Design and Prototype of a Train-to-Wayside Communication Architecture

Johan Bergs[1], Erwin Van de Velde[1], Daan Pareit[2], Dries Naudts[2], Milos Rovcanin[2], Ivan De Baere[3], Walter Van Brussel[4], Chris Blondia[1], Ingrid Moerman[2], and Piet Demeester[2]

[1] IBBT - PATS, Dept. of Mathematics and Computer Science, University of Antwerp
Middelheimlaan 1, 2020 Antwerpen, Belgium
{johan.bergs,erwin.vandevelde,chris.blondia}@ua.ac.be
[2] IBBT - IBCN, Dept. of Information Technology (INTEC), Ghent University
Gaston Crommenlaan 8 bus 201, 9050 Gent, Belgium
[3] Newtec Cy N.V., Laarstraat 5, 9100 Sint-Niklaas, Belgium
[4] Nokia Siemens Networks, Atealaan 34, 2200 Herentals, Belgium

Abstract. Telecommunication has become very important in modern society and seems to be almost omnipresent, making daily life easier, more pleasant and connecting people everywhere. It does not only connect people, but also machines, enhancing the efficiency of automated tasks and monitoring automated processes. In this context the IBBT (Interdisciplinary Institute for BroadBand Technology) project TRACK (TRain Applications over an advanced Communication networK), sets the definition and prototyping of an end-to-end train-to-wayside communication architecture as one of the main research goals. The architecture provides networking capabilities for train monitoring, personnel applications and passenger Internet services. In the context of the project a prototype framework was developed to give a complete functioning demonstrator. Every aspect: tunneling and mobility, performance enhancements, and priority and quality of service were taken into consideration. In contrast to other research in this area, which has given mostly high-level overviews, TRACK resulted in a detailed architecture with all different elements present.

1 Introduction

In the IBBT (Interdisciplinary Institute for BroadBand Technology) project TRACK (TRain Applications over an advanced Communication networK), one of the main research topics was the definition and prototyping of an end-to-end train-to-wayside communication architecture, which is also a research topic of the ESA ARTES project [17]. It is important to use a centralized approach for the communication system, connecting the onboard network transparently to the wayside, because it offers a more flexible and scalable solution. It allows for better coverage on board and joint bandwidth optimizations, traffic conditioning and differentiation.

The TRACK communication architecture can be used to bring passenger Internet on the train, but more importantly, to enable a much broader spectrum of railway applications, such as CCTV (Closed Circuit TV) surveillance, train control and diagnostics, etc. If these kind of operational services are to be supported alongside passenger entertainment, a mechanism must be deployed to provide sufficient Quality of Service (QoS) guarantees in the dynamic train environment. The main topics that were researched in order to tackle these challenges, are:

- Analyze the behavior of relevant wireless technologies (satellite, Wi-Fi, 3G) in the dynamic train environment in terms of variations in bandwidth, delay, jitter and bit error rate.
- Optimize the throughput by introducing Performance Enhancing Proxies (PEP).
- Investigate suitable IPv6-based solutions for:
  - End-to-end Quality of Service (QoS);
  - Mobility;
  - Link aggregation techniques and the associated policy decision functions.

This paper will first give an overview of the network architecture that has been designed in section 2. Section 3 will describe the most important components in the architecture in more detail and will comment on the design decisions that have led to that architecture. In section 4, some of the performance results, obtained from a functioning prototype of the network architecture are discussed. Finally, section 5 formulates our conclusions and addresses future work.

2 Network Architecture Overview

Research on the topic of Train-to-Wayside-Communications-Systems has already been extensive [6], but most of the described architectures have resulted in high-level overviews so far [9]. In the TRACK project an architecture has been worked out in great detail, describing and implementing all elements.

2.1 TRACK Architecture

In figure 1, a general overview of the TRACK network architecture is shown. The left side of the picture represents the network architecture on board a train. In the TRACK architecture, this collection of components is referred to as "MCE" or Mobile Communications Equipment. Each train contains one MCE, as can be seen by the fact that multiple MCEs are stacked in the picture, representing multiple trains. In the picture, each MCE is depicted with two uplinks. This is not a requirement, however; each MCE can be equipped with as many or as few uplinks as is required. On the right side, the "WCE" or Wayside Communications Equipment is shown. Only one WCE is deployed in a control center. However, the WCE contains a complete set of components per MCE, as indicated by the stacked components in the WCE. As can be seen in the picture, the WCE only has one Internet uplink, as we assume it is connected via a broadband Internet

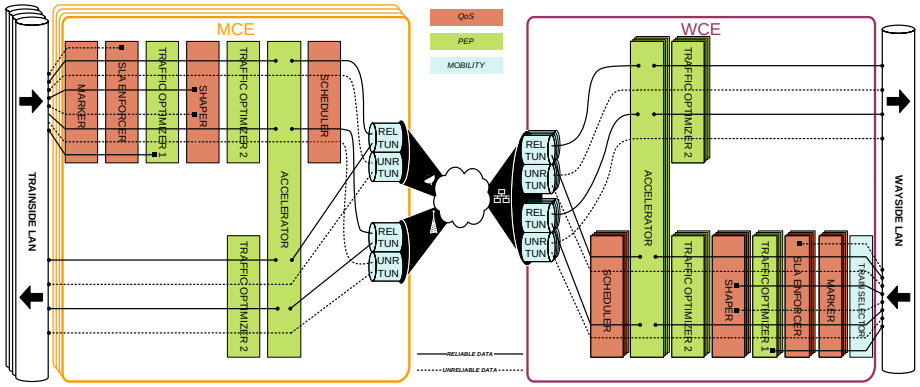


Fig. 1. TRACK network architecture overview

connection. The dimensioning of this Internet uplink is out of the scope of the work described in this paper.

Figure 1 uses a color coding scheme to indicate the function of each individual components: red components are responsible for QoS functions, green components offer a performance enhancing function and the blue components take care of mobility. Furthermore the TRACK architecture was designed with an end-to-end IPv6 based network in mind.

2.2 Implementation

Prototype Framework One of the goals of the TRACK project was to have a functioning prototype version of the network framework by the end of the project. This prototype has been implemented using the Click Modular Router framework[8]. All of the components listed in figure 1 were implemented using one or more Click elements. Most of the elements were standard, while some of them have been custom made for the TRACK framework.

In order to design and implement the TRACK architecture, a number of existing mobility/addressing/multi-homing protocols have been investigated. For the performance enhancing elements, existing solutions have also been reviewed. Wherever possible, existing solutions or standards are preferred over custom solutions in order to maximize compatibility with existing systems.

Tunneling & Mobility To support network mobility and multi-homing, which is required since every train has multiple uplinks that are not necessarily under the control of the train operator, some mobility solutions that have been taken into consideration are: Mobile IPv6[14], HIP[10], SHIM6[12], NEMO[5] and SCTP [19]. An important factor that was taken into account was the requirement that the end devices, whether they are passenger smartphones or diagnostic devices on board the train should not have to be modified in order

to support mobility. Moreover, the mobility solution deployed needs to ensure that connections from an onboard device to the Internet are not affected by mobility and that the mobility solution should not break any of the optimizations introduced by the performance enhancing elements.

Performance Enhancements The performance enhancing elements in the architecture can be divided in two classes: those elements that perform local optimizations (i.e. they can operate at the WCE or the MCE separately) and those that can only function if they are deployed both at the MCE and at the WCE. Performance enhancements can be made in numerous ways, such as deploying a local http proxy server (e.g. squid[18]) or a local DNS proxy; two examples of purely MCE-based proxies that do not require a WCE equivalent. Another possible performance enhancement is reducing the bandwidth required to transfer a given amount of data by installing a ZIP proxy[21] or employing robust header compression techniques like those described in [7], [4] and [3], which do need a counterpart at the WCE. A final optimization technique that has been considered, is the use of a TCP Performance Enhancing Proxy (PEP) such as SCPS[11].

2.3 Priority and Quality of Service

The architecture works with both the concept of priority and QoS. Every service type is categorized with a priority and a QoS class, thus it is possible that a service type has a high priority and a low QoS class or vice versa.

Priority	**Services**
1 (low)	Passenger Internet
2	Crew Intranet
3	Diagnostics
4	Application update
	Content Update
	TCMS Event
5	CCTV (security)
6	Intercom (VoIP)
	CCTV (safety)
	TCMS cyclic
7	Public address
	PIS data
8 (high)	Configuration traffic

Table 1. Priority Definition

The priority level determines the importance of the service type: the higher the priority, the more important it is that data of this service type is sent through,

the lower the priority the sooner packets from this service will be dropped when the available bandwidth is less than the needed bandwidth. Table 1 gives an overview of the eight different priority levels and which types of services are .

Class	**Delay**	**Jitter**	**Loss**	**Services**
A	< 1 s	-	-	Passenger Internet Crew Intranet Diagnostics Application update Content update
B	< 0.5 s	-	-	TCMS event
C	< 1 s	-	$< 1 \cdot 10^{-3}$	CCTV (security)
D	< 0.07 s	< 0.016 s	$< 1 \cdot 10^{-2}$	Intercom (VoIP)
E	< 0.2 s	-	$< 1 \cdot 10^{-2}$	CCTV (safety)
F	< 1 s	-	$< 1 \cdot 10^{-6}$	TCMS cyclic
G	< 1 s	< 0.1 s	$< 1 \cdot 10^{-2}$	Public address PIS data Configuration data

Table 2. QoS Definition

On the other hand, the QoS class defines the requirements for a type of service to function properly. It does not take bandwidth into account, but does define bounds for delay, jitter and packet loss. We follow the IETF Differentiated Services (DiffServ) architecture[2] for traffic classification. Table 2 gives an overview of the different QoS classes that are used in our architecture. QoS and priority are not mapped one-on-one as such. However, if due to priority restrictions a flow with QoS class C-G encounters too much packet loss, that flow will be dropped because its QoS constraints can no longer be met. Similarly, even if there is enough bandwidth, but on links with delays and/or jitter higher than the delay or jitter allowed by the QoS class of the service, the flow cannot be set up.

3 Component Description and Design Decisions

As you can see in figure 1, the architecture is symmetrical for trainside and wayside, except for the fact that there is only one wayside network operating center for all trains. The components also fulfill the same functions on both sides of the architecture. In this section, the different components will be discussed and also the design decisions that were made in the project concerning these components. The components which handle mobility will be discussed in greater detail as they form the main topic of the paper.

3.1 Quality of Service Components

The Marker The Marker is the first element in the chain for outgoing packets and is used to determine

- The flow ID of the packet
- QoS class of that flow
- Priority of that flow

A triple source IP address, destination IP address and Flow label in the IPv6 header define a flow uniquely and in our architecture the marker sets the flow label if the source did not do so. More information about the assignment of the flow label can be found in [13] and a description of the different QoS components and there implementation is worked out in [16] in more detail than in what follows.

The SLA Enforcer The SLA (Service License Agreement) Enforcer will shape traffic flows according to the applicable SLA and drop or reduce maximum throughput for those flows for which the SLA cannot be met (e.g. when data volume limit has been reached). Thereafter the SLA Enforcer shapes all the flows of the same SLA together. The SLA is specified by VLAN tagging.

Shaper The Shaper shapes the flows when there is more data to be sent than there is bandwidth available on the link. Different flows are handled appropriately considering flow priority and link occupation. The shaper receives the link occupation and the link-flow mapping from the Scheduler. If necessary, the Shaper drops packets with higher probability for low priority flows while trying to prevent starvation and to maintain a fair share policy for flows of equal priority.

Scheduler The Scheduler decides which link to put the traffic on, based on matching service classes of the flows with delay and jitter properties of the link. The Scheduler also informs the Shaper of its decision, because the Shaper needs to know which flows belong to a link when it becomes overloaded. The Scheduler is based on Active Queue Management (AQM) [1], which means that there are two thresholds used for the queue: below the first threshold, the Shaper is permitted to allow more traffic going out. Between the first and second threshold, the Scheduler signals the Shaper to start dropping packets gradually. Starting on the second threshold, it is even possible that the Scheduler alerts the Marker to drop flows entirely to make sure that the most important flows are still coming through, meeting there QoS constraints.

The decision for the flow-link mapping is made based on the first packet and all consecutive packets of that flow are put on the same link to avoid jitter, unless the link goes down. In that case, the flow is rescheduled on a link that is still available.

3.2 Performance Enhancing Proxy (PEP)

The trains are connected to the wayside by means of costly media like 3G and satellite connections. Therefore it is important to optimize bandwidth utilization, which is taken care of by the PEP in the system architecture. In this section, the different components of the PEP will be discussed: two traffic optimizers and an accelerator.

Traffic Optimizer 1 As a first optimization along the way, the Traffic Optimizer 1 (TO1) tries to decrease the load on the wireless links by

- Caching information responses locally for requests it sees
- Responding with the cached information to future requests

Most of the functionality of the TO1 is situated on the application layer of the ISO OSI model [20] and includes caching proxies like web proxies, DNS caching, SMTP proxies. The caching does not only mean less load on the wireless network, but also a gain in response time for the user.

Traffic Optimizer 2 The second Traffic Optimizer (TO2) uses another technique to reduce the actual used bandwidth. The sender compresses the date, reducing traffic volume. The data is subsequently decompressed on the receiving end, so the eventual receiver perceives the data flow as unaltered. Compared to TO1, which tried to avoid sending traffic over the network by performing local caching, TO2 tries to minimize the amount of data sent over the network, reducing the bandwidth needed.

It is important to keep in mind that not all traffic should be taken under consideration for TO2: while e.g. HTTP and FTP traffic can be compressed well, data that is already compressed or appears to be random, like encrypted data do not get a benefit from TO2 and small packets like e.g. NTP protocol that are delay sensitive should not incur extra delay by compression.

Accelerator Several long delay wireless connection types are used, especially satellite connections. It is well known that TCP does not react well to big bandwidth-delay products and therefore we have introduced the Accelerator in our architecture. It tries to solve the problem by

- Mitigating performance degradation resulting from the large round trip time between MCE and WCE
- Mitigating performance degradation resulting from multiple competing TCP connections on such links.

The Accelerator does this by breaking up one end-to-end TCP connection in three different connections: one TCP connection between the host on the train and the Accelerator there, then a non-TCP connection between the Accelerator on the train and the one at the wayside and thereafter a TCP connection again between the Accelerator on the wayside and the destination.

Both TCP connections are terminated at the Accelerators to give a perception of faster round trip time to the end hosts, allowing the TCP windowing mechanism to open the TCP window faster. Between the two Accelerators however, the data flow is converted to a "stateless TCP" stream, keeping the source and destination addresses and ports, but removing the congestion mechanisms of TCP, preventing competition between different data streams. The receiving end of the Accelerator sets up a new TCP connection to the destination of the packet and conceals the fact that the connection has been split by the Accelerators. In contrast to other TCP accelerators, this Accelerator is thus distributed between a sending and receiving module.

3.3 Network and Mobility

The last components have should be discussed, are the components that connect the train and the wayside to each other over the Internet, taking into account the (network) mobility on the moving train. To accomplish this, the TRACK architecture uses Reliable and Unreliable Tunnel elements, which will be discussed in this section. To set up these tunnels, the architecture uses SCTP (Stream Control Transmission Protocol)[19]. The advantages of using SCTP as a tunnel protocol can be described as follows:

SCTP can bundle reliable and unreliable streams into one SCTP stream with only one overall congestion control mechanism. This removes any contention between different streams. In a more standard network situation, it often happens that unreliable (UDP) traffic pushes the reliable (TCP) traffic away.

In many architectures, like e.g. VPN solutions it is sufficient to use an unreliable tunnel, since the reliable traffic inside the tunnel will take care of any packet loss, as it would do without the tunnel. It would even lead to performance degradation if reliable data traffic was put in a reliable tunnel. In the case of the TRACK architecture however, the Accelerator removes the reliability of the reliable connections between the accelerators at MCE and WCE. However, for both end points of the transmission, the connection should still be reliable or connections will break abruptly in the case of packet loss. Therefore, we need to re-introduce reliability. Since the SCTP protocol supports sending data reliably, packets that have been accelerated by the Accelerator will be transmitted in a reliable SCTP stream. SCTP will thus take care of any retransmissions and resequencing of packets. For unreliable traffic, there is no need to tunnel it reliably. SCTP offers the possibility to send data in an unreliable SCTP stream [15]; both reliable and unreliable streams are managed by a single SCTP connection, with a single congestion control mechanism.

In order to support mobility, the following scheme is used: one SCTP tunnel per link is set up by each train. Since the connection setup originates from the train to a fixed IP address on the wayside (the IP address of the WCE), no further mobility solutions are needed: the connections that are set up are the SCTP tunnel connections. Any data originating from a device on the train or on the Internet is unaware of the SCTP tunnel. This has the advantage that the end-to-end IP connections can function without deploying any additional

mobility solutions. It is the responsibility of the SCTP tunnels to make sure data is exchanged between train and wayside. Every time a link comes up, the connection setup procedure is repeated.

Another advantage of SCTP is multihoming: a connection can survive link failures as long as at least one of the links is available. SCTP maintains a main path and several backup paths, with heartbeat messages going over each of them. If the main path fails, SCTP falls back on one of the backup paths. In the TRACK architecture, link failure is also signaled to the Scheduler to ask whether the QoS constraints can be met. If due to link failure the available bandwidth gets too small, the Marker and the Shaper will act accordingly.

4 Results

4.1 Test Setup

As mentioned in section 2.2, a prototype framework has been implemented using the Click[8] framework. All building blocks shown in figure 1 were implemented using Click, with the exception of the "Traffic Optimizer" 1 and 2. In the prototype, these elements were left empty. As these elements provide well-known and mature functionality, such as proxies or header compression, they were not taken into account when testing the performance of the TRACK framework. If required, an existing implementation that provides these functions can be easily plugged in to the overall TRACK framework.

The main test focus was on the performance of Accelerator, Shaper, Scheduler and Tunnel modules. Most importantly, the cumulative effect of all these elements was tested to prove that they do not impact each other negatively when all of them are enabled. In order to test this prototype, an emulated setup was created using IBBT's Virtual Wall infrastructure. This infrastructure allowed us to emulate the network between a (virtual) moving train and a (virtual) wayside control center. All hosts ran Debian Linux, with SCTP support enabled. The TCP stack used is the standard Debian TCP stack, cubic TCP.

The test setup used is shown in figure 2; it consists of:

- Two *Train Hosts*, which are used to simulate onboard equipment;
- a *MCE* that has two Internet uplinks;
- *Impair*, the machine that is responsible for the network emulation. It adds and can constantly vary delay, jitter and, packet loss and varies the bandwidth between MCE and WCE;
- a *WCE* that has only one Internet uplink;
- a *Wayside Host*, simulating equipment in the control center.

To test the performance of all components, various test scenarios have been defined. The most important ones are:

- Prioritizing some flows over others;
- Switching traffic flows from one link to another:

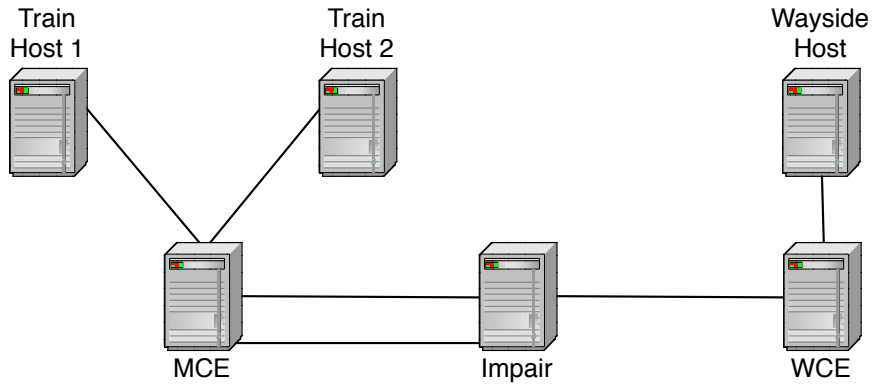


Fig. 2. Virtual Wall Setup

  - Switching between links with equal delay characteristics;
  - Switching from a low delay (10 ms) to a high delay (200 ms) link;
  - Switching from a high delay (200 ms) to a low delay (10 ms) link;
- Bandwidth and delay variation on a link;
- TCP accelerator effect on multiple TCP flows;
- SCTP tunnel performance.

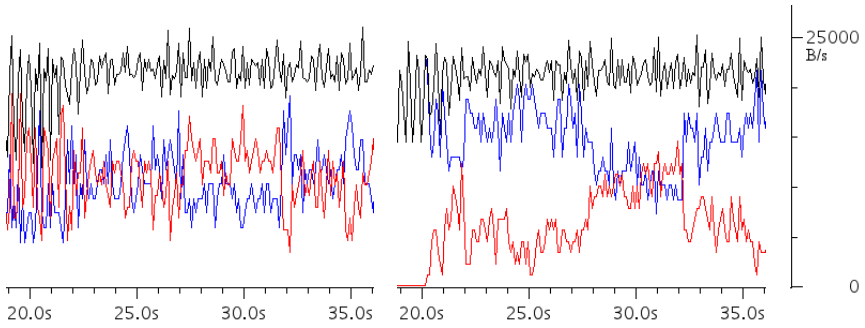
4.2 Test results

Flow Priorities As an example of prioritizing some flows over others, figure 3 shows two (excerpts of) graphs, each containing two traffic flows. The blue line corresponds to a simulated CCTV video feed, which has high priority. The red line is a background bulk file transfer with low priority. The black line shows the total bandwidth used.

It should be noted that, when running this test, the link emulation was set in such a way that the link could not support both flows simultaneously at all times. In figure 3(a), no priority scheme was enabled. In figure 3(b), the CCTV video was prioritized over the file transfer. As can be seen in the results, the total amount of bandwidth consumed is roughly the same in both cases. However, in the second case, the CCTV feed consumes considerably more bandwidth, while the file transfer is using less. The variations in bandwidth used by the CCTV, as seen in figure 3(b), are caused by the Variable Bit Rate (VBR) codec used.

The playback of both video feeds, which were recorded at the receiver, also show a very noticeable difference in both cases. While the first case renders the CCTV useless due to too much packet loss, the priority scheme ensures the CCTV video feed has sufficient quality to watch.

TCP Accelerator & SCTP Tunnel In figure 4, two graphs are shown that demonstrate the effect of both TCP accelerator (PEP) and SCTP tunnel. The



(a) Flows have the same priority (b) Flows have different priority

Fig. 3. Prioritization Test

graph shows the throughput, in bytes per second, of three concurrent secure copy sessions (SSH over TCP) from a host on the wayside to a host on the train. The three sessions copy a 30 MB file over a 10 Mbps link that has a RTT (round trip time) of 400 ms. Each session was started 10 seconds after the previous one. Figure 4(a) shows the baseline scenario: both PEP and the SCTP tunnel have been disabled. Figure 4(b) shows the same test, but this time PEP and SCTP tunnel are both enabled. In both figures the black line represents the total amount of data on the link, while the red, blue and purple lines plot the individual SCP sessions. The data to generate these plots was gathered on the receiving host.

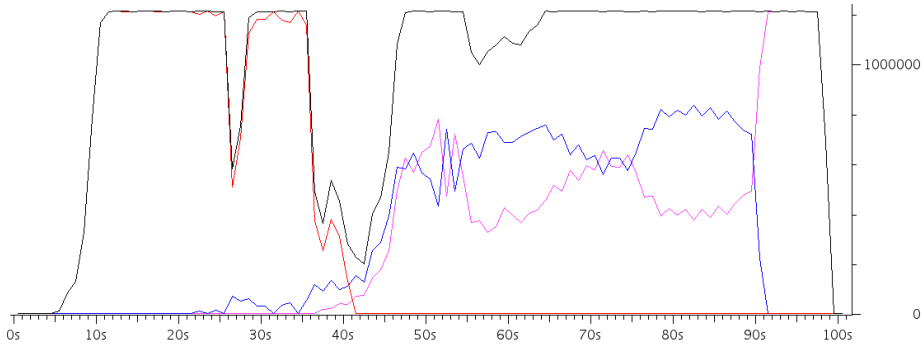
As can be seen in figure 4(a), in the baseline scenario, the second (blue) and third (purple) SCP sessions start very slowly. They only reach a reasonable throughput after the first SCP session stops. After that, it still takes some seconds before the second and third SCP sessions are able to fill the link, due to the TCP windowing mechanism operation.

When we enable both PEP and the SCTP tunnel, the result is shown in figure 4(b). Several effects can be observed:

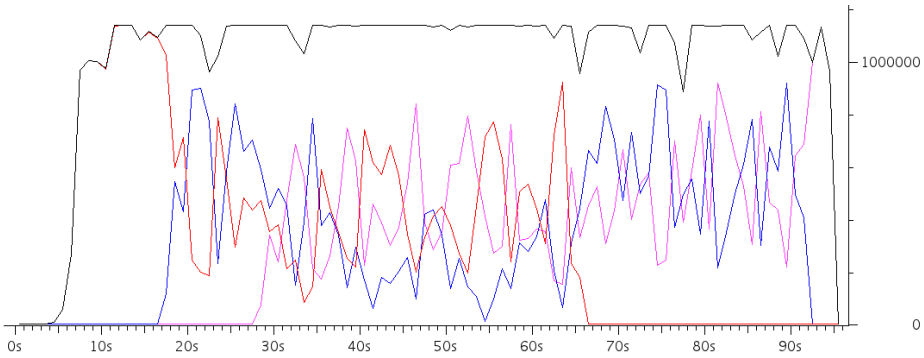
1. The second and third session sessions start much more quickly;
2. Bandwidth utilization remains close to maximum, without large dips as in 4(a);
3. Each SCP sessions receives, on average, the same amount of bandwidth at all times. There is no longer one session that receives considerably more bandwidth than the others.
4. The total amount of time needed to perform the three SCP file transfers, is 5% shorter.

5 Conclusion and Future Work

In this paper, we presented an overview of the TRACK network architecture. This architecture has been designed to provide a reliable train-to-wayside com-



(a) TCP Accelerator and SCTP tunnel disabled



(b) TCP Accelerator and SCTP tunnel enabled

Fig. 4. TCP Acceleration and SCTP Tunneling Test

munication architecture that can use multiple uplink technologies simultaneously and supports mobility without requiring modifications to the end hosts. To test the architecture, a prototype has been built using the Click Modular Router platform. This prototype has enabled us to run a multitude of simulations to test the functionality and performance of each individual component, as well as the architecture as a whole. The results of two of these experiments have been included in this paper as examples.

A follow-up project for TRACK called RAILS (Railway Applications Integration and Long-term networkS) has started on January 1st, 2012. This follow-up will build upon the TRACK network architecture, extending reliability and load balancing features, as well as integrating 3G/4G femtocells in the train coaches in order to increase in-train cellular network coverage. This introduces some challenges that need to be examined: in normal circumstances, femtocells are assumed to be placed stationary and have a fixed, high-quality Internet uplink. Onboard a train coach, however, neither of these assumptions still holds. While the train is moving, the radio environment around the femtocells continuously and rapidly changes. Moreover, the Internet uplink for the femtocells is also

continuously changing in bandwidth, delay and jitter. These effects will most probably impact the performance of the femtocells. During the project, not only will the TRACK network architecture be extended, but full cumulative RF (Radio Frequency) exposure inside a train coach will be examined, most notably to compare the situation with/without cellular repeaters and with/without onboard femtocells. A final important network-related subject that will be treated in RAILS is the development of specialized handover mechanisms that take the specific characteristics of a train environment into account (e.g. the fact that a train follows a fixed route). The mechanisms developed could try to minimize the number of handovers, more intelligently choose a handover moment and introduce a self-learning process that dynamically builds up a database of preferred cells along a train route.

Acknowledgements This research is partly funded by the Flemish Interdisciplinary Institute for BroadBand Technology (IBBT) through the TRACK project. The authors would like to thank all academic and industrial partners of this project.

References

1. Babiarz, J., Chan, K., Networks, N., et al.: Configuration guidelines for diffserv service classes. RFC 4594, Internet Engineering Task Force (August 2006), http://www.ietf.org/rfc/rfc4594.txt
2. Blake, S., Black, D., Carlson, M., et al.: An architecture for differentiated services. RFC 2475, Internet Engineering Task Force (December 1998), http://www.ietf.org/rfc/rfc2475.txt
3. Bormann, C., Burmeister, C., Degermark, M., et al.: RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP, and uncompressed. RFC 3095, Internet Engineering Task Force (Jul 2001), http://tools.ietf.org/html/rfc3095
4. Casner, S., Jacobson, V.: Compressing IP/UDP/RTP headers for low-speed serial links. RFC 2508, Internet Engineering Task Force (Jan 1999), http://tools.ietf.org/html/rfc2508
5. Devarapalli, V., Wakikawa, R., Petrescu, A., Thubert, P.: Network mobility (nemo) basic support protocol. RFC 3963, Internet Engineering Task Force (Jan 2005), http://tools.ietf.org/html/rfc3963
6. Fokum, D., Frost, V.: A survey on methods for broadband internet access on trains. Communications Surveys & Tutorials, IEEE 2, 171–185 (2010)
7. Jacobson, V.: Compressing TCP/IP headers for low-speed serial links. RFC 1144, Internet Engineering Task Force (Feb 1990), http://tools.ietf.org/html/rfc1144
8. Kohler, E.: The Click Modular Router. Ph.D. thesis, Massachusetts Institute of Technology (Nov 2000), http://read.cs.ucla.edu/click/click
9. Liang, X., Ong, F., Chan, P., Sheriff, R., Conforto, P.: Mobile internet access for high-speed trains via heterogeneous networks. In: Proceedings on IEEE PIMRC 14th International Symposium. vol. 1, pp. 177–181 (2003)
10. Moskowitz, R., Ricander, P.: Host identity protocol (HIP) architecture. RFC 4423, Internet Engineering Task Force (May 2006), http://tools.ietf.org/html/rfc4423

11. NASA: Space communications protocol standards, http://www.scps.org/
12. Nordmark, E., Bagnulo, M.: Shim6: Level 3 multihoming shim protocol for IPv6. RFC 5533, Internet Engineering Task Force (Jun 2009), http://tools.ietf.org/html/rfc5533
13. Pareit, D., Van de Velde, E., Naudts, D., Bergs, J., et al.: A novel network architecture for train-to-wayside communication with quality of service over heterogeneous wireless networks. EURASIP Journal on Wireless and Networking (2012)
14. Perkins, C., Johnson, D., Arkko, J.: Mobility support in IPv6. RFC 6275, Internet Engineering Task Force (Jul 2011), http://tools.ietf.org/html/rfc6275
15. R. Stewart, M. Ramalho, C.S.I., et al.: Stream control transmission protocol (sctp) partial reliability extension. RFC 3758, Internet Engineering Task Force (May 2004), http://www.ietf.org/rfc/rfc3758.txt
16. Rovcanin, M., Naudts, D., Pareit, D., Van de Velde, E., Bergs, J., Moerman, I., Blondia, C.: Data traffic differentiation and qos on the train, in fast parameter varying, heterogeneous wireless networks. In: Proceedings on 18th IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT). pp. 1–6. 22–23 (2011)
17. Scalise, S.: Summary of conclusions and outlook esa artes 1 project internet to trains initiative, presentation. Tech. rep., ESA-ESTEC, Noordwijk NL (2008)
18. Squid: Optimising web delivery, http://www.squid-cache.org/
19. Stewart, R., Xie, Q., Motorola, et al.: Stream control transmission protocol. RFC 2960, Internet Engineering Task Force (October 2000), http://www.ietf.org/rfc/rfc2960.txt
20. Zimmermann, H.: Osi reference model - the iso model of architecture for open systems interconnection. IEEE Transactions on Communications 28(4), 425–432 (April 1980)
21. Ziproxy: the HTTP traffic compressor, http://ziproxy.sourceforge.net/