Technical Specification Group Services and System Aspects **TSGS#27 (05)0088**

Meeting #27, Tokyo, Japan, 14-17 March 2005

3GPP SA4#34 meeting *Tdoc S4-050250*
21-25 February 2005, Lisbon, Portugal

Source:	**TSG SA WG4**
Title:	**Report of FEC selection for MBMS**
Document for:	**Information**
Agenda Item:	**7.4.3**

1. Introduction

This document summarizes the activities carried out in SA4 for developing FEC at application layer.

2. Motivation for FEC and historical background

About one year ago SA4 was made aware of the radio link quality offered by GERAN and UTRAN, and decided that error rates perceived at application layer were higher than those achievable for other services (such as PSS).

One possibility was to accept the radio link quality and not try to improve the performance. However, despite this was also initially considered, SA4 decided to include Forward Error Correction (FEC) as a mechanism for improving the error resilience of a data stream.

FEC algorithms allow reducing the application residual error rate to a target level that is acceptable for a given application. This is achieved by adding a certain amount of redundancy (or FEC overhead) to the transmitted data stream. The amount of redundancy depends on the radio layer error rate, and on the target level of residual error rate for a given application. For example, for higher radio error rates, more FEC redundancy is needed than for lower radio error rates; and for applications like streaming, less FEC overhead is required than for application like file download (the latter requiring generally nearly error-free delivery).

3. Process in SA4

SA4 has set-up a process for developing FEC algorithms. This can be summarized in the following steps:

1. Requesting the necessary information from the GERAN/RAN WGs about characteristics for the radio link performance
2. Defining a set of simulation guidelines for SA4 internal use, with the objective of evaluating the performance of different FEC algorithms.
3. Select one (or more) FEC algorithms based on the performance and other characteristics.

3.1 Request of information

The request of information to the GERAN/RAN WGs took a non-negligible amount of time, as the radio layer was still in the specification phase, and SA4 did not receive all the information requested (today it is known that RAN4 will complete the performance characterization of MBMS by September ’05). However, a certain amount of information was received and SA4 could start working on FEC development.

3.1 Simulation guidelines

SA4 has defined a set of simulation guidelines in document S4-040348. These guidelines contain a set of simulation parameters and assumptions useful to conduct experiments for FEC. Here is a brief summary of the guidelines (extract from the minimum set of parameters):

- File sizes (for download): 100KB, 500 KB, 3 MB;
- UTRAN bearer bit rate: 16-128 kbps
- GERAN bit rate: 8-29.58 kbps
- Random transport block losses of 0.5%, 1%, 5%, 10% for UTRAN
- Random RLC data block losses of 0.05%, 0.1%, 0.5%, 1% for GERAN
- Cell change losses of 1s, 2s, 3s.

In addition, system level simulations for a large amount of users (e.g., 1000) were done by using the following classes. Each class is representative of one or more user experiencing a certain channel condition.

Class	PDU BLER [%]	Handover per minute	Applicability
1	0.1	0	UTRAN/GERAN
2	1	0	UTRAN/GERAN
3	10	0	UTRAN
4	0.5	1	UTRAN/GERAN
5	5	1	UTRAN
6	1	3	UTRAN/GERAN

The total network user population could be characterized by a certain fraction of users belonging to each class. We used, for example, weights for each class of {20%, 50%, 4%, 20%, 1%, 5%} or {10%, 50%, 10%, 20%, 5%, 5%} (each % value represents the amount of users in each class out of a total of 100% population.

The simulation guidelines were followed as close as possible during the simulation period. However, in some cases, companies were producing results not always exactly according to these guidelines. It has to be pointed out that the simulation guidelines were based on assumptions and information received in the beginning of 2004, and frozen in May 2004. No review or update of these guidelines was performed.

3.2 FEC selection

Initially there were 6 candidates. As of today, the current candidates are:

- Reed-Solomon based FEC
- Raptor FEC

After a set of simulations conducted by the proponents, here is presented a summary of the characteristics of each of the candidates based on the following use cases and error rates:

Use Cases:

- Low bit rate streaming (32, 64 kbits bearer)
- High bit rate streaming (128, 256, 384 kbits bearer)
- Low file size download (50 – 512 kbytes)
- High file size download (1Mbytes – 3 Mbytes)

Error Rates (RLC-PDU, statistically independent)

- Low error rate (1%)
- Medium error rate (5%)
- High error rate (10 %)

Reed-Solomon codes

Reed-Solomon codes are systematic Codes. These codes are a well-known technology. The main aspects of Reed-Solomon codes are as follows:

Resource usages

NOTE: these figures below must be considered provisional due to recently identified discrepancies discovered between different company simulations. This issue must be resolved before the next SA plenary and the figure below will be updated or confirmed.

This is measured in terms of extra network bandwidth required.

	Low Bit rate streaming (64 kbps)	High Bit Rate streaming (384 kbps)	Low file size download (512 KB)	High file size download (3MB)
Low error rate	7%	6%	12%	5%
High error rate	35%	34%	31%	28%

Figure 1 - FEC overhead for error-free reception

Computational Complexity

This is measured in terms of %CPU load for streaming and decoding time for file download. The CPU load for streaming is spread over the protection period.

	Low Bit rate streaming (64 kbps)	High Bit Rate streaming (384 kbps)	Low file size download (512 KB)	High file size download (3MB)
ARM 11	<= 1.2%	N/A	0.13s	1.6s
Future platform	N/A	<=1.42%	0.035s	0.4s

Figure 2 - % CPU load for streaming, and decoding time for file downloading (high error rates)

Memory Consumption for download

- For low file sizes (512 KB) Reed-Solomon requires 700KB of fast memory.
- For high file sizes (3 MB) Reed-Solomon requires 4MB of fast memory.
- Memory management algorithms can be implemented to reduce the usage of fast memory (with increase in decoding time). These algorithms are not subject of standardization

Flexibility

- Reed-Solomon is applicable for wide ranges of bearer bit rates (from small bearer bit rates up to 384 kbps bearers) with competitive performance (*)
- Reed-Solomon offers the possibility of having flexible protection periods (e.g., from 5s to 20s) to fight against cell change losses, with competitive performance (*)
- Reed-Solomon offers the possibility of handling variable source packet size (with low additional overhead but with computational complexity and performance implications).

Latency

- The tune-in delay is a function of the protection period and the decoding delay (typically the tune-in delay is *protection_period*+ε with an upper bound of *2*protection_period*).
- For live content, the end to end delay is equal to *protection period+decoding delay*, except when data is protected with interleaved blocks, where the end-to-end delay is at least twice the protection period.

(*) Digital Fountain does not agree that competitive performance of these Reed Solomon codes has been demonstrated in these cases.

Raptor codes

Raptor Forward Error corrections codes are systematic erasure codes. The same code is proposed for both download and streaming cases and the specific code proposed has been available and stable since May 2004. Full specification text is available for approval at SA.

The key aspects of Raptor codes are summarised as follows:

Flexibility:

Raptor codes can efficiently support all present identified MBMS requirements in terms of bearer rates, file sizes, loss rates, packet sizes, packet size variability and protection period for streaming without negative tradeoffs in terms of other parameters or computational complexity. This is true for all loss rates. Code performance in all cases is close to the ideal code.

Raptor codes therefore allow for efficient use of optimisations such as high-bit-rate cases such including delivery of multiple bundled media streams and very large file downloads.

Resource usages:

NOTE: these figures below must be considered provisional due to recently identified discrepancies discovered between different company simulations. This issue must be resolved before the next SA plenary and the figure below will be updated or confirmed.

Low file sizes (50-512kbytes): Raptor codes require within 0.8% of the transmission resources of an ideal code at low loss rates (1% BLER). At high loss rates (10% BLER) the figure is < 0.2%.

NOTE: these figures are relative to the theoretical minimum overhead required to overcome the stated loss conditions (i.e. the overhead of an ideal code). The total overhead required is the sum of theoretical minimum overhead and the figures above.

High file sizes (1MB-3MB): Raptor codes require within 0.1% of the transmission resources of an ideal code at both high and low loss rates.

Streaming: For high quality streaming (FEC block decoding failure rate between 10^{-2} and 10^{-4}) with short protection periods (5s or less), Raptor codes tolerate 93-95% of the BLER tolerated by an ideal code for low bit-rates (32kbit/s, 64kbit/s) and 97-98% for high bit-rates (128kbit/s, 256kbit/s). Longer protection periods (10-30s) can also be efficiently supported if required, with tolerable BLER even closer to the ideal code.

Computational complexity:

Raptor codes have very low CPU load, allowing large blocks of data to be decoded over a short time period whilst maintaining a low CPU load.

At high loss Raptor is less complex than Reed-Solomon codes whereas at low loss the codes are comparable.

Memory consumption for file download:

The working memory requirement for Raptor codes is 512k.

Latency:

Raptor codes maintain the order of sending of source packets for streaming so that no additional encoding latency is introduced and so that no additional delay is introduced at decoders when a stream is joined within a protection period (under no losses or low losses).

The decoding delay for Raptor codes can efficiently be set as a small fraction (<10%) of the protection period with low CPU load during decoding even in worst case loss conditions. Hence Raptor is able to efficiently support any desired overall tune-in delay (=protection period +decoding delay).

0. Conclusion

In SA4 there was no final agreement on the FEC algorithm selected for MBMS Rel. 6. There is also no agreement on whether to adopt one or two FEC codes and whether to have it mandatory or not.