transport solutions. This paper proposes to follow model-driven and ontology-driven architecture approaches to allow application programmers to design distributed systems able to use a common knowledge base enabling self-configuration and self-adaptation autonomic properties of transport services. Next section presents the CIM and PIM level models of a QoS ontology aimed at guiding the design and development of an autonomic transport layer.

## 3 QoS transport ontology for an autonomic transport layer

As previously introduced, the constant evolution of application and network layers has produced an important impact at the transport layer. As a consequence a large diversity of extensions and enhancements to the traditional protocols as well as the design and implementation of new transport protocols have deeply complexified the transport layer, making the selection of the adequate transport services a difficult task to be programmed at the application layer. For instance, traditional hard-coded

strategies for transport socket selection (e.g. static selection of UDP or TCP service) are not well suited anymore in this dynamic context.

A novel approach is required to easily integrate the dynamicity required for a transport layer of next generation. This new approach should facilitate the selection of services and could allow the dynamic deployment of the required transport mechanisms and functions. Due to the complexity related with the diversity of services, protocols, functions and mechanisms, an important effort of semantic characterization and representation is required. In order to apply the Model Driven Architecture approach, next paragraphs introduce a standard framework aimed at providing a CIM-level model for the quality of service. Based on this referential model, we have defined a PIM-level QoS ontology transport model integrating the semantic of transport requirements, services, protocols, functions and mechanisms for the next generation transport layer.

### 3.1 ITU-T X.641 QoS framework

The ITU-T recommendation X.641 has proposed a QoS framework intended to develop standards related to QoS in the area of information technology [17]. The X.641 framework provides definitions and inter-relationships between these definitions in order to supply a common context for defining, representing and expressing QoS. This framework introduces the concepts of service, service user, service provider, QoS characteristic, QoS requirement, QoS parameter, QoS management function and QoS mechanism. Figure 1 illustrates the main concepts introduced by X.641.

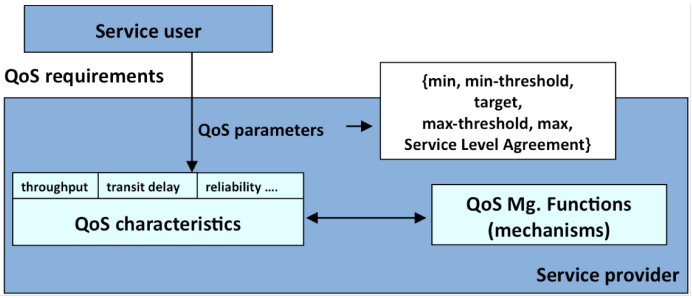


**Fig. 1.** ITU X.641 QoS framework

- Service: in the ITU-T X.641 framework, service is a very general term that can be applied to the provision of functions such as processing, storing, transmitting, delivering, etc. A service is provided by a service provider to a service user.
- Service user: a service is delivered to the service user. This user may have QoS requirements such as the maximum delay tolerated to transmit data. These QoS requirements are expressed as QoS parameters conveyed to the service provider.
- Service provider: a service provider is the entity responsible to deliver a service to the service user. The QoS parameters describing the QoS requirements of the

service user are conveyed to the service provider. The service provider analyzes the service user requirements and determines the management functions and mechanisms that are required to meet them. The QoS parameters can be conveyed to other entities involved in providing the service. These parameters could be used to produce more detailed QoS requirements to be conveyed to other entities.

- QoS characteristic: a QoS characteristic is defined as a quantifiable aspect of QoS of a system, service or resource that is defined independently of the means by which it is represented or controlled. QoS characteristics are intended to be used to model the actual, rather than the observed, behavior of the systems that they characterize. QoS characteristics definitions include name, description, quantification unit and optionally statistical derivations and specializations. Examples of QoS characteristics related to communication services are throughput, delay, jitter, order or reliability.
- QoS requirement: a QoS requirements expresses part of or all the user requirement expected on one or several QoS characteristics.
- QoS parameter: a QoS parameter is a vector of scalar values describing a QoS requirement in terms of:
  - A desired or target level of characteristic
  - A maximum or minimum level of a characteristic
  - Threshold values enabling warning or alert signals to be triggered or operations to be executed
  - A measured value, used to convey historical information
  - A service level agreement concerning the parameter. The term service level agreement is used to describe the nature of the commitment of the service provider to deliver the service required by the service user. The agreement nature determines the actions that the service provider and/or the service-users agree to take to maintain agreed levels of QoS:
    - Best effort is the weakest agreement indicating that there is no assurance that the agreed QoS will be provided.
    - Compulsory agreement indicates that the service must be aborted if the QoS degrades below the agreed level.
    - Guaranteed agreement indicates that the service provider must guarantee the QoS required by the service user, and that the service will not be initiated unless it can be maintained within the specified QoS parameters.
- QoS function: QoS management refers to the activities related to the control and administration of QoS within a system or network. QoS management functions are designed to assist in satisfying one or more user QoS requirements. QoS functions are composed by one or several QoS mechanisms.
- QoS mechanism: QoS mechanisms are intended to support establishment, monitoring, maintenance, control, or enquiry of QoS. QoS mechanisms are driven by users' QoS requirements expressed as QoS parameters. These mechanisms commonly operate in collaboration with other QoS mechanisms.

Based on these definitions, Figure 2 represents a CIM-level QoS ontology model of the X.641 framework. This basic QoS ontology will be extended in order to incorporate requirements, mechanisms, functions, protocols and service concepts from the transport layer point of view.

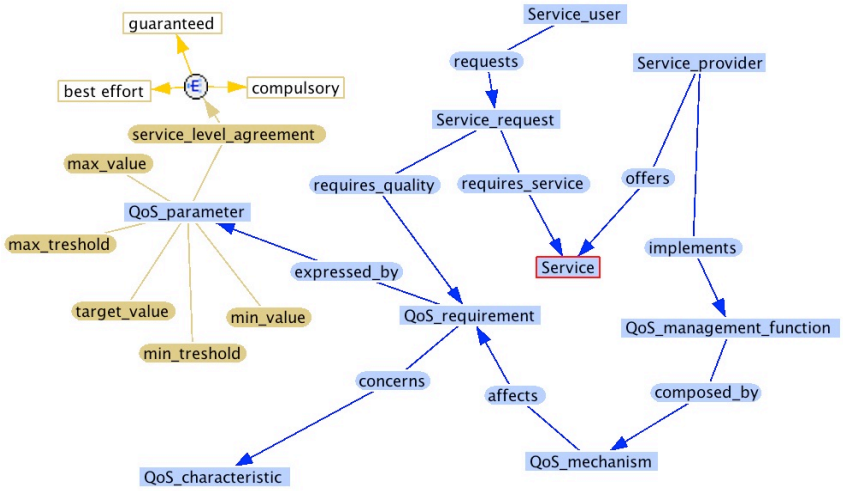


**Fig. 2.** QoS ontology model

### 3.2 QoS transport semantic model

Based on the previous CIM model, a QoS transport ontology integrating requirements, parameters, services, protocols, functions and mechanisms has been defined. This ontology is aimed at providing consistent transport layer semantic model aimed at enabling managing different levels of representation and validation. This model will be presented in the next paragraphs.

#### 3.2.1 QoS transport requirements

The expression of application requirements in terms of QoS parameters at the transport layer is built based on the following QoS characteristics:

- Reliability: packet loss rate (PLR) tolerance
- Order: out of sequence tolerance
- Throughput: transmission capacity per time unit
- Delay: end-to-end transmission time
- Jitter: variation of the delay

Based on these characteristics, QoS requirements can be expressed in terms of QoS parameters. For instance, for interactive video conferencing applications, examples of parameters expressions for QoS transport requirements are:

- Minimum and target values: reliability requirements could be expressed by a minimum value of 97% (or 3% of PRL tolerance) and a target value of 100%.
- Maximum and target values: delay requirements could be expressed by a maximum delay of 400 ms and a target value of 150 ms.

• Service level agreements: best effort agreements could be expressed for all or part of the requirements. For instance, best-effort agreement for reliability and compulsory agreement for delay (e.g. the service should be stopped when the delay exceed the maximum value).

### 3.2.2 QoS transport mechanisms, functions and protocols

Based on the RFC specifications of standard transport protocols, the various mechanisms implemented by TCP, UDP, SCTP, DCCP and MPTCP protocols have been integrated in the QoS transport ontology. Likewise, transport functions including basic functions (e.g. connection management, multiplexing/demultiplexing, etc.), advanced functions (e.g. multi-streaming and multi-path management), error control functions (e.g. ARQ or FEC) and flow and congestion control functions (e.g. window-based congestion control, TFRC, etc.) have also been integrated in this ontology. Likewise, the various mechanisms implementing these functions have also been incorporated. Figure 3 represents the QoS transport ontology integrating application requirements as well as the transport mechanisms, functions and protocols.

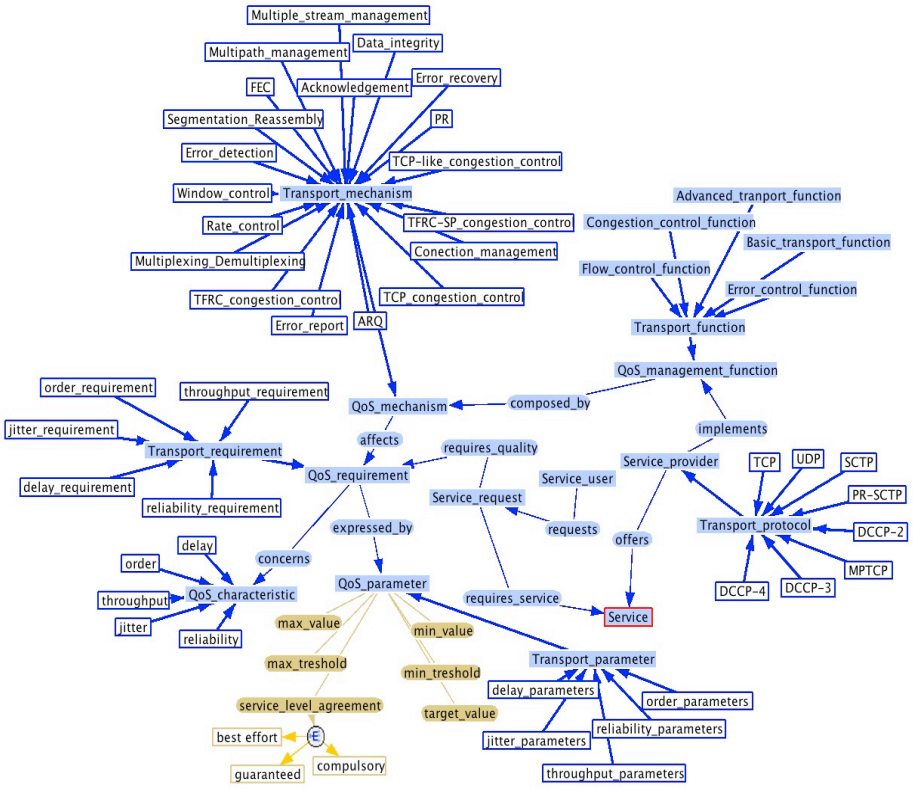


**Fig. 3.** QoS transport ontology model

## 4 Evaluation of the ontology-based autonomic knowledge base

Based on the previous QoS transport ontology model, this section is aimed at illustrating how a knowledge base, composed of this semantic model, can be used to guide self-configuring properties of an autonomic transport layer.

### 4.1 QoS transport services characterization

Based on the main functions provided at the transport layer, the following characterization of transport services has been defined:
• Error controlled: integrating fully reliable, partially reliable, fully ordered and partially ordered.
• Throughput controlled: includes congestion, flow and rate controlled services.
• Time controlled services: integrates delay and jitter controlled services. This class integrates the services implemented by QoS-oriented transport functions based on specializations of error-control and throughput control functions.
This classification allows to characterize the service offered by the transport protocols as following
• TCP: fully reliable, fully ordered, congestion-controlled and flow-controlled, time-uncontrolled
• UDP: error-uncontrolled (unreliable, unordered), throughput-uncontrolled, time-uncontrolled
• SCTP: fully reliable, fully ordered and unordered, congestion-controlled, flow-controlled, time-uncontrolled. As SCTP offers a multi-stream transport service, it can be considered as intra-stream fully ordered as TCP but inter-stream unordered service.
• PR-SCTP: partially reliable, fully ordered and unordered, congestion-controlled, flow-controlled, time-uncontrolled
• MPTCP: fully reliable, fully ordered, congestion-controlled, flow-controlled, time-uncontrolled
• DCCP-2, DCCP-3 and DCCP-4: error-uncontrolled (unreliable, unordered), congestion-controlled, time-uncontrolled.

Figure 4 illustrates this transport service classification for error-based and throughput-based service classifications. This service classification can largely facilitate the dynamic selection of the adequate transport service based on application requirements and following a service-oriented design approach. Moreover, this selection can be performed by integrating both functional and non-functional properties of the expected transport service. Furthermore, this semantic model can easily be enhanced in order to integrate future mechanisms and services. In this way, applications designed following this approach will be able to dynamically discover and use future services.

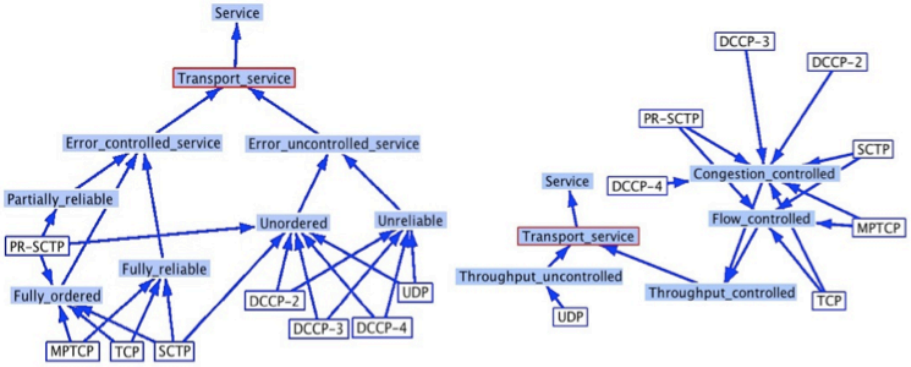


**Fig. 4.** Error-based and throughput-based transport services classification

### 4.2 Transport components and transport composite characterization

Most of the previously presented transport protocols are based on implementations where mechanisms offering different functionalities (i.e. error control or congestion control) are merged within a same monolithic implementation. However a component-based approach can be followed to characterize them. Figure 5 illustrates this composite approach in the representation of the TCP transport protocol functions.

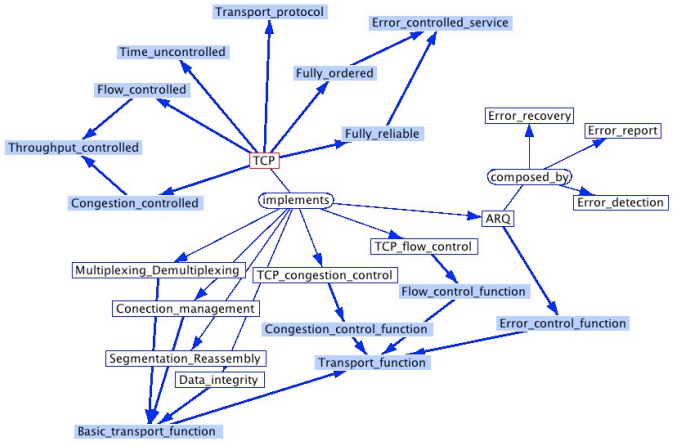


**Fig. 5.** Example of composite-based approach for the TCP transport functions

Actually, each transport protocols can be represented as the implementation of one or several transport functions. The composition relationship between functions has been integrated in this ontology. Likewise, transport functions can be represented as the implementation of one or several transport mechanisms. The composition

relationship between functions and mechanisms has also been integrated in the ontology. Figure 6 illustrates how the ARQ error control function and the TFRC congestion control function are implemented as a composition of mechanisms. Both functions share common transport mechanisms (i.e. error detection and error report). However, the specificities of each function are achieved by the addition of an error recovery mechanism for ARQ and a rate control mechanism for TFRC.

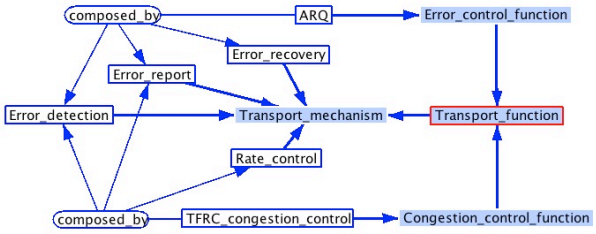


**Fig. 6.** Example of a component-based approach for ARQ and TFRC functions

The use of such component-based approach could widely facilitate the design and development of new transport protocols. Indeed, new transport services could result of the dynamic combination of pluggable components offering the services properties required by the applications. Further information about the classes, individuals and properties definitions of this ontology can be found in [18]. Likewise, the semantic description of standard protocols (i.e. TCP, UDP, SCTP, DCCP and MPTCP) including the assertions about their mechanisms and the inferences characterizing their services and aimed at verifying the consistency of this semantic model (verified using the Pellet reasoner) can be found in [18].

## 5 Conclusions and perspectives

This paper has presented a methodological approach based on model-driven and ontology-driven architecture design principles and aimed at building a knowledge base well suited to self-manage an autonomic transport layer.

In order to better characterize the diversity and complexity involved within the transport layer, a Computation Independent Model (CIM) providing an abstract and high-level service model has been presented. The QoS basis for the CIM model has been provided by the ITU X.641 framework. Based on this abstract model, a Platform Independent Model (PIM) of the transport layer has been elaborated and its semantic representation based on ontologies has been presented. This PIM model is intended to characterize and classify the large diversity of available transport services, protocols, functions and mechanisms. Likewise, this model provides an unambiguous transport layer vocabulary and enables model consistency checking and validation capabilities.

This ontology model offers the required semantic basis for managing different levels of representation and for integrating current and future services to be offered by the next generation transport layer.

The proposed methodology also provides the basis for developing a service-oriented and component-based architecture for transport protocols. Systems designed and developed following MDA and ODA approaches will gain major benefits in terms of flexibility and extensibility by integrating a service oriented approach. Indeed, Service Oriented Architecture (SOA) approach can be used for designing applications focused on services composition and coordination which is the specification level required by PIM models. Likewise, component-based approach facilitates the discovery and dynamic composition of reusable components, which can satisfy the service specification required for platform specific models. Current works targeting the use of this semantic model to design and develop the self-adapting functionalities of autonomic transport protocols based on the observed network conditions and the application preferences and requirements are being carried out.

## References


1 Postel Jon, Transmission Control Protocol, DARPA Internet Program Protocol Specification, RFC 793, September, 1981
2 Postel Jon, User Datagram Protocol (UDP), RFC 768, August, 1980
3 M. Duke, R. Braden, W. Eddy, E. Blanton, A Roadmap for Transmission Control Protocol (TCP) Specification Documents, RFC 4614, September, 2006
4 H. Inamura, G. Montenegro, R. Ludwig, A. Gurtov, F. Khafizov, TCP over Second (2.5G) and Third (3G) Generation Wireless Networks, RFC 3481, February, 2003
5 W. Stevens, TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms, RFC 2001, January, 1997
6 M. Allman, D. Glover, L. Sanchez, Enhancing TCP Over Satellite Channels using Standard Mechanisms, RFC 2488, January, 1999
7 L-A. Larzon, M. Degermark, S. Pink, L-E. Jonsson, G. Fairhurst, The Lightweight User Datagram Protocol (UDP-Lite), RFC 3828, July, 2004
8 T. Bova, T. Krivoruchka, RELIABLE UDP PROTOCOL, INTERNET-DRAFT, February, 1999
9 R. Stewart, Ed., Stream Control Transmission Protocol, RFC 4960, September, 2007
10 E. Kohler, M. Handley, S. Floyd, Datagram congestion control protocol (DCCP), RFC 4340, March, 2006
11 A. Ford, C. Raiciu, S. Barre, J. Iyengar, Architectural Guidelines for Multipath TCP Development, IETF Internet-draft, June, 2010
12 MDA Guide Version 1.0.1, OMG, June, 2003
13 Stephen J. Mellor; Kendall Scott; Axel Uhl; Dirk Weise, MDA Distilled: Principles of Model-Driven Architecture, Addison-Wesley Professional, March, 2004
14 Anneke Kleppe; Jos Warmer; Wim Bast, MDA Explained: The Model Driven Architecture: Practice and Promise, Addison-Wesley Professional, April, 2003
15 Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering, W3C Working Draft, April, 2006
16 O. Lassila, J. Hendler, T. Berners-Lee, The Semantic Web, Scientific American, May, 2001
17 ITU-T Information Technology - Quality of Service Framework, ITU-T X.641 (ISO/IEC IS13236), December 17th, 1997
18 QoS transport ontology, http://homepages.laas.fr/eexposit/hdr/QoSTransportOntology.htm