Journal of Mobile Multimedia, Vol. 2, No.3 (2006) 259-282
© Rinton Press

MODELING OF IMS CALL FLOWS: LOAD ESTIMATION FOR CORE NETWORK COMPONENTS

B. FALCHUK, D. SHALLCROSS, K.R. KRISHNAN, R. MORERA, S. LOEB
Telcordia Technologies, Inc.
Applied Research Lab
{bfalchuk, davids, krk, raquel, shoshi}@research.telcordia.com

Received March 7, 2006
Revised June 26, 2006

The support of mobile multimedia applications will require powerful control capabilities, such as those envisaged in the IP Multimedia Subsystem (IMS), which enables operators and service providers to control bearers, sessions and charging of multimedia services using Internet protocols. IMS enables operators and service providers to control bearers, sessions and charging of multimedia services using Internet protocols. IMS is seen as the main "enabler" to fixed-mobile convergence, as IMS provides services independently of the access technology and enables a smooth interoperation between different network types. Estimating the load and stress on core network components for IMS services is a requirement for the successful implementation of the IMS architecture. In this paper, we describe a formal procedure for characterizing a network service or application by means of annotated sequence diagrams, and deriving analytical models that allow us to investigate the load on different network elements imposed by the application. Our approach brings together software models that capture the semantics of the application and analytical models that describe the application in terms of states and state transitions. Our procedure is embodied in a software component and here, as a prototypical example of its utility, we apply it to the estimation of the load on the Serving Call Session Control Function (S-CSCF) for a Voice Call Continuity (VCC) Service between a GSM network and a WLAN - to demonstrate generality, we also apply it to an IMS compliant content distribution scenario. Such semi-automated results are critical in the planning and provisioning of IMS-compliant architectures, particularly those that support resource intensive mobile multimedia applications.

Key words:

1. Introduction

The IP Multimedia Subsystem (IMS) enables operators and service providers to control bearers, sessions and charging of multimedia services using Internet protocols. IMS is seen as the main "enabler" to fixed-mobile convergence, as IMS provides services independently of the access technology and enables a smooth interoperation between wireless and wireline networks, mobile and fixed networks and circuit-switched and packet-switched networks. A simplified IMS architecture is shown in Figure 1. IMS encompasses many components - the most salient to this paper include those listed below (other key definitions can be found in IMS, SIP and 3GPP specifications and other resources [1,2,3]):

- Call Session Control Function (CSCF) – CSCF provides various types of session control for users accessing IMS networks; a SIP stack is a capability at the core of its functionality. There are three different types of CSCF: Proxy, Interrogating, and Serving /Call Session Control Function/ ((P)(I)(S)-CSCF). The S-CSCF is the key component for session control, is in the signaling path of every session and inspects every signaling message. The S-CSCF downloads the user profile from the Home Subscriber Server (HSS). Based on the user profile and the SIP message headers, it selects the appropriate application server to which to forward messages.

- Application Server (AS) - a server running several applications in a reliable, scalable fashion.
- Media Gateway (MGW) and Media Gateway Control Function (MGCF) - a translation unit acting between disparate communication network types.

Estimating the load and stress on IMS core network components is more complex than in other service platforms, given the wide range of services that can be offered and the number of network components involved. Different services require different levels of involvement from the IMS components and different amounts of processing and bandwidth at each of these components. IMS services can be quite complex involving interworking between packet switch and circuit switched networks. IMS is widely seen as a service enabler; some of the services that are possible on IMS compliant networks include:

- Voice over IP (VoIP) and other voice and messaging applications
- Streaming Video services
- Content Mediation - managing authentication, billing, and rights management for various content interchanges, including those of a peer-to-peer nature
- Voice Call Continuity (see next section)

Therefore, more than in other systems, it is important for the operators, service providers and equipment manufacturers to be able to estimate the operational load that these components can support (or anticipated loads incurred by planned services) for a wide range of services and applications. The systematic approach and tools described in this document help to achieve this goal. The way service planning is done today is mostly an ad-hoc process using incomplete models of services and support systems. Deployment labs of large telecommunication providers (both traditional and triple play) do some simple calculations of the impact of a planned new service impact on network capacity followed by a lab prototype and some stress testing. This results in sub-optimal network utilization and CapEx investment and the inability to profitably acquire and retain QoS sensitive customers, establish brand equity, and provide billable value added services.

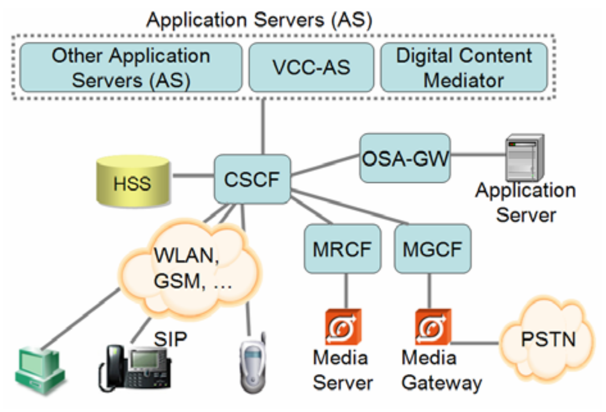


Figure 1. Simplified view of the relevant components of the IP Multimedia Subsystem

To describe our approach and tools in this paper we have chosen to analyze the Voice Call Continuity (VCC) “service”, though readers should note that this is not a one-off example and that the tools are intended to both a) generalize, and b) produce results semi-automatically. VCC allows voice callers to change access networks (from CS cellular to IP Access Network like WLAN and vice versa) in the middle of a call without service or call interruption. This service has recently been the focus of standardization efforts in several standard bodies and it is of great interest for the success of fixed mobile convergence (note the standard is not yet complete). The VCC Application Server (VCC-AS) is shown in Figure 1 together with a simplified picture of the rest of IMS components. Note that other SIP applications, such as digital content mediation to enable legal peer-to-peer transactions [4], are also embodied in applications servers and callable by the CSCF.

2. Application Models for Estimating Resource Requirements

2.1 Approach Architecture

The “approach architecture” we present in this section is a systematic way of modeling and evaluating network resource requirements of services and can be embodied in a semi-automatic software component. In Section 3 we use the approach to estimate the resource requirements on an S-CSCF in an IMS architecture that can support mobile terminals which might switch access networks during a session. Our approach brings together:

1) Software models that capture the semantics of application sessions, such as the Unified Modeling Language 2.0 (UML), proceeding from the specification of a sequence diagram of the events that could occur in a session of the application.
2) Analytical models specified by Markov chains that describe application sessions with respect to states and state transitions.

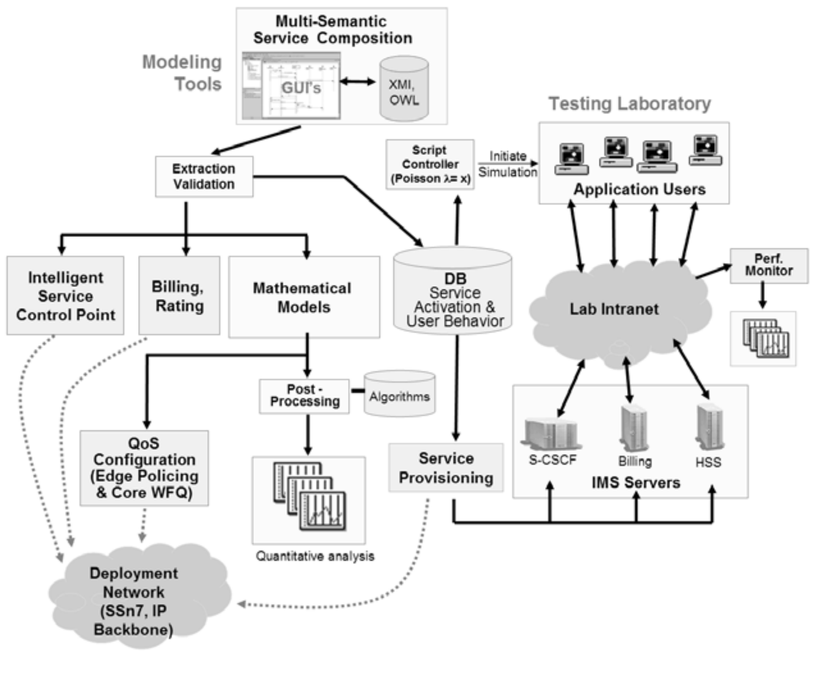
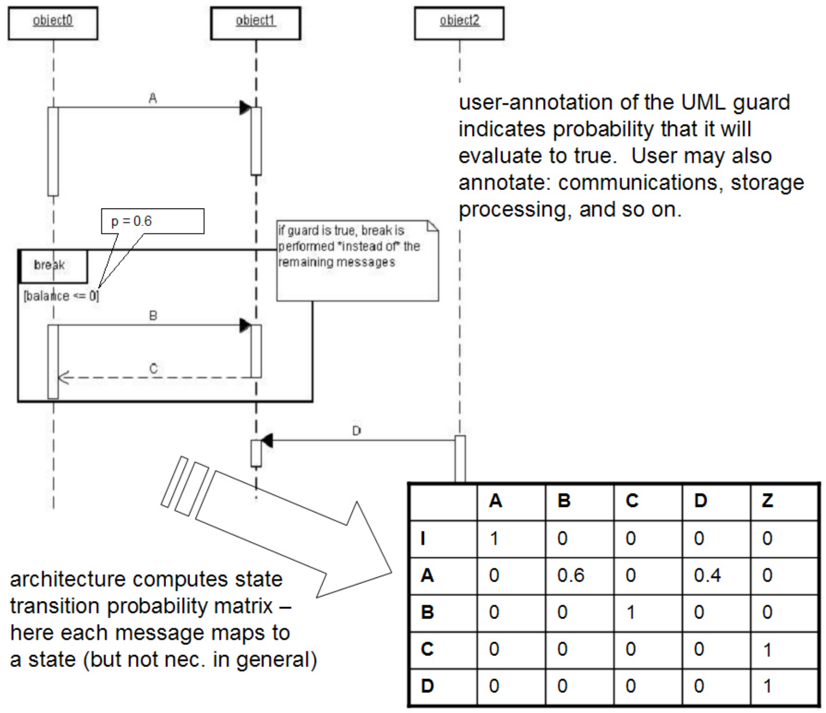


Figure 2a. Approach architecture in network operator environment


object0
object1
object2
A
user-annotation of the UML guard indicates probability that it will evaluate to true. User may also annotate: communications, storage processing, and so on.
p = 0.6
if guard is true, break is performed "instead of" the remaining messages
break
[balance <= 0]
B
C
D
architecture computes state transition probability matrix – here each message maps to a state (but not nec. in general)

	A	B	C	D	Z
I	1	0	0	0	0
A	0	0.6	0	0.4	0
B	0	0	1	0	0
C	0	0	0	0	1
D	0	0	0	0	1

Figure 2b. Approach for conversion to statistical model

The former are used by the ISP software engineers to define and describe services and protocols, and the latter by mathematicians and analysts. The main focus here is a software component that converts a sequence-diagram representation of an application to a finite-state *semi*-Markov process (a Markov process in which the sojourns in the states are not necessarily of exponential distribution) that describes the unfolding of events in a session of the service. By considering the resources required in each state, and by examining the probabilistic evolution of the states, we can estimate the resource requirements imposed by the service.

Figures 2a and 2b illustrate two aspects of the approach. In Figure 2a) a high-level architectural view of the approach is shown. In the figure, arrows show information flows and dependencies between steps and tools - components with dashed arrows connecting to the Deployment Network can be considered OSS's[a]. From a 500-ft view one should note that we envisage a connected-ness between service modeling, mathematical modeling, laboratory applications, and live systems (via OSS) and while we do not yet have quantitative evidence of optimality we have shown in our demo laboratory that this is a feasible and very useful practice. In particular, we have focused on the connected-ness

[a] Operations Support Systems (OSS) comprise functionality such as billing, network management, and inventory. Such systems are operated by, and essential to, all large communication service providers

between the service modeling phase, validation, mathematical modeling formation, and analysis. In general however, several useful component interactions are possible (but not yet implemented) including:

- a modeler, using an annotated sequence diagram tool, changes an annotation related to billing aspects. During validation, the system connects to the *Billing and Rating* OSS, and verifies that the change is possible, or ensures that the change does not conflict with existing billing rules. *Result*: The architecture has helped the modeler - at an early stage - ensure that certain billing violations are not "modeled in".
- a modeler, using an annotated sequence diagram tool, has already drawn the transaction diagrams for some service (e.g. say peer-to-peer content mediation). The Lab (testbed) deployment mirrors the scenario modeled in the tool. As the modeler changes the arrival rate of requests on the tool and saves it, the system routes the changes in such a way as the script controlling the execution of the testbed deployment is changed accordingly. *Result*: The architecture helps to maintain a sort of synchronization between the modeled service and the scripts controlling the prototyped service in the lab. In reality, having such a lab testing area for new services is common practice for large communication service providers.

One of the key parts of the overall approach (labeled "Modeling Tools" in Figure 2a) is the annotation of the sequence diagram. It has the following key steps: (1) The modeler (e.g. engineer or CTO) draws a valid UML2.0 sequence diagram(s) that corresponds to a typical session of a proposed service (e.g. a call set-up or handover as in Section 3). (2) The modeler uses a GUI to annotate individual artifacts on the diagram with instances of the pre-defined (but extensible) grammar. The resulting "annotations" comprise everything relevant (and probabilistic) that the modeler knows a priori about this service. (3) Through a number of sub-steps in the conversion to statistical model-Approach the system semi-automatically converts the diagram into a semi-Markov process and allows the analyst to obtain results on resource requirements to support the application.

2.1.1 Software Model: Sequence Diagrams

Our approach to estimating resource requirements for IMS is based upon existing standards, as well as on our past experience in XML, UML, structured schemata and taxonomies, content distribution and Operations Support System (OSS) issues [4,5,6,7].

The UML [8] is an Object Management Group (OMG) specification that helps software developers communicate, design and validate large, complex, software systems. UML Sequence Diagrams (SeqD), a type of timing diagram showing sequential flow of messages between system actors, are typically used to document how a system will behave in certain use-cases. The SeqD is a useful stepping stone in system design as it provides a somewhat formal and detailed view of system behavior and is easy to read. SeqD notation is widely used and understood by engineers and planners; however, no method exists for the systematic addition of non-deterministic annotation. For example, while one can draw a message arrow between a UE and S-CSCF, one cannot easily declare the processing, bandwidth, and storage implications of such a message. Typically, such declarations and analysis are done offline and/or in other, separate, tools. While UML admits a canonical XML format (called XMI), the expressivity of XML Schemata can be limiting [9]. To answer this the W3C Web Ontology Language (OWL) [10] is an application and extension of Resource Description Format (RDF) Schema (and others) and allows the creation of rich collections of classes, properties, individuals, and their interrelationships. OWL admits a description logics representation and, in turn, enables various types of reasoning over its assertions. Unlike XML Schema, reasoning over OWL may reveal new implicit assertions beyond what was explicitly declared. At its most minimal, the OWL syntax and tools allow rich expression of interrelated assertions and (sometimes highly abstract) concepts. In the large, several complex ontologies have emerged in medical and other fields[b]. We have used both XML Schema and OWL syntaxes in supporting roles.

2.1.2 From Sequence Diagram to Semi-Markov Model

The heart of the computing phase of our approach is the derivation of the Markov model and its parameters from the annotated sequence diagram describing the service session. The initial pass within this phase assigns a state to each labeled SeqD message (including its size, computation required, etc.). Subsequent passes derive computation, bandwidth, and storage events at each node and each link in each state. Using well-known methods (e.g., [16]), final passes may reduce the computed state space.

The following example (depicted in Figure 2b) gives the reader an idea of the transformation technique. Here the *break* fragment is analyzed - recall that *break* adds an alternative path within the SeqD followed by a transition to the end state (i.e. termination). We derive the nodes: object0, object1, object2, between which data flows. Since the transition from A to B is dependent on the annotated probability ($p = .6$) and the next independent message is D, the probability of transition from A to D will be $1-p = 0.4$. Additionally, the semantics of the *break* fragment require that the SeqD terminates after message C. Therefore C's next state is always the termination state. The figure illustrates the derived state transition probabilities where states I and Z are the initial and terminating states, respectively. In section 3 this Markov model derivation is applied to a "VCC user" scenario in which a multi-mode wireless phone moves between GSM and WLAN sessions; in section 5 it is applied to content distribution.

2.2 Ontology-based Markup

In order to make annotated diagrams automatically, or semi-automatically (i.e., assisted by some degree of human user input) convertible into models that can be analyzed on various dimensions, the set of annotations must derive from a well understood ontology. This section outlines a few of the underlying OWL concepts that form a basis for annotations; the case study of Section 3 elaborates on how it is used in practice.

Underpinning the ability for modelers (e.g. those drawing sequence diagrams) to annotate their diagrams with probabilistic information (i.e. distributions, delay bounds) not normally associated with sequence diagrams, is an ontology. Using the ontology, the modeler expresses the quantitative and probabilistic information that she knows about the service. Our ontology to date is comprised of about 60 classes and 60 properties and we used the Stanford Protégé tool to create and manage the ontology, HP Labs Jena2 Java API, and Altova *SemanticWorks* for publishing[c]. In the ontology, an *annotation*

[b] e.g., IEEE Standard Upper Ontology (SUO) (http://suo.ieee.org/), OpenGalen (http://opengalen.org)

[c] http://protege.stanford.edu/, http://www.hpl.hp.com/semweb/, http://www.altova.com

connects to a diagram element (e.g. a UML guard or message), and can be adorned by one or more *adornments*. Each *adornment* asserts something about the element's *computational*, *communication*, *storage*, *billing*, *security*, or *operational* aspects. Figure 2 depicts the annotation concept; that is, attaching rich metadata to diagrams in order to help mathematical model derivation. Figure 3a illustrates the "annotation" concept; note that any and all adornments must connect to a diagram element or diagram; each adornment may also have (a) probability of occurrence (i.e. messages on a diagram may occur with probability <1.0), (b) probability of advancement, (c) URI and human readable description. Figure 3b illustrates the *burst_communication* class (a subsclass of *communication*) which is used, for example, to annotate messages in a peer-to-peer (P2P) scenario where burst-type information is sent to a P2P mediation server. In such an annotation the user attaches a specification of the burst's *delay bounds* (choosing some temporal units), the *burst size*, and the probability that the communications *violates* the delay bound. Note also that in general, values (e.g. for burst size) may be discrete or take the form of probabilistic distributions which is one key differentiator of this work from the UML and other related tools.

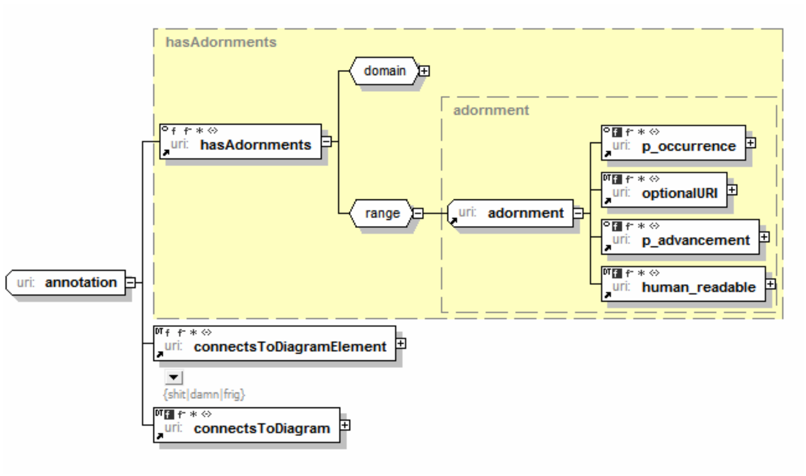


Figure 3a. Definition of annotation element

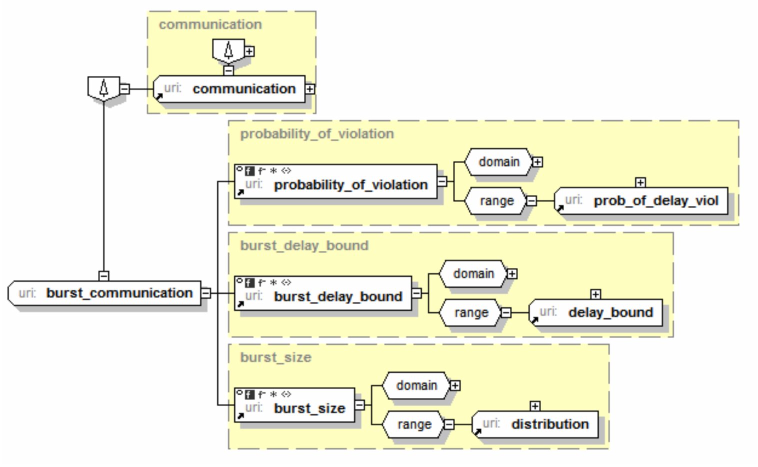


Figure 3b. Definition of burst communication subtype and its properties

In summary, the ontology provides a machine-readable representation of the abstract and non-deterministic information that is associated with the service in question. In addition, any annotation on a SeqD is a valid instance of an OWL ontology. The ontology concepts, once encoded and attached to SeqD via annotations, can be exploited by the system in many ways.

3. Case Study: Calculating Load on IMS Infrastructure

Although IMS control flows are complex, the UML2.0 specification allows the modeler to capture and notate nested flows and various other intricacies of sequence flows (see Section 2). As mentioned however, such UML specifications almost never encompass computing, storage or communication quantities. In this section we illustrate how a network planner can leverage sequence diagrams picturing IMS flows corresponding to session migration between WLAN and GSM technologies, use our approach, and better understand the processing load on key components - for example, the S-CSCF. It is important for the Network Operator (traffic engineer) to understand the processing loads on these servers in order to quantify the effects of various IMS signaling traffic. Given certain arrival rates of applications, such traffic could be non-trivial. Note that in this section's running scenario:

- It is convenient to imagine a user with a dual mode mobile phone - e.g. one that has both WLAN and GSM interfaces[d] (such phones are currently manufactured by several equipment providers)
- We depict a voice session that can switch between two networks- e.g., one that begins on a WLAN interface on the mobile phone, and subsequently moves to a GSM network. Teardown and setup happen as a result.
- We begin by considering a single "moving" session, and investigate the load it imposes on the S-CSCF. Then, we consider the situation corresponding to the sessions arriving at a certain rate into the network and examine the load as a function of the arrival rate. Using our model, the Network Operator can pose various "what-if" questions to the analytic engine

A basic tenet of the approach is to allow non-functional annotations upon sequence diagrams. In this section, the main annotation of interest is computational load. However, as noted in Section 2 other attributes can be attached to the diagram. Figure 4 illustrates this concept by showing a popup window that appears when a modeler chooses a sequence diagram element to annotate. For example, the modeler may choose the S-CSCF at the point where an incoming message arrives and annotate it such that the system understands that the incoming message causes x transactions to occur, where x is a normal distribution around 5, etc.

Figure 4 also illustrates the various dimensions of attributes that we are studying: computational (e.g. processing), storage (e.g. RAM and disk space), communications (e.g. message sizes), billing (e.g. costs of actions to operator and user), security (security requirements that need to be met)[e]. Note that as in the figure, the operator may choose to annotate computational aspects of the message; in this

[d] we will mostly consider roaming between WLAN and GSM. GPRS to GSM roaming (and vice versa) is not of great interest here

[e] billing and security are aspects currently under study

case the operator must supply units and a distribution to quantify such computation (e.g. a normal distribution, or simply a discrete value); values are entered values through input forms as needed. Underlying the pop-up options is a more formal representation of the allowable annotations and their interrelationships from which the user/modeler is somewhat shielded.

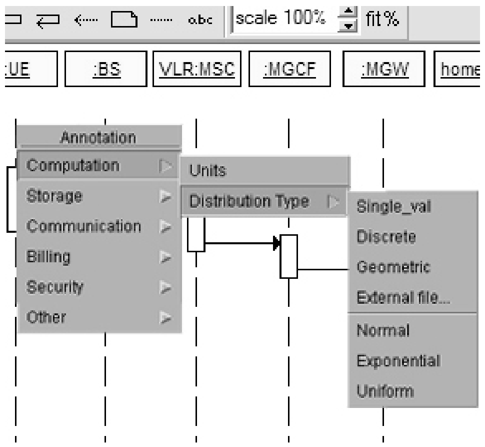


Figure 4. Context-sensitive graphical widget on the UML diagram that allows annotation of various diagram elements on various dimensions: this figure shows a user is in the process of specifying a computational value

3.1 *Application to Voice Call Continuity (VCC) Scenario*

Computing the load on the S-CSCF node from IMS flows involves, obviously, understanding the flow logic itself. Valid and useful flows in this section have been adapted from the various specifications describing IMS-related flows [1,2,3]. In the general scenario considered here, a mobile user initiates a call on one of the networks. During the holding time of the call, the moving user might switch back and forth between the two networks before the call ends.

The IMS signal flows that correspond to call initiation in the WLAN, possible migration to the GSM network, and tear-down (either because of the hand-over or because of call termination) are shown in Figure 5. The corresponding flows for call initiation in the GSM, with possible migration to the WLAN, are shown in Figure 6. A specific instance of the general scenario, for example, could be as follows:

1. User initiates a VoIP call using a WLAN interface (see Figure 5)
2. Call is setup and commences (see Figure 5)
3. User moves to GSM network (see Figure 5)
4. Call is migrated to GSM network (see Figure 6)
5. Call continues on GSM Network (see Figure 6)
6. Call terminates in the GSM OR user moves back to WLAN network and call is migrated (go to step 1)

For simplicity, we assume here that the destination of the call initiated by the mobile user is a fixed device. Also, the IMS flows for a call tear-down could be slightly different for a handed-over call and for a call that terminates. Similarly, the IMS flows for termination might depend on whether the termination is initiated by the moving user or by the fixed device. It is clear that all these variations can be handled in a straightforward manner by the introduction of additional states in the sequence diagrams. However, in the interest of simplicity of illustration, we shall assume that all these cases of tear-down involve the same set of flows.

Figures 5 and 6 illustrate fairly accurate and valid UML2.0 sequence diagrams describing IMS flows initiated by the steps 1-6 above[f]. In particular, Figure 6 illustrates the initiation of a VoIP call in an IMS-compliant WLAN environment. In it the User Equipment (UE) sets up a call using SIP via the Access Point (AP) and visited Proxy Call Session Control Function (P-CSCF). The SIP message is forwarded to the home network Interrogating-CSCF, that forwards the SIP message to the service CSCF (S-CSCF) where profile information had been previously downloaded from HSS. As the user is a VCC subscriber, the Initial Filter Criteria (iFC) stored in the S-CSCF indicates the SIP INVITE shall be forwarded to the VCC Voice Call Continuity Application Function (which might be a combination of several AS). The VCC Application function anchors the call and forwards the SIP INVITE to the S-CSCF. If no other AS is to be invoked for this SIP INVITE, the S-CSCF forwards the SIP INVITE across the data network through various Network Elements (NE) to the other user. The middle of Figure 5 shows the flow of data through the established session. The bottom of Figure 5 features a so-called parallel fragment that indicates that the two blocked in flows may happen in parallel. Concretely, the modeler of this UML is stating that for analysis purposes it is not important if we consider the WLAN session teardown as happening before after or during the setup of the new session on the GSM technology.

In Figure 6 an analogous IMS flow is depicted which represents the moving-to-GSM flow indicated in Figure 5 by the reference fragment called “Roam to GSM”. In other words, when the WLAN session is to move, this sequence of flows is kicked off. It depicts the flow of messages through the MSC, Media Gateway (MGW), I-CSCF, HSS, VCC and finally the data network to setup the GSM call. The session begins once it is setup (meanwhile the WLAN session may be terminated as shown in Figure 5). Note that the VCC does not play a role in the GSM session.

Once the interrelated sequence diagrams have been laid out by the Operator (e.g. the modeler), our tool and approach allow the annotation of various non-functional attributes of messages and actors on the diagrams. By clicking on the diagram the modeler has the opportunity to choose the annotation type (shown in Figure 4). For example, when focusing on the computation load of IMS flows on the VCC Server, the modeler may click on the focus of control bar which begins when the S_CSCF messages with VCC. The modeler would then quantify the computation by using the pop-ups. Indeed, Table 1 illustrates the types of diagram artifacts to which annotations can be attached using our tool.

[f] Operators can draw (almost) arbitrary UML sequence diagrams in this step. While complete accuracy with regard to the standard specification flows is usually desired, it may be the case that the operator is testing “what-if” scenarios by drawing alternative (but similar) flows for analysis. Flows shown in this paper strongly resemble those found in the specifications, but may be simplified - message names are mostly removed for clarity.

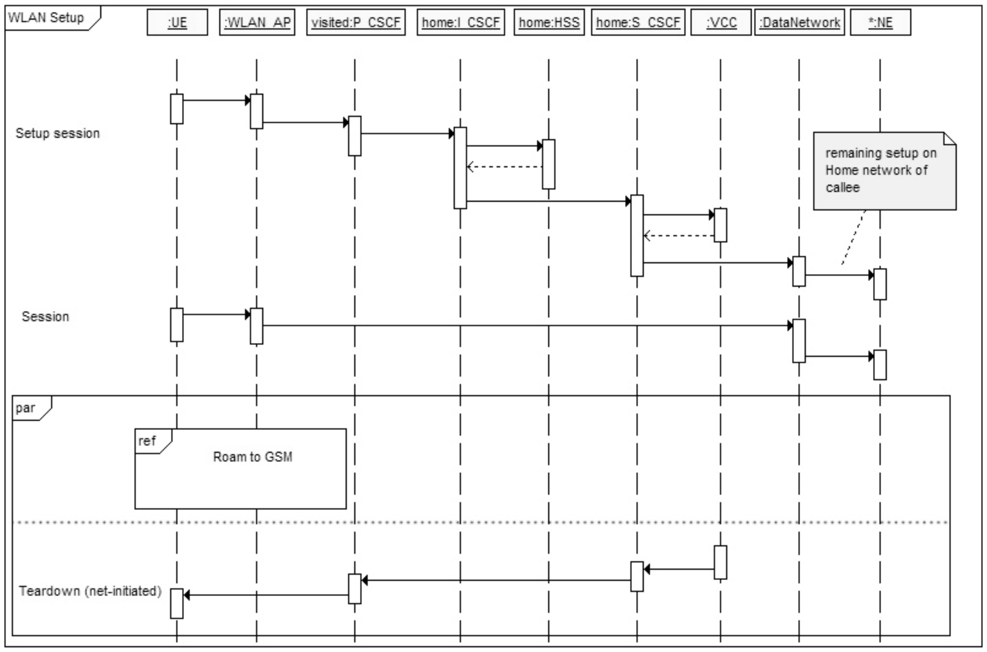


Figure 5. Sequence diagram drawn in the tool depicting WLAN VoIP session followed by a domain transfer to GSM. This diagram is ready for further quantitative annotation (which in turn allows derivation of the Markov model).

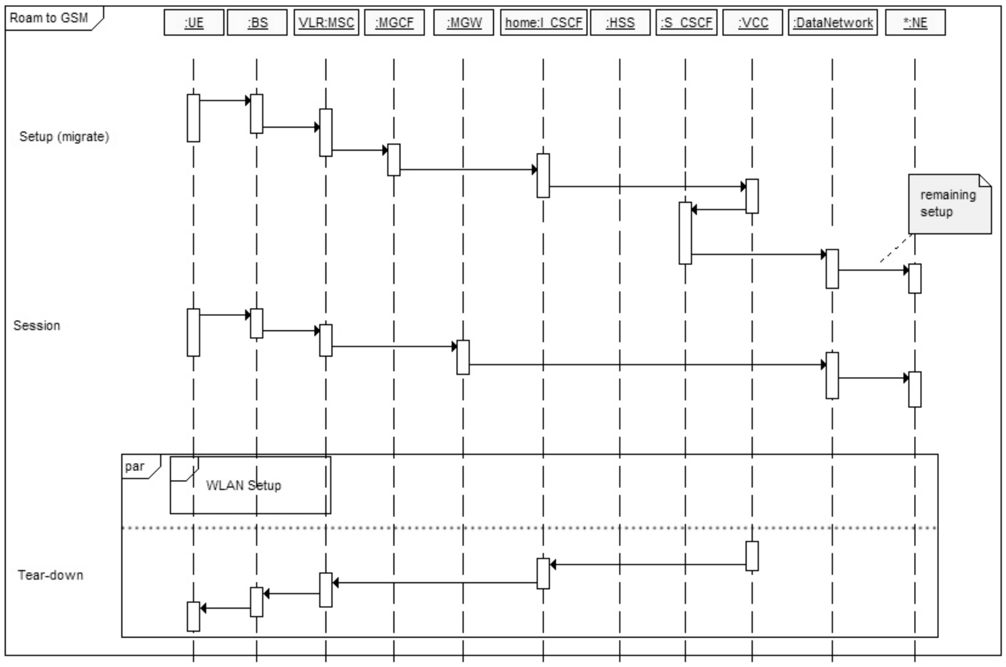


Figure 6. Sequence of flows represented by “Roam to GSM” in Figure 5. Depicts the setup and teardown or a GSM session handed-off from a WLAN session. This diagram is ready for further quantitative annotation (which in turn allows derivation of the Markov model).

Diagram Artifact	**Allowable Annotations and their Meaning**
Messages	• *communication subclasses* - a message may be annotated as either a burst or a fixed rate stream with regard to how it carries data. A message always has units. A burst communication has a size, delay bound, and probability of violating that delay bound. A fixed rate stream communication has a rate, time units (e.g. per sec), a duration (and units).
Messages, Endpoints / Foci of Control	A message always has at least two ends - a source and one or more targets. Each endpoint, or, the focus of control that either just precedes or just follows the message, may have associated properties • *storage subclasses* - a message endpoint or focus of control may have associated storage properties. Any description of storage declares storage units and a distribution of storage values (which may be a singleton value) • *computation subclass* - a message endpoint (or its associated foci of control) may be annotated with computation aspects. Any such annotation has units, and a distribution of computation values.
Guards	• *probability* - in composite fragments other than *loop,* guard conditions can be adorned with a *probability of occurrence*. In cases where multiple options are presented such as in *alt* fragments, the option probability adornment values should sum to one and any individual probability is in the range [0,1]
Fragments	Fragments can group any number of messages together. If the intent is to force the system to consider a fragment as a complete whole (rather than its pieces) then it may be annotated with any of: • *communication, storage, computation, probability* In addition, *loop* fragments can be annotated with: • *iteration* - indicating the number of iterations in the loop

Table 1. Annotation types

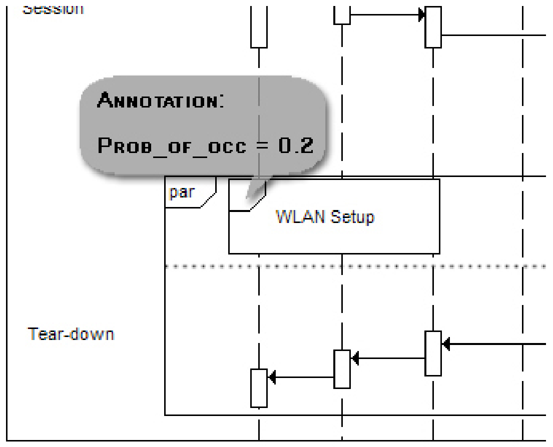


Figure 7. Annotating the probability that the control flow moves to GSM from WLAN (shown abstractly). This data may be a “what-if” value or may arise from the operators past statistics related to multi-network sessions

Figure 7 illustrates the notion of attaching probabilistic information to parts of the message flow from Figure 6 (GSM to WLAN moving). In the figure the operator indicates that the probability moving to WLAN from GSM is 0.3. Similarly, the probabilities of moving from WLAN to GSM and that of starting up in the GSM or WLAN network would be added (via the "Other" menu popup). Other information would likely include arrival rate specifications.

At this point in the "case study", the modeler has drawn or imported the UML sequence diagrams depicting the flows of interest. In particular, the modeler is interested in the effects of IMS protocols on a particular node (e.g. VCC or CSCF) so the sequence diagrams include prototypical flows containing these actors. The modeler has annotated the actors and messages with the non-functional data of interest. In this case we focus on computational requirements of the node. So, for example, the tool now understands that the computational load on the VCC after a particular message arrives is "1 transaction" (i.e., causes a single transaction to run). The next step is derivation of the Markov model and the analysis of the computational requirements using robust mathematical techniques. Again, note that this flow-through approach combining both the sequential and abstract quantitative information about protocols is something unique to our approach.

3.2 Semi-Markov Model for Voice Call Continuity (VCC)

Recall that we model an IMS use-case in which someone with a terminal device that can talk both to a WLAN and to a GSM access network. In this case the user initiates a call using one of the two access methods. The user continues the call for some length of time. During the length of the call, it may switch back and forth between the two access methods. Eventually, the call terminates. We are interested in determining the load on IMS components that results from various rates of call initiation, various rates of switching access methods, and various call holding times. Load shall be measured in numbers of "transactions per second"[g]. Our tool derives a semi-Markov model from the use-case Sequence Diagrams with ten states.

State 1. The user initiates a call on the WLAN. This is followed by state 2.
State 2. The user continues the call on the WLAN. This may be followed by states 3, 4, or 5.
State 3. The user switches the call from the WLAN to the GSM network. This is followed by state 7.
State 4. The user terminates the call on the WLAN. This is a final state.
State 5. The user's call on the WLAN is terminated from the other end. This is a final state.
State 6. The user initiates a call on the GSM network. This is followed by state 7.
State 7. The user continues a call on the GSM network. This may be followed by states 8, 9, or 10.
State 8. The user switches the call from the GSM network to the WLAN. This is followed by state 2.
State 9. The user terminates a call on the GSM network. This is a final state.

[g] This is a generic measure of load; more specific ones such as FLOPS (floating point operations per second) can also be used, if desired,

State 10. The user's call on the GSM network is terminated from the other end. This is a final state.

The parameters of the model, derived from user modeling, are:

- The rate of call initiation α;
- the probabilities $p_{0,1}$ and $p_{0,6}$, that a call will be initiated on the WLAN or on the GSM network;
- the transition probabilities $p_{2,3}$, $p_{2,4}$, and $p_{2,5}$, and $p_{7,8}$, $p_{7,9}$, and $p_{7,10}$, at the two states that permit branching (of the rest of the transition probabilities, $p_{1,2}=p_{3,7}=p_{6,7}=p_{8,2}=1$, and all others are zero); and
- the loads $l_{i,j}$ (measured in transactions) on IMS component j and the acceptable durations d_i for the states i that impose a load on the IMS components (1, 3, 4, 5, 6, 8, 9, and 10). We do not require the durations for states 2 and 7 for this study.

The values of these parameters are either taken from annotations in the sequence diagram or given as separate inputs for the scenarios to be studied.

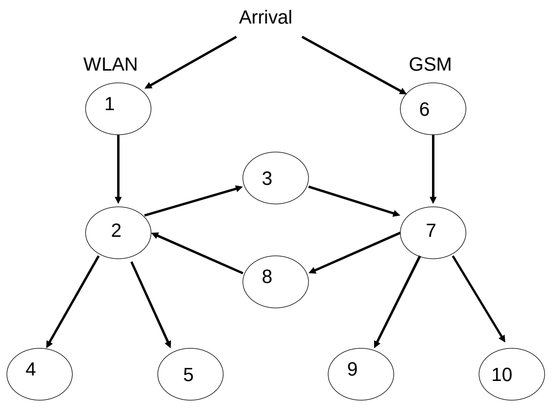


Figure 8. The states in the Markov model (partially) derived from the Sequence diagrams

From the transition probabilities, we can determine the expected number of visits v_i a call will make to state i before termination, using methods standard for a Markov chain. Under the assumption that calls are initiated according to a Poisson process, the number of calls in any state i at a given instant is a Poisson random variable n_i. With the v_i, the arrival rate α, and the state durations d_i we can compute the mean of n_i.

$$\mathrm{E}\, n_i = \alpha v_i d_i$$

With this and the loads we can compute the mean and variance of the transactions per second required at any given instant from component j.

$$\mathrm{load}_j = \sum_i \frac{l_{i,j}}{d_i} n_i$$

$$\mathrm{E}\left(\mathrm{load}_j\right) = \alpha \sum_i l_{i,j} v_i$$

$$\mathrm{Var}\left(\mathrm{load}_j\right) = \alpha \sum_i \frac{l_{i,j}{}^2}{d_i} v_i$$

Although the actual distributions of these loads area weighted sums of Poisson random variables, it can be useful to approximate them by normal random variables with the same means and variances. We then can easily compute, say, the 99th percentile, to indicate how much capacity the IMS system components need to be able to handle this kind of customer load reliably.

4. Computation of Load on S-CSCF

Here we illustrate the use of this model to compute loads on the S-CSCF at various arrival rates.

Suppose, in a region under study, calls have a probability $p_{0,1}$ of 0.7 of being initiated through the WLAN, and a probability $p_{0,6}$ of 0.3 of being initiated through the GSM network. Suppose a WLAN call has probability $p_{2,3}$ of 0.4 of being handed over to GSM, probability $p_{2,4}$ of 0.3 of being terminated by the mobile user, and probability $p_{2,5}$ of 0.3 of being terminated by the other end. Suppose a GSM call has probability $p_{7,8}$ of 0.2 of being handed over to the WLAN (see Figure 7), probability $p_{7,9}$ of 0.4 of being terminated by the mobile user, and probability $p_{7,10}$ of 0.4 of being terminated by the other end. Suppose, in addition, call initiations and handovers are each required to take no more than 1 second to progress through the S-CSCF.

state	l_i	d_i	v_i
1	4	1	0.700
2	0		0.826
3	2	1	0.330
4	1	1	0.248
5	1	1	0.248
6	1	1	0.300
7	0		0.630
8	4	1	0.126
9	0	1	0.252
10	0	1	0.252

Table 2. State Parameters

We can then populate Table 2. Loads (in transactions) are from the sequence diagrams and are for the S-CSCF. Durations (in seconds) are by our suppositions above. Average numbers of visits to a state per call are derived from the probabilities we have supposed.

For a call arrival rate of α, we compute a mean S-CSCF load of 4.761 α transactions per second, with a variance of 15.335α. We plot the mean, 99th percentile, and 99.99th percentile of the normal approximation to this random load as a function of the arrival rate in the Figure 9. The results show, for example, that to serve a call arrival rate of 20/sec, with a success rate of 99%, with the assumed probabilities of call-origination in the two networks and of handovers between them, the S-CSCF needs to be able to handle 136 transactions/sec. On the other hand, if we want to serve the same call arrival rate with a success rate of 99.99%, the S-CSCF needs to be able to handle 161 transactions/sec.

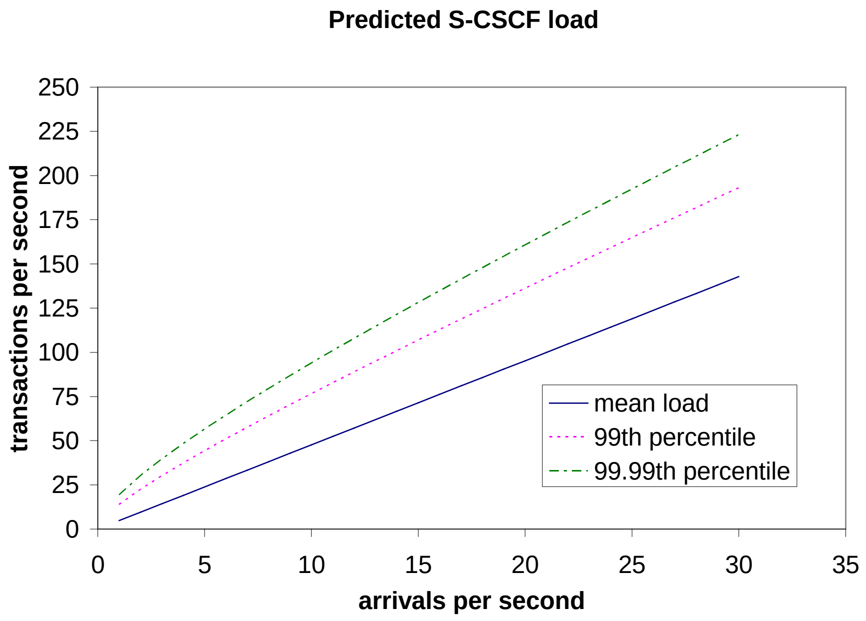


Figure 9. Load on S-CSCF as function of call arrival rate

5. Content-Authentication and Billing Services

As another example of the use of the Markov model for estimation of the network impact of IMS applications, we consider an IMS compliant content-based billing application [4] (it is useful to imagine that the session is initially initiated via a SIP INVITE). The particular example is that of the introduction of a back-end *mediator* for a peer-to-peer transaction. The transactions involves the direct downloading of a file from a 'source' peer by another peer, the authentication of content and peers, and the charging and billing of the peer(s) and content rights holder. Such a Digital Content Mediator (DCM) could be part of the function performed by a server in the box marked as "Other Application Servers" in Figure 1, acting as a sort of mediator between users: authenticating them, providing encrypt/decrypt keys, validating hashes, authenticating them, providing encrypt/decrypt keys, validating hashes, and interacting with backend billing and recording systems that carriers might already have, such as the Telcordia® ISCP® (Intelligent Service Control Point) and Data and Reports System® (DRS). From a model of the DCM transactions, we wish to estimate the total rate of the traffic between the CSCF and the DCM (situated in the box marked as "Other Application Servers" in Figure 1).

We consider a transaction in which a user (B) requests the download of a file from another user (A). The DCM accomplishes the tasks of client-authentication, content-certification, and billing in this transaction by the following sequence of message-flows:

1. A and B communicate the particulars of the desired file-transfer to DCM.
2. DCM communicates with DRM and with Billing System.
3. DCM sends A the encryption key for the file,
4. A sends DCM the hash of the encrypted file.

5. DCM acknowledges to A the receipt of the hash, and certifies the authenticity of the file to A and to B.
6. A sends B the encrypted version of the requested file to B.
7. B sends DCM the checksum for the received encrypted file.
8. DCM communicates with the Billing System to complete billing and credits.
9. DCM sends B the decryption key for the downloaded file.
10. The end of the transaction.

The detailed annotated UML sequence-diagram for this transaction is shown in Figure 10, in which the following symbols are used:

S = communication size (plus units)
DB = delay bounds for communication (plus units)
PV = probability of violation of delay bounds
F = storage requirements (plus units)
C = computing cycles (plus units)

For the DCM session, the following nodes are computed based on diagram and annotation parsing: A, B, Digital Content Mediator (DCM), Billing, Content Rights Holder (CRH), DRM (Digital Rights Manager). Node computation is relatively straightforward in most cases, simply reflecting each distinct actor at the top of the Sequence Diagram.

We now develop the corresponding semi-Markov model. Since the DCM session as drawn is completely deterministic, it translates into straightforward semi-Markov model. To construct the model, the system extrapolates a state for each message source, ignoring un-annotated and UML self-messages, since these do not imply communication (and can be combined with previous messages and states). A state is made for each message 1 - 17, except for messages 8 and 14. Therefore, states 1-7 correspond to the same named messages, states 8-12 to messages 9-13, and states 13-15 to messages 15-17. Note that Z is a terminating state. Given these deterministic states, the P_{ij} matrix of the semi-Markov model is essentially defined as: $(P_{ij} = 1, \text{ if } j = i + 1, \text{ 0 otherwise})$. In this case, there is communication only between one pair of nodes in any given state. Ultimately, six nodes, and ten states emerge (see Table 3). In fact, the ten states comprise a *reduced* space: since the analyst was interested only in network effects at DCM node, messages neither starting nor ending at the DCM can be disregarded (for this case only - e.g. we don't model the distribution of the size of the file transferred from A to B in state 6). Thus, the number of derived states is less than the number of messages on the sequence diagram. The roles are the source peer "A", the requestor peer "B", the DCM, the billing system, the content rights holder, and the DRM system. An instance begins after A and B have found each other, determined that A has a file B desires, and agreed to make use of the DCM. It then progresses through the ten states in order, without any branching. The message sizes given are as measured from our laboratory experiments. Note that, for the Poisson arrival rate α of transactions, user-input has specified a value of 300/sec.

The following questions are among those that our algorithms can answer:

- What is the average traffic to or from a given node?
- If we assign enough bandwidth for the transmission of each message so that it satisfies given delay requirements, what will be the distribution of the total assigned bandwidth into or out of a given node at any given instant? This will be a weighted sum of Poisson random variables, with a mean

given by the answer to the previous question, so this question is really “what is the variance of this total assigned bandwidth?” What fixed bandwidth would one actually *deploy* in this situation?

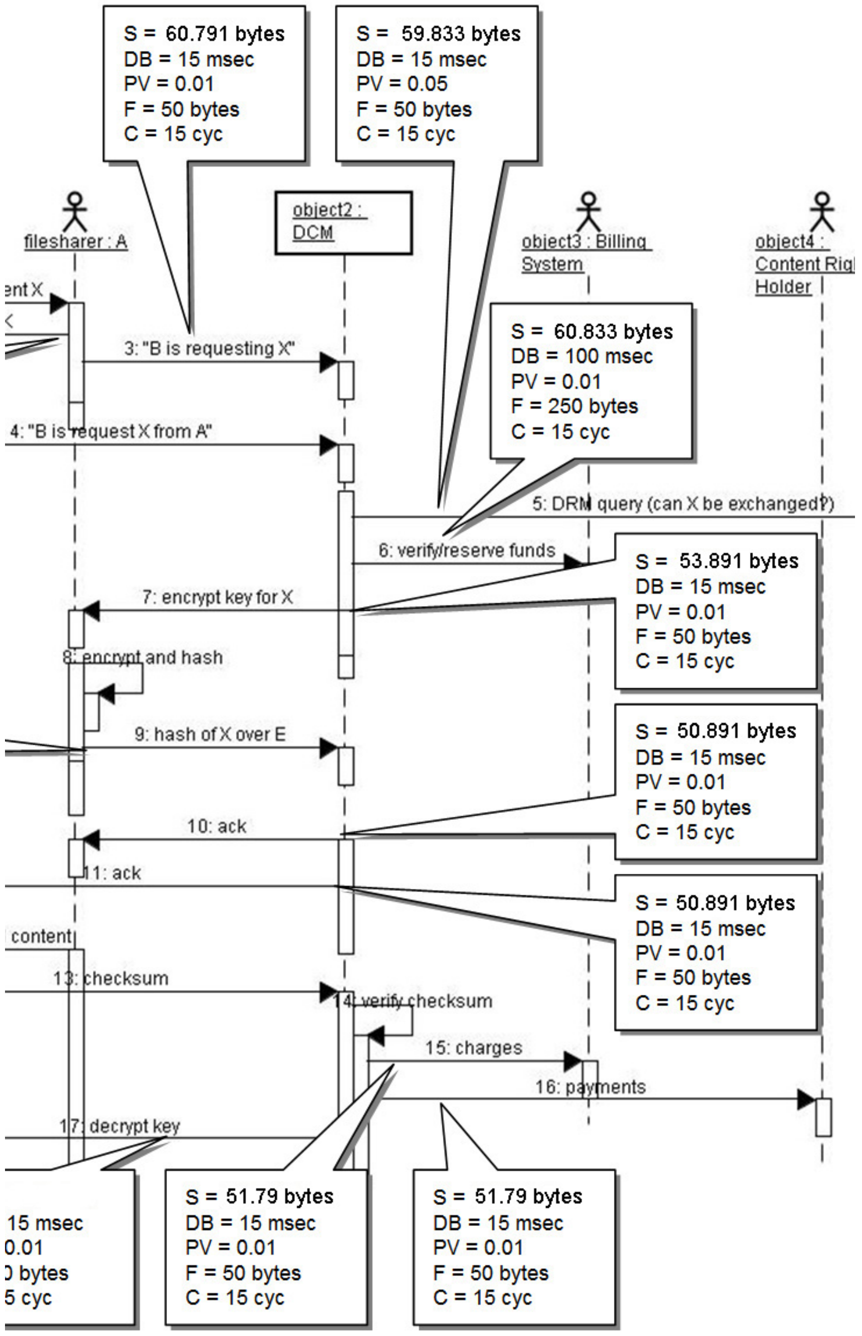


Figure 10. Prototypical application session of DCM during peer-to-peer file exchange; the sequence diagram is annotated with metadata (but simplified for clarity’s sake)

5.1 Laboratory Verification of the Model using the DCM Example

The accuracy of the average traffic rates and variances computed from our semi-Markov model-based algorithm was tested through lab experiments. A set of SUN workstations were used for hosting simulated clients (file requesters and providers) and servers (DCM Server, Billing System, Content Right Holder, and DRM System) in a Gigabit Ethernet LAN (see Figure 11). To facilitate the control of the file request rate α in the network, a single workstation is used for generating file requests*, i.e.,* all file requesters are co-located in the single workstation. The client and server programs were written in Java with thread-based processes. Traffic was monitored at the DCM Server using Ethereal to capture the throughput of incoming and outgoing messages. Message sizes were determined by measuring the average at the application level, and taking samples at the Ethernet level to determine the additional packet overhead. Message sizes were not constant – they depended on the sizes of the numbers contained in the messages – but they clustered closely around the reported averages. We ran the experiment for varied values of the arrival rate α. For each value, we gathered the time series of one-second average traffic rates into and out of the DCM. For each time series, we determined the overall average traffic rate, and the variance of the one-second averages, and compared this with the value predicted from our state model.

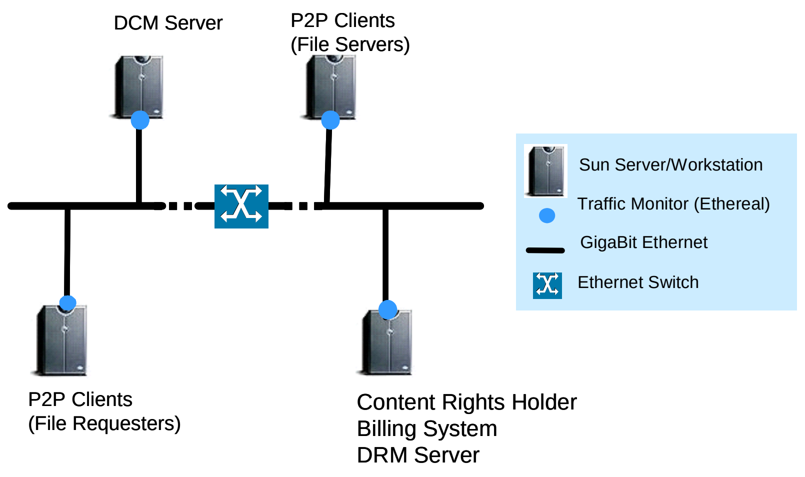


Figure 11: Experimental lab setup for DCM Server-based File-Exchange Application

The Markov model in Table 3 predicts the mean traffic rate *into* the DCM to be:

$$\alpha\left(m_{1a} + m_{1b} + m_4 + m_7\right),$$

and the mean traffic rate *out of* the DCM to be:

$$\alpha\left(m_{2a} + m_{2b} + m_3 + m_{5a} + m_{5b} + m_{8a} + m_{8b} + m_9\right).$$

Figure 12 compares the measured and predicted average traffic rates. There is a good match until the highest arrival rate, at which point we conjecture that the simulation was unable to actually generate arrivals according to the required Poisson process, which would have required a significant probability of successive arrivals in a time shorter than it took the code to generate them.

State	Message source	Message destination	Message size (bytes)	Delay bound (msec)
1	A	DCM	m_{1a}=60.791	15
	B	DCM	m_{1b}=50.893	15
2	DCM	DRM	m_{2a}=59.833	15
	DCM	Billing	m_{2b}=60.833	100
3	DCM	A	m_3=53.891	15
4	A	DCM	m_4=57.839	15
5	DCM	A	m_{5a}=50.891	15
	DCM	B	m_{5b}=50.891	15
6	A	B	variable	500
7	B	DCM	m_7=64.893	15
8	DCM	Billing	m_{8a}=51.79	15
	DCM	Rights Holder	m_{8b}=51.79	15
9	DCM	B	m_9=53.891	15
10	termination state – no messages			

Table 3: States and Messages corresponding to the DCM Transaction

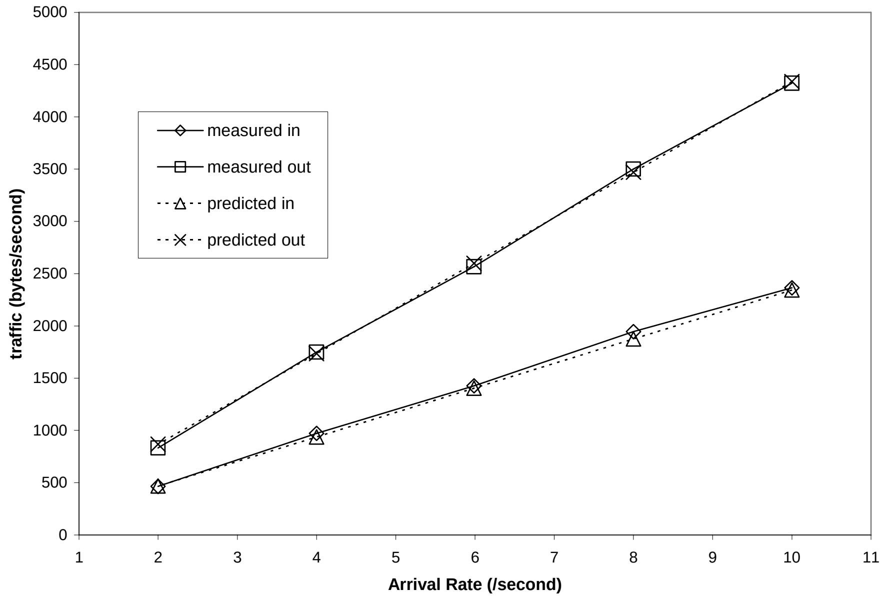


Figure 12: Measured and Predicted Average Traffic Rates

The exact value of the variance of the one-second traffic averages depends on how long an instance stays in each state, and on how long it takes to transmit each message. However, we can compute an upper bound, which should be a good approximation when the total duration of an instance is small in comparison to a second. For traffic into the DCM this upper bound is

$$\text{variance} \leq \alpha\left(m_{1a} + m_{1b} + m_4 + m_7\right)^2.$$

For traffic out of the DCM the bound is

$$\text{variance} \leq \alpha\left(m_{2a} + m_{2b} + m_3 + m_{5a} + m_{5b} + m_{8a} + m_{8b} + m_9\right)^2.$$

Figure 13 compares the measured and predicted variances of one-second average traffic rates. Here the match is not as good, but still in the right order of magnitude. The assumption of independence of behavior of the instances tends to fail as more instances are competing for network resources, and at times when many instances are running at once, they are slowed down by congestion, resulting in higher variances. The results of our lab experiments indicate that the predicted bandwidth requirements from the semi-Markov model-based algorithm are in acceptable agreement with our lab results.

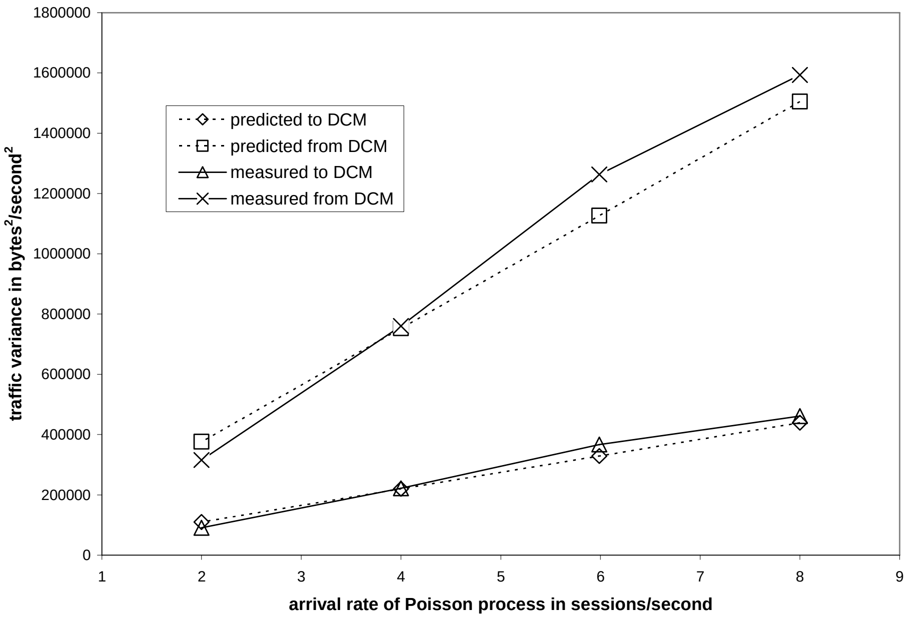


Figure 13: Measured and Predicted Variances of 1-sec. Average Traffic Rates

5.2 Choosing Bandwidth to be deployed

In the expressions above, we have estimates of the mean and variance of the inbound and outbound flow rates at the DCM. The randomness in the flows from the random number of sessions simultaneously present in the network in various states. Of course, for capacity provisioning, one has to specify a deterministic value for the bandwidth actually to be deployed. To do this, we approximate the distributions of inbound and outbound traffic flows by Gaussian distributions, with the means and variances shown there. We then choose a value for the bandwidth to be deployed, in each direction, such that the probability of the random flow-rates exceeding the chosen values is smaller than a specified acceptable value, say, ε.

6. Related Work

Past research and standards efforts have attempted to create languages for capturing mathematical expressions; some for (Web) presentation purposes, others for more rigorous purposes. MathML (see also OpenMath) is an XML-based markup language allowing the specification and embedding of math expressions in Web pages, as well as their exchange. MathML could play a role within annotation specification, if necessary. In [11] the authors extend the OWL language to support uncertainty at a fundamentally low level (lower then we need in this case); for example, their extended OWL can model imprecise subclass relationships. In [12] and others, researchers study the possibility of supporting a "global problem solving framework" through mathematical service descriptors based on XML and inspired by Web best-practices. This differs from our goal which is to provide a minimal (extensible) ontology to capture abstract and probabilistic attributes of messages and other UML artifacts. In [13][14][15] the authors study systemic methods for automatically generating UML state-charts from other types of UML diagrams, notably sequence diagrams. State-charts are a useful visual notation in the UML for describing the states that a single component or actor (as opposed to a service spanning several actors) goes through in its lifetime (start state through to end state). Transitions are deterministic, however, and the notation is not as widely used nor as deeply understood as that of the sequence diagram. Based on this, we have chosen the sequence diagram as the fundamental unit, attaching probabilistic metadata and then generating a probabilistic model.

Meanwhile, several companies provide network planning and prediction tools. While each performs their own roles reasonably well, a key difference is that none of them tie together a high level systems model, probabilistic conditions on service-level messages, and rigorous statistical analysis. In addition, our approach is model-based, while many commercial products take a traffic monitoring approach. These commercial tools analyze how an application performs across a network, with a focus on application-level delays; they also use traffic monitoring at the link level and sophisticated aggregation to provide insight. Some products for service design and test are available.

7. Summary

The IP Multimedia Subsystem (IMS) is designed to enable smooth interoperation between wireless and wireline networks, mobile and fixed networks, and circuit-switched and packet-switched technologies in a manner independent of the access technology. In particular, IMS is regarded as the main "enabler" for fixed-mobile convergence.

Estimating the load and stress on core network components for IMS services is a requirement for the successful implementation of the IMS architecture. This task of load-estimation is more complex than in other service platforms owing to the number of network components involved and the wide range of services that can be offered. Services can be quite complex, involving interactions between IMS-compliant and non-IMS networks (like the PSTN), with different services requiring different levels of involvement of the different components (e.g. applications that do not inter-work with the PSTN do not make use of MGF), and different amounts of processing and bandwidth at each of these components. Therefore, more than in other systems, it is important for the operators, service providers and equipment manufacturers to have a tool that can estimate the load that these components can support in an operational environment with changing traffic loads and for a wide range of services and applications.

In this paper, we have described a formal procedure - and an embodiment in a software component - for characterizing the session of a network service or application by means of annotated sequence diagrams, and deriving, from such a representation, a semi-Markov model for the session that allows us to investigate the load on different network elements imposed by sessions of the application. Our approach brings together two classes of models: software models that capture the semantics of application sessions, such as the Unified Modeling Language 2.0 (UML), and analytical models specified by Markov chains that describe application sessions with respect to states and state-transitions. We have applied the formal procedure to the estimation of the load imposed on the S-CSCF in an IMS infrastructure that supports the hand-off of a voice call between a GSM network and a WLAN, in terms of the transaction load of the individual events of the moving-scenario. In particular, such results are valuable in the planning and provisioning of IMS-compliant architectures that are required to meet specified performance levels under various traffic levels.

In future work, we expect to investigate other services and applications to gain an understanding of the nature of resource requirements imposed by IMS-based services. We also expect to apply this methodology to other mobile multimedia applications. These and other results continue to guide the development of future mobile multimedia networks.

Acknowledgements

We would like to thank Bryan Lyles for suggesting this area of research. We also thank Abdelhakim Hafid for his valuable input, and Mark Garrett and Stan Moyer for their helpful comments.

References

[1] 3GPP TS 23.206 V0.3.0 (2006-02). “Voice Call Continuity between CS and IMS; Stage 2 (Release 7)”

[2] 3GPP TS 23.228, “IP Multimedia Subsystem (IMS)” (Stage 2 Release 7), http://www.3gpp.org/ftp/Specs/html-info/23228.htm

[3] Session Initiation Protocol (SIP) RFC 3261, http://www.rfc-editor.org/rfc/rfc3261.txt

[4] B.Falchuk, D.Gorton, D.Marples, “Enabling Revenue-Generating Digital Content Distribution for Telecom Carriers”, *Proc. IEEE CCNC’06 - Digital Rights Management Workshop,* Las Vegas, 2006

[5] L.Bahler, F.Caruso, J.Micallef, “Experience with a Model-Driven Approach for Enterprise-Wide Interface Specification and XML Schema Generation”, *Proc. 7th IEEE Int’l Enterprise Distributed Object Computing Conf.* (EDOC 2003), Brisbane, 2003.

[6] D.Bassu, A.Jain, R.Zbib. “Automating XML Schema Specification from Data Models”, *Proc. Int’l Conf. on Software Eng. Research, Management, and Apps.*, Los Angeles, 2004.

[7] S.Loeb, B.Falchuk, M.Garrett, A.Hafid, K.Kim, K.R. Krishnan, D.Shallcross, “Impact of Services on Network Capacity: Tool for Seamless Integration of Service and Network Modeling”, *Proc. IEEE CCNC’06*, Las Vegas, 2006

[8] Object Management Group, UML specifications, http://www.uml.org

[9] T.Berners-Lee, J.Hendler, O.Lassila, “The Semantic Web”, *Scientific American*, 279(5), pp.34-43, 2001

[10] W3C Web Ontology Language Working Group (closed), http://www.w3.org/2004/OWL/

[11] Z.Ding, Y.Peng, “A Probabilistic Extension to Ontology Language OWL”, *Proc. IEEE Int’l Conf. on System Sciences”*, Hawaii, 2004

[12] O.Caprotti, “Towards a Mathematical Services Description Language”, *Proc. Int’l Congress of Mathematical Software (ICMS’02)*, Beijing, 2002

[13] I.Khriss, M.Elkoutbi, R.Keller, “Automating the Synthesis of UML Statechart Diagrams from Multiple Collaboration Diagrams”, *Proc. UML’98*, Mulhouse, June 1998

[14] I.Kruger, R.Grosu, P.Scholz, M.Broy, “From MCS’s to Statecharts”, *Distributed and Parallel Embedded Systems*, pp.61-71. Kluwer Academic Publishers, New York, 1999

[15] T.Ziadi, L.Helouet, J.Jezequel, “Revisiting Statechart Synthesis with an Algebraic Approach”, *Proc. IEEE Int’l Conf on Software Engineering*, 2004

[16] Z.Kohavi, “Switching and Finite Automata Theory”, McGraw-Hill, New York, 1978