Project Number:	**Contract Number: INFSO-ICT-224287**
Project acronym:	**VITAL++**
Project Title:	**Embedding P2P Technology in Next Generation Networks: A New Communication Paradigm & Experimentation Infrastructure**
Title of Report	**Reference Services Deployment and Validation**

Instrument:	STREP
Theme:	ICT-2-1.6
Report Due:	M27
Report Delivered:	
Lead Contractor for this deliverable:	TID
Contributors to this deliverable:	Konstantinos Papanikitas (UoP), Jens Fiedler (Fokus), Konstantinos Koutsopoulos (BCT), Gregorio Martín, David Flórez (TID), Shane Dempsey (WIT), Evangelos Markakis, Evangelos Pallis (CTRC), Nico de Abreu Pereira (RBB), Wolfgang Brandstätter (TA), George Kapelios (VG)
Estimated person Months:	31 MM
Start date of project:	1st June 2008
Project duration	30 months
Revision:	Draft
Dissemination Level:	PU - Public
Internal Review:	Wolfgang Brandstätter (TA)

This page intentionally blank

Table of Contents

Table of Contents 3
List of Figures 5
List of Tables 6
1 Executive summary 7
2 VITAL ++ Reference Services 8
2.1 SoftMix 8
2.1.1 File Upload 9
2.1.2 File Retrieval 10
2.1.3 PlayList Broadcast 11
2.2 Video Broadcasting 11
2.2.1 TV Broadcast 12
2.2.2 Video Webcam 14
2.3 Mobile Service– Media Relay 15
2.4 Transcoding Enabler Service 16
3 Testing of Vital++ Elements 18
3.1.1 Peer to peer authentication P2PA-SA 18
3.1.1.1 Performance Test 1 18
3.1.1.2 Performance Test 2 18
3.1.1.3 Functional Test 1 18
3.1.2 Content Protection CP-SA 18
3.1.2.1 Licensing Performance (without accounting) 18
3.1.2.2 Aggregate Licensing & Accounting Performance 19
3.1.3 Content indexing CI-SA 20
3.1.3.1 CI Performance Measurements 20
3.1.3.2 CI Message Length Limitations 21
3.1.4 Overlay Management OM-SA 21
3.1.4.1 OM-SA performance under churn 22
3.1.5 Transcoder Service 22
3.1.5.1 Transcoding Performance vs. file size 22
3.1.5.2 Transcoding Performance vs. query's load 22
3.1.6 DVB-T environment 23
3.1.6.1 DVB_T-TC1 (System Performance 1) 23
3.1.6.2 DVB_T-TC2 (system performance 2) 23
4 Real User Experiments 25
4.1.1 Aim and Scope 25
4.1.2 Tests 26
4.1.3 Tests 27
5 Vital++ and PII integration 28
5.1.1 Aims and Scope 28
5.1.2 Technical Details 28
5.1.3 Validation Tests 30

6 Summary 32

List of Figures

Figure 1: SoftMix Vital++ testing ground 9
Figure 2: Sequence of events in File Upload 10
Figure 3: Sequence of Events in File Retrieval 11
Figure 4: The TV Broadcast demo 13
Figure 5: Satellite TV feed to UoP testbed 13
Figure 6: Video Webcam demo 15
Figure 7: Mobile Service - Media Relay 16
Figure 8: Transcoder Service 17

List of Tables

This page intentionally blank

1 Executive summary

D5.1 is devoted to the interconnection and configuration of the various testbeds made available by consortium's partners, in order to build a single shared testground for Vital++ purposes. The aim of this deliverable is twofold, on the one hand, to describe which services have been deployed on the Vital++ infrastructure and how they are used to demonstrate the validity of the Vital++ concept, and on the other hand, to specify the tests required to achieve this aim.

This deliverable consists of the following sections

- Section 2 describes the services deployed on the infrastructure described in D5.1
- Section 3 enumerates the tests designed for the validation/assessment of each SA/architectural element implemented for Vital++
- Section 4 defines the experiments to be carried out by involving real users in order to assess the scalability/stability/performance of Vital++
- Section 5 aims at describing how Vital++ and PII are going to be integrated, to what extent this target will be pursued and the definition of the tests which will check that this integration has been achieved

2 VITAL ++ Reference Services

In D4.2, the Vital++ a number of use cases were been defined as a template for Vital++ services in D5.2. The aim of this section is to provide a short description of these final services and how they are used to validate Vital++ functionalities

2.1 SoftMix

The Softmix Service is the main testing service for Vital++ and enables the user to interact with a community in the following ways:

- upload and share contents (only audio) with other Vital++ users,
- request the download and play of contents available in the system, and
  compile playlists of content items and share them with other Vital++ users

The rationale behind this sort of service is to assess how Vital++ would behave in a real Internet user's environment. In other words, Softmix will emulate a service resembling a social network where users join in order to share contents and experiencies, building on a preexisting database of audio contents. Thus, SoftMix enables the user to receive a personalised radio service which delivers exactly the kind of content that matches the available content and the user's individual preferences. The profiling and matching happens on the client side and does not involve the VITAL++ architecture.

The Softmix Service also involves Content Protection SA (cf. 3.1.2), the Content Indexing SA (cf. 3.1.3), Overlay Management SA (cf 3.1.4) and the Transcoding Service (cf. 3.1.5) .These elements are deployed in different location across Vital++ testing ground, as it is shown in the following figure:

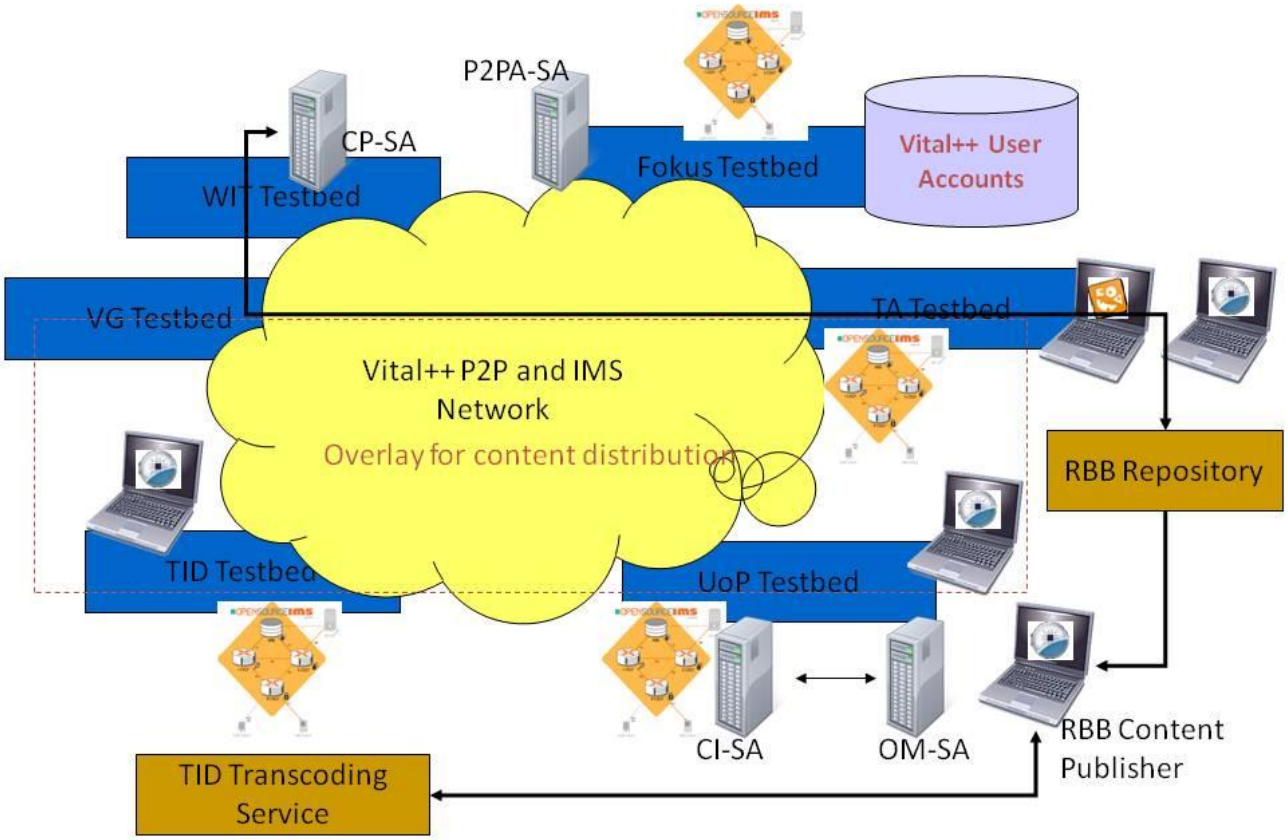

Figure 1: SoftMix Vital++ testing ground

The following subsections 2.1.1-2.1.3 briefly describe the features of Softmix For a more detailed description relevant features refer to D4.2.

2.1.1 File Upload

File upload stands for the means put at the user's disposal in order to upload contents and share them with other users, by storing it in Vital++ systems and make the different SA aware of its existence. In the following figure, it is illustrated the sequence of events trigerred while uploading and the order in which the various Vital++ subarchitectures are involved.

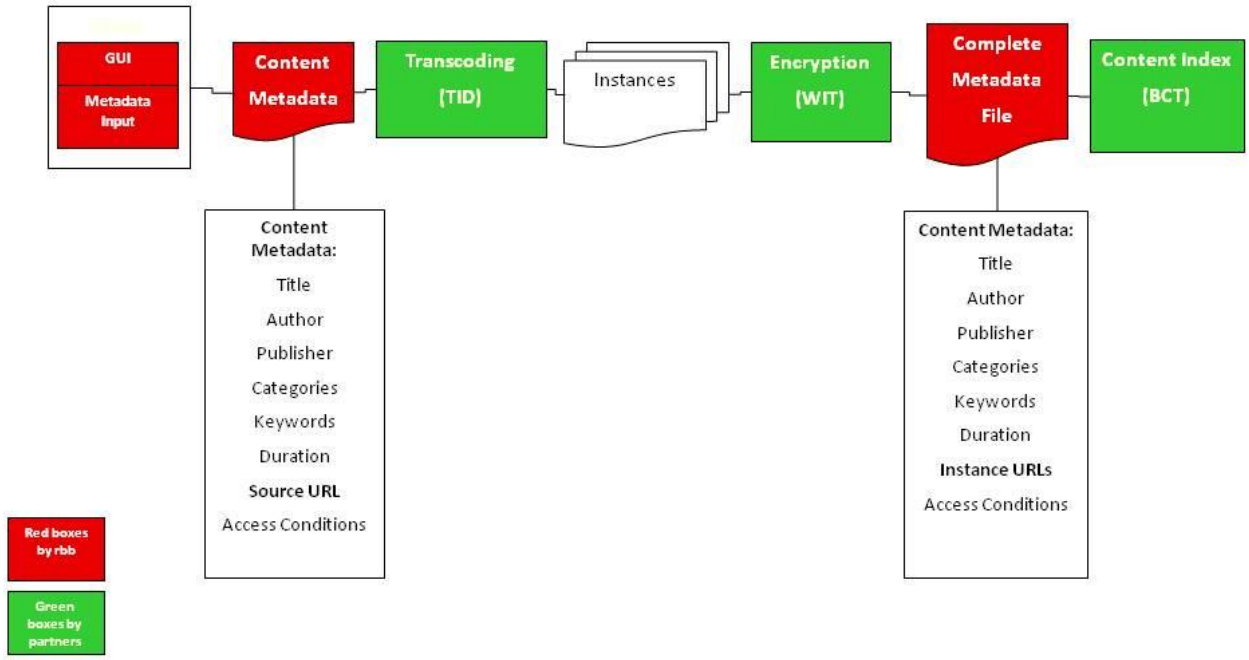

Figure 2: Sequence of events in File Upload

The retrieval of the contents is the subject of the next section.

2.1.2 File Retrieval

File retrieval stands for the means at user's disposal to retrieve content available in the system. It comprises to main tasks:

- Search of contents matching user's preferences
- Trigger and control of its download

The following figure depicts the sequence of events triggered while retrieving contents, and the order in which the various Vital++ subarchitectures are involved.

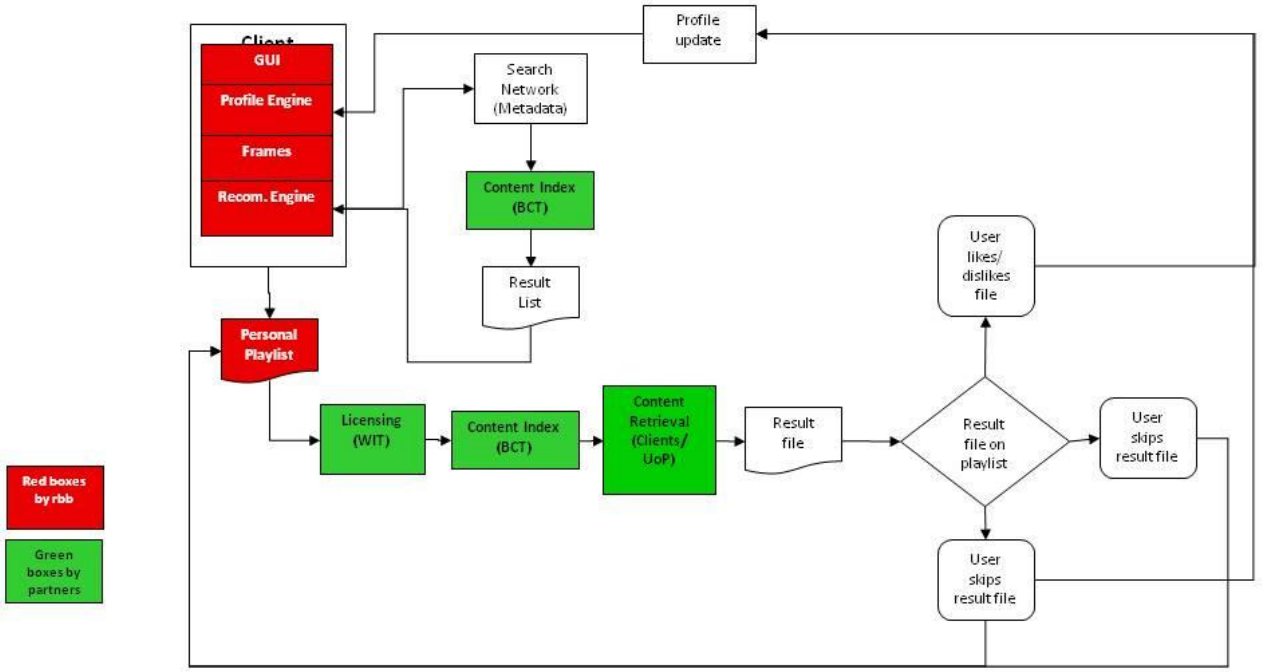

Figure 3: Sequence of Events in File Retrieval

2.1.3 PlayList Broadcast

Play list Broadcast has a threefold meaning.

- First, the means a user has to compile playlists and share them with other users.
- Second, how a user search and locate playlists
- Third, how the download is triggered and controlled

It will not include the actual download, which will be carried out by the File Retrieval procedures explained above.

2.2 Video Broadcasting

Since Softmix is mainly focused on broadcasting audio content, two services specifically devoted to video matters have been developed and implemented.

The main difference between them is that in one the content is provided by a Vital++ user, by sharing the feed of his/her webcam, while for the other a commercial TV broadcast is put at any user's disposal.

2.2.1 TV Broadcast

Description

This scenario realises the IPTV injection scenario, where live streaming TV content is fed from an active-user (potential Broadcaster) located within the regenerative DVB-T platform at CTRC premises onto the UoP test-bed, via a VPN connection. The received stream is processed by the P2P engine and distributed over the entire VITAL++ infrastructure.

Rationale

The Scenario of the Live Streaming Service provides the following add-on's to the VITAL++ project. The VITAL++ P2P technology is integrated in a real transmission/reception DVB-T infrastructure located in Heraklion City. We provide access to real users located inside the broadcasting Area. The Users located inside the Broadcast area are potentially users of the whole VITAL++ test-bed. Finally this experiment is a more resource demanding experiment stretching out VITAL++ architecture implementation.

Vital++ Implementation

The TV broadcast scenario of VITAL++ is implemented as follows: The active user of a DVB-T platform receives a DVB-S satellite TV programme, manipulates it (transcode / transrate) and streams it to the DVB-T platform (unicast) via the CMN and over the uplink. At the regenerative DVB-T platform, this stream is a) transmitted to the entire broadcasting footprint, and b) forwarded to the UoP test-bed via the VPN connection. The stream is then forwarded to a stream relay agent to the UoP content provider. In UoP the content provider will generate a public & private key pair. The public key will then be published to the VITAL++ certificate authority. Every piece of content is encrypted and a symmetric key is generated. This is encrypted using the private key of the content provider and distributed to the CP subsystem. The content is encrypted within the content provider's "super peer". In practice for a live content stream the SRTP symmetric encryption mechanism is used. The Content Provider associates accounting and licensing rules with that content identifier. The Content Provider notifies the CI of the new content. The content is now accessible via the Vital++ content overlay. In the following diagram the TV broadcast of a live streaming service is presented.

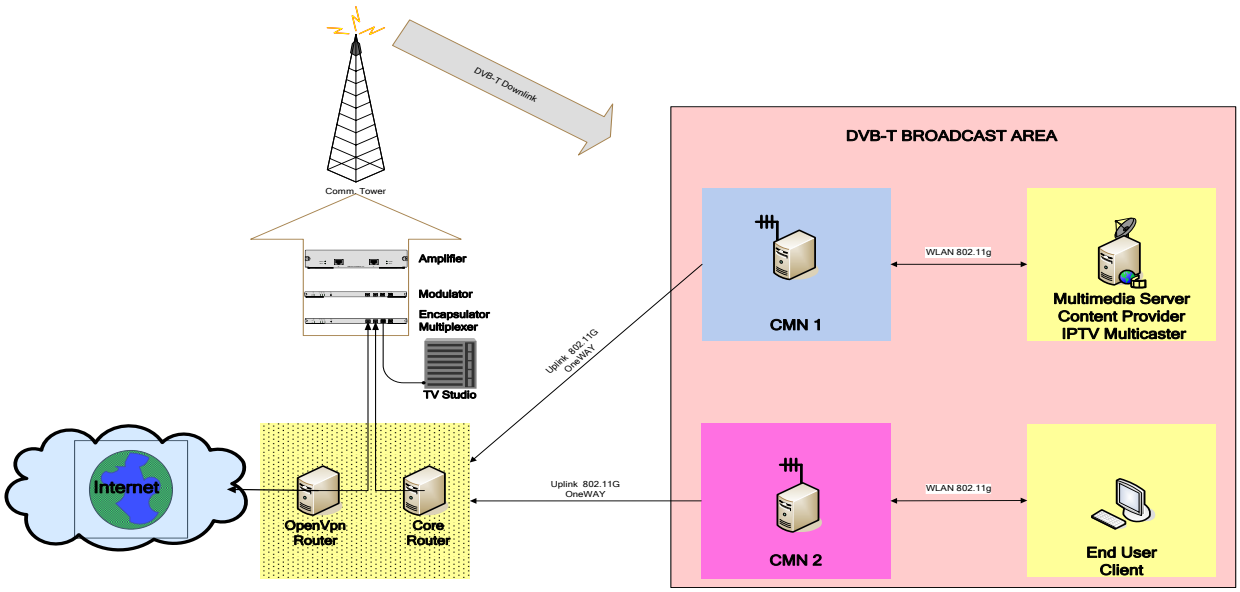

Figure 4: The TV Broadcast demo

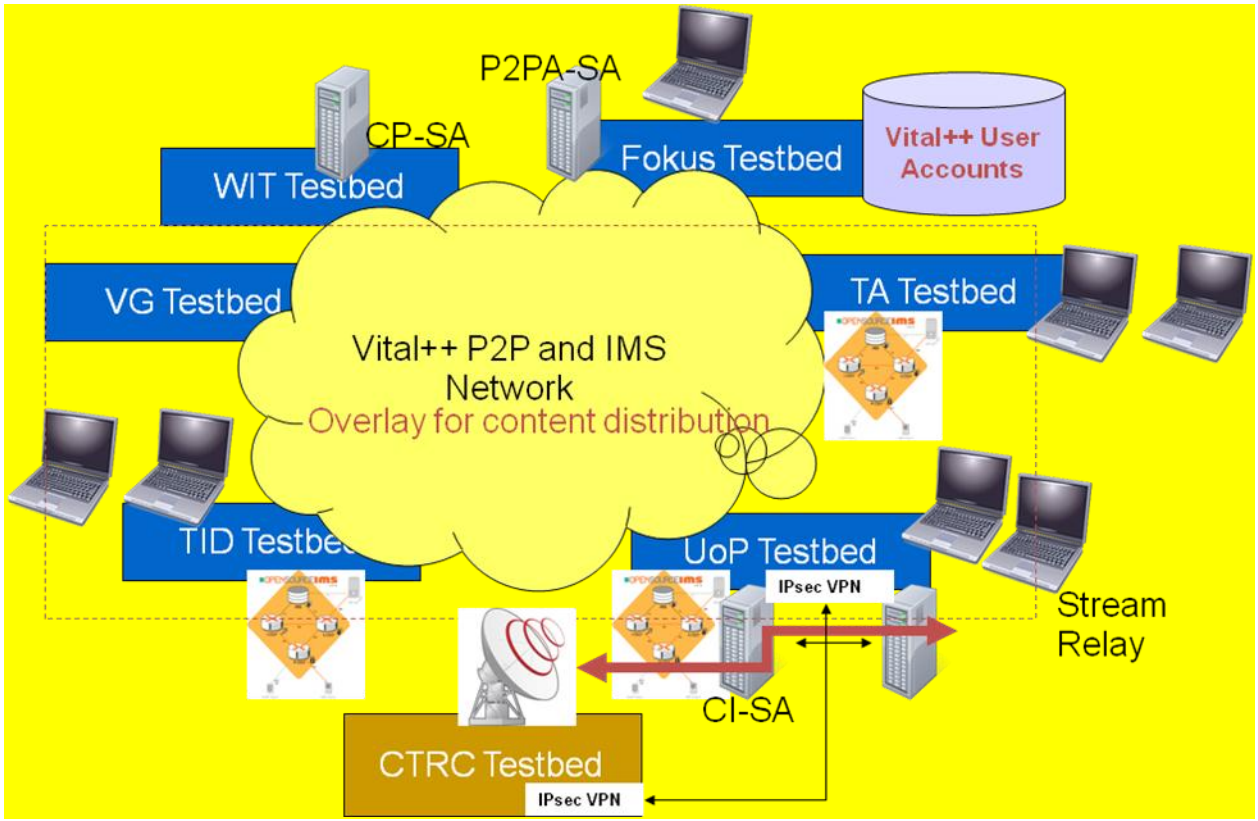

Figure 5: Satellite TV feed to UoP testbed

2.2.2 Video Webcam

Description

The vital++ user plays the role of a content publisher by providing a webcam stream to the Vital++ community.

Rationale

The concept of the webcam service s twofold. On the one hand, from the point of view of the technology and methodology implemented, this use case evaluates and validates the capability of the Vital++ platform and mainly of the P2P Engine to support provision of live content in a P2P manner without the need for any kind of caching or introduction of delays. On the other hand, from the user perspective, it introduces a new paradigm of user driven content beyond standard webcam techniques surpassing bandwidth limitations.

Vital++ Implementation

The Vital++ client is launched and a webcam stream is published to the Vital++ network. The content protection and publication mechanism is the same as in the TV broadcast demo and regards publication of the content key to the CP-SA, publication of the content to the CI-SA and initialisation of a new overlay in the OM-SA. Clients wishing to consume this stream send CI queries based on keywords and stream name to locate the available content. After acquiring the content license from the CP-SA, they get a list of peers from the OM-SA containing the overlay neighbourhood for the distribution of the content. According to the demo context, there were two choices for webcam broadcasting one for low and one for higher resolution. The underlying network is created on top of VPN connections established among the Vital++ partners' test sites. The webcam publication-consumption functionality is integrated in both BCT and Monster clients.

User Experience

The use case is not introducing any additional elements for the Vital++ client software. The requirement for the publisher is the usage of the word "webcam" in the title of the stream for low resolution or "webcam 320x240" for higher resolution. Publication may include any kind of keywords. On the other hand, the user wishing to locate such offerings has to define "webcam" in the stream title of the query.

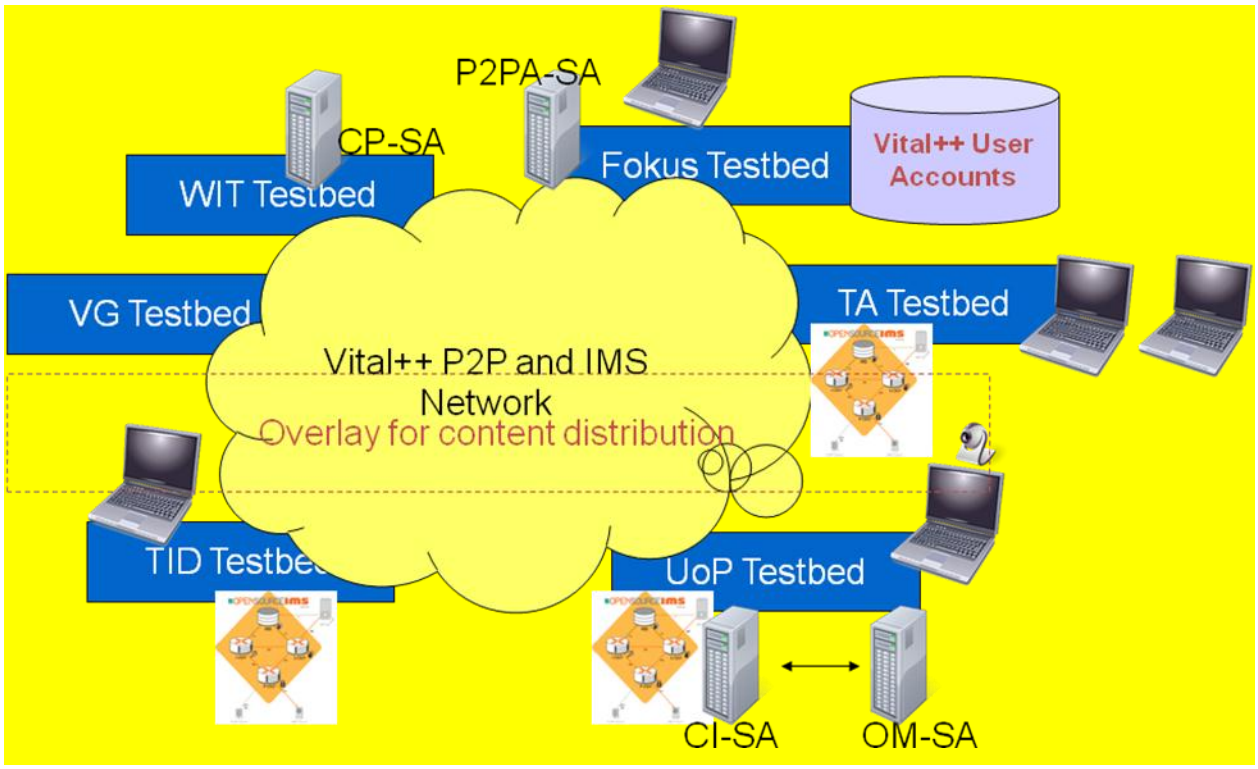

Figure 6: Video Webcam demo

2.3 *Mobile Service– Media Relay*

Description

The so-called mobile demo regards the deployment of software entities that are based on the BCT Client which have a hybrid role. On the one hand, these entities are configured to locate and consume media offerings from the Vital++ network. On the other hand, they make available the content collected via the overlays as regular RTP streams established via standard SIP calls.

Rationale

A successful deployment of the Vital++ platform beyond the project scope will potentially result in the availability of high volumes of media offerings. Assuming therefore a high degree of popularity, the media relay concept (or mobile) allows for further exploitation of the traded and/or free offerings in the mobile market. Additionally, the billing infrastructure in commercial SIP networks can be integrated with the Content Protection and Billing infrastructure of Vital++ and fee collection can be better performed.

Vital++ Implementation

The realisation of the mobile demo depends on entities that behave like normal Vital++ client instances for collecting media from the overlays towards the project's platform. While towards the regular SIP network they wait for SIP Invitations to establish RTP channels through which they forward the media to end users. As far as content location and retrieval is concerned there are not any differentiations from the standard client operation. Keywords and stream titles can be defined and sent to the CI-SA for retrieval of relevant item lists. From there items can be selected by acquisition of the relevant license from the CP-SA and overlays can be

joined via the OM-SA provided information. At the pure SIP tier of the relay entity a SIP account is configured and published via other means (e.g. web pages or SIP Instant Messaging provided information) so that end users can call for getting the stream via RTP. The SIP operator can define pricing schemes so that fees can be collected and part of these be paid back to the media publisher.

User Experience

End users invoke the services without any additional complexity beyond regular SIP agent management and usage. Obviously, for video items the SIP agent should be able to display the video over RTP media packets.

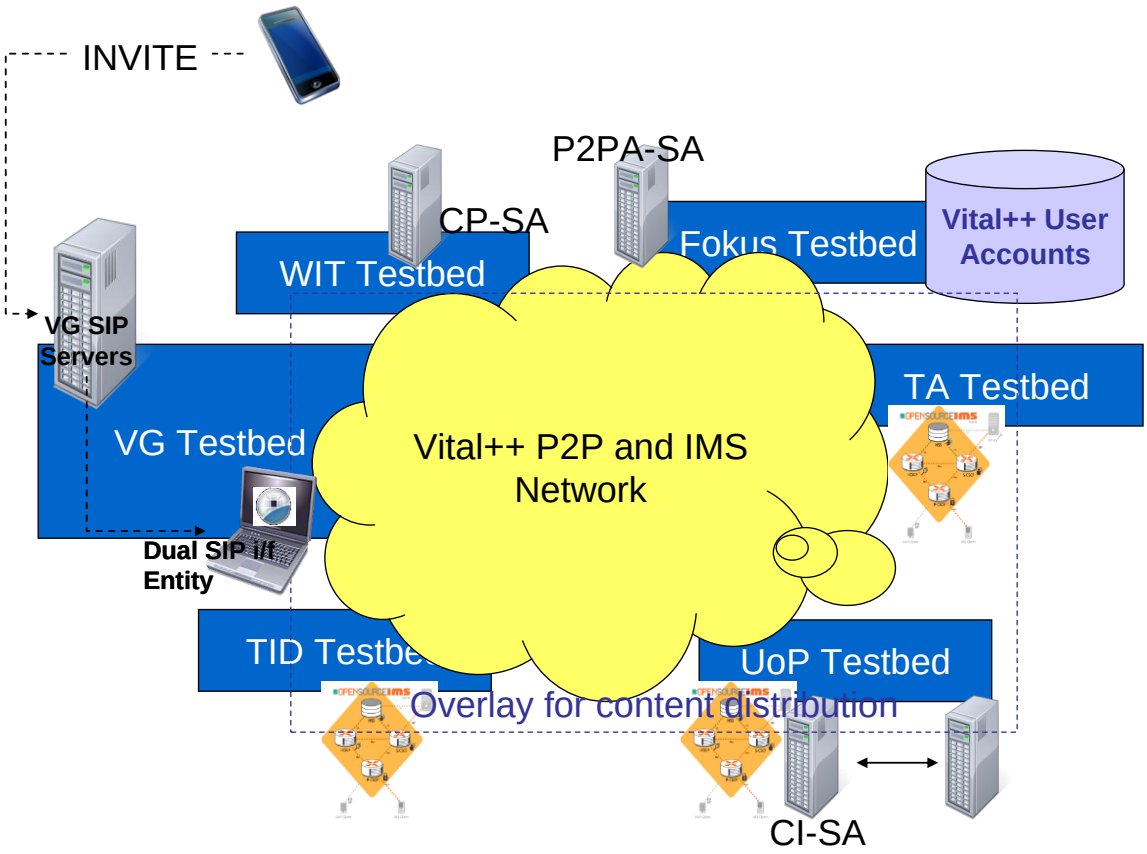

Figure 7: Mobile Service - Media Relay

2.4 *Transcoding Enabler Service*

Video download services like youtube, dailymotion… normally store the uploaded content of their users in different formats (resolution, frame rates). Thus, terminals endowed with fewer resources than those required by the original content can download and play the content without jeopardizing user's experience. However, no provision for transcoding of streaming contents has been carried out.

The purpose of the static transcoder module integrated in the vital++ architecture is to provide a system that allows multimedia content conversion, independently of content type like image, audio or video, but with different technical features.

Each time a vital++ content is required to be converted in format changes (container), audio or video bitrate modification, etc. the Transcoder Service is triggered through a web service using SOAP messages sent through HTTP.

Inside the transcoding engine the transcoding process takes place and the output is returned to the requester.

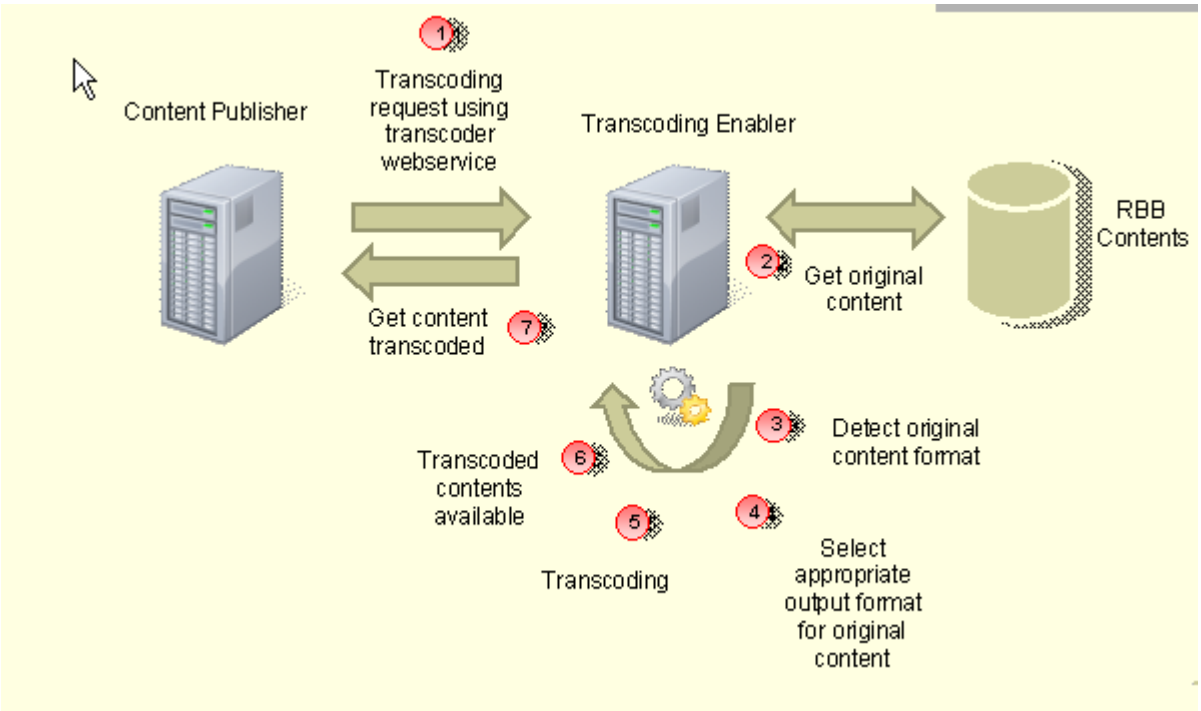

Figure 8: Transcoder Service

3 Testing of Vital++ Elements

The implementation of Vital++ has required the development of a number of SA in order to ensure that the Vital++ functionality is fully covered. Each SA requires a set of tests to be passed.

In this section, the definition of the tests is presented. The corresponding results will be discussed in D5.3. Expected results, if any, are considered as part of the definition and therefore belong to D5.2.

3.1.1 Peer to peer authentication P2PA-SA

This section contains the tests to be performed in order to check out P2PA proper working.

3.1.1.1 Performance Test 1

Procedure of Testing

Measure delay between exchanged messages and duration of authentication transactions between Client-Server (during initial credential exchange) and Client-Client (during credential exchange for P2P authentication).

3.1.1.2 Performance Test 2

Procedure of Testing

Measure server performance (delay) over increasing number of clients, realised by non-GUI clients.

3.1.1.3 Functional Test 1

Procedure of Testing

Deploy multiple non-GUI clients with pre-programmed scripts for P2P message exchange.

Then evaluate success/failure rate of authenticity checks over time.

3.1.2 Content Protection CP-SA

This section will contain the tests to be performed to check out CP proper's working.

3.1.2.1 Licensing Performance (without accounting)

A number O(10^3) of "Fair Use" licensing requests are made to the licensing service using the web interface. The total completion time is registered and an average is determined. The Apache JMeter tool is used to generate the traffic and log the results permitting an analysis of the produced data to determine mean and deviation of the request results, demonstrating the performance and scalability of the implementation.

Purpose of the Test

The purpose of this test is to determine the performance of licensing, representing an end-to-end exercise of the functionality of the Content Protection Subsystem, except the Accounting subsystem. A fair-use license is requested as this does not require the generation of user charging information and hence, does not exercise accounting functionality which is dealt with separately in a further test.

Our reliability metrics are:

- The unavailability rate corresponding to the number of requests which are not responded to;
- The error rate where the response is determined to be incorrect, bearing in mind that a valid response is easily checked using a text/string compare function.

Our performance metrics are:

- Mean and standard deviation of latency for the requests;
- Scalability by comparing performance with different numbers of clients making concurrent requests. 1, 5, 10, 50;
- Frequency analysis of latency/request and plotting of cumulative frequency distribution

Procedure of Testing

The tests will be carried out using the JMeter performance analysis tool for web services. The content protection service will be provisioned with a set of content to match the number of clients making requests. E.g. for 10 clients we will provision 10 content items which are available via a fair use license.

The resulting data will be analysed quantitatively and graphically using a spreadsheet with integrated statistics package such as Microsoft Excel.

The following data will be gathered for each test run

- A time stamp indicating when the test was run, in milliseconds;
- The Response code of the HTTP request;
- The amount of bytes in request and response;
- The latency of Request/Response in milliseconds.

Expected Results

We expect to see quite consistent latency performance within an acceptable range of values. As we use a load-balancing system per request we believe the results should be evenly distributed over the range of latency values. As we increase the number of clients making licensing requests the scalability should be quite linear, without large and unpredictable spikes but accepting the inevitably of occasional outliers.

3.1.2.2 Aggregate Licensing & Accounting Performance

A number O(10^3) of content licensing requests are made to the licensing service using the web interface. These requests are of a type and for content that requires a content charge to be calculated for the licensing transaction. Data is recorded for each request that can be used

to analyse the system's performance. The consistency of the licensing and charging data is checked.

The Apache JMeter tool is used to generate the traffic and log the results permitting an analysis of the produced data to determine mean and deviation of the request results, demonstrating the performance, scalability and consistency of the integration of accounting and licensing within the associated Subsystem Architecture implementations.

Purpose of the Test

The purpose of this test is to determine the performance of licensing, representing an end-to-end exercise of the functionality of the Content Protection Subsystem including the Accounting subsystem.

Our reliability metrics are:

- The unavailability rate (as before)
- The error rate (as before)

Our performance metrics are:

- Mean and standard deviation of latency for the requests;
- Scalability by comparing performance with different numbers of clients making concurrent requests. 1, 5, 10, 50;
- Frequency analysis of latency/request and plotting of cumulative frequency distribution

Procedure of Testing

The tests will be carried out using the JMeter performance analysis tool for web service. The content protection service will be provisioned with a set of content to match the number of clients making requests. E.g. for 10 clients we will provision 10 content items which are licensable and have an associated micro-charge.

The same performance data is collected as before. It is also analysed quantitatively and graphically using a spreadsheet with integrated statistics package such as Microsoft Excel.

Expected Results

As before, we expect to see quite consistent latency performance within an acceptable range of values. The same expectations regarding scalability and reliability apply as before, accepting that the Accounting/Rating engine is quite complex and is not running in an optimum "production" configuration for real-time micro-charging. The most important aspect of the results would be a very low unavailability rate (ideally zero) , a zero error rate and consistent performance.

3.1.3 Content indexing CI-SA

3.1.3.1 CI Performance Measurements

Purpose of the test

This test aims at evaluating the time required for the CI to perform certain actions that regard the possible queries from the users. User requests regard content publication, searching, and content selection. Each of these result in a certain response from the CI-SA the content of which is processed by the client entity either for presenting information to the user or for configuring client internal building blocks.

Procedure of testing

Since CI-SA is using an XML database to store and process published information, evaluation of the time required for certain transactions with the database will be evaluated by gradually increasing the volume of information stored in the database. Similarly, the time required for retrieving bigger result sets from the database due to availability of more items will be also measured.

All the transactions will be logged and the average values will be associated with the volume of the stored information.

Expected Results

The performance analysis and evaluation is expected to reveal any performance issues that may be linked with the volume of stored information and also with relation to the XML schema adopted for the database. In case significant degradation in the performance is detected, enhancements that may aid indexing will be evaluated and used. This test will also include the time required for the transactions between CI-SA and OM-SA.

3.1.3.2 CI Message Length Limitations

Purpose of the test

The availability of similar publications in the CI-SA may result in quite lengthy responses towards the users trying to locate media items. This may result in responses larger than 1.500 bytes in SIP Message Body that may be rejected by the involved IMS proxy functions.

Procedure of testing

Variations of a publication will be stored in the database so as to allow determining the average number of items that can be sent back in a response listing the outcome of search query to the user. The resulting information will be analysed so that schemes of compressing the outcome without decreasing the contained information can be applied.

Expected Results

This test is expected to highlight situations where not useful information can be omitted from the responses (e.g. empty XML elements) and situations where the organisation of information can be improved so as to reduce message length. An upper limit is expected to be estimated that may signal the segregation of the result in more than one message. In such case the additional delays will be evaluated.

3.1.4 Overlay Management OM-SA

This section will contain the tests to be performed in order to check out OM's proper working

3.1.4.1 OM-SA performance under churn

Purpose of the Test

The dynamic nature of p2p networks poses several obstacles in the live streaming systems. This test focuses in the frequent arrival and departure of peers which phenomenon is called 'churn' and results in the degradation of quality of p2p services. The purpose is to determine the performance of scheduling and overlay management procedures under churn.

Procedure of Testing

During the execution of the experiment, the degradation of the stream distribution under 'churn' will be evaluated in contrast to a static overlay scenario. Specifically, the static overlay will include up to 100 peers while the scenario under 'churn' will use a uniform function for random arrivals and departures of peers between 1-5 seconds. Both scenarios will be performed for a fixed video duration and the metrics will include the control message overhead for overlay reorganization, the packet loss, the wasted bandwidth through duplicates and the unused bandwidth through the time that peers remain idle.

Expected Results

We expect that churn will affect the transmission of live streaming video but as proved in simulations, the overlay management algorithms will limit the degradation of the quality to minimum.

3.1.5 Transcoder Service

This section will contain the tests to be performed in order to check out Transcoder's proper working

3.1.5.1 Transcoding Performance vs. file size

Purpose of the Test

The purpose is to determine the performance of the transcoder service as the file size of the query increases.

Procedure of Testing

A request is made to the transcoder service in order to convert an audio file of N Kbytes into the available output formats. The total completion time for the operation is recorded. Files with increasing sizes are processed and the transcoding time is measured.

Expected Results

The total completion time for the operation increases with the file size.

3.1.5.2 Transcoding Performance vs. query's load

Purpose of the Test

The purpose is to determine the performance of the transcoder service as the query's load increases.

Procedure of Testing

A number of parallel queries request the transcoder service and the total completion time is recorded. The amount of simultaneous transcoding requests is increased and the completion times are measured.

Expected Results

The total completion time increases with the number of parallel queries.

3.1.6 DVB-T environment

3.1.6.1 DVB_T-TC1 (System Performance 1)

Name	DVB_T-TC1
Purpose	System Performance
Prerequisities	DVB-T environment working
Testing Procedure	In this experimental scenario, the users located within the rural area CMNs will request content hosted by the active-user located within the urban area CMN. The duration of this experiment will be five minutes.
Testing Date	25/11/2010
Expected results, if any	Evaluate the system performance under typical Client/Server applications, i.e. when no P2P overlay is formed during a content distribution procedure.
Any other involved topics, if any	N/A

3.1.6.2 DVB_T-TC2 (system performance 2)

Name	DVB_T-TC2
Purpose	System Performance
Prerequisities	DVB-T environment working
Testing Procedure	In the second experimental scenario the VITAL++ P2P modules will be activated in the regenerative DVB-T side and the corresponding VITAL++ P2P-client application at the end user's side. The following metrics will be taken : * utilization

	* Round trip time * Unique bytes sent over the DVB-T downlink vs. the number of active peers * ADSL uplinks utilization * Percentage contribution of each network segment in the content distribution procedure.
Testing Date	25/12/2010
Expected results, if any	Alleviate the DVB-T utilisation and enabling more balanced recourse exploitation among all network segments.
Any other involved topics, if any	N/A

4 Real User Experiments

Since P2P overlays typically involve hundreds of participants with a sizable rate of churn, it is important to check if the Vital++ is able to cope with these requirements. A battery of specific tests is required in order to assess how Vital++ will behave as:

- The number of overlay participants grow
- The churn rate increases.

As in the previous section, only the definition of the tests is included, since the results and its relevance are discussed in D5.3. Expected results, if any, are considered as part of the definition and therefore belong to D5.2.

4.1.1 Aim and Scope

Real user experiments are performed in order to evaluate the performance, the scalability and the sustainability of the Vital++ components in a real and dynamic environment. The main scope of these experiments is to test the data distribution mechanisms (scheduling, overlay management, transport protocols etc) in a controllable local environment (UoP laboratories), as well as in a completely dynamic environment, the UoP Campus. Both experiments involve up to 100 clients, different operating systems (Linux, Windows 7, Vista and XP) and different quality of the transmitted video. Also, the degradation of the provided service can be measured under 'churn', which is a critical phenomenon in all the existing p2p applications.

The experiments will provide concrete results relating to the implementation of the involved vital++ components and the interfaces between them. The metrics that are reported either real-time by the clients to the server or at the end of transmission, can be used for the overall assessment of the P2P client, while these metrics may lead us to identify the leaky parts of the implementation and fix possible bugs or redesign some modules. Moreover, the involvement of real users (students) may give us some feedback for the usability of the P2P Client and impose possible improvements in the future.

The P2P client includes the core distribution mechanism of vital++ architecture which is the P2PEngine.

The Overlay Management SA/Component will be used to organize the involved peers either at the bootstrapping of each scenario, or during the execution of the experiment.

The statistics can be visualized either real time in the P2P Client GUI or in the logging service which is part of the server side Overlay Management SA.

As far as the laboratory experiments are concerned, the P2P Clients will be hosted in a Virtual Machine. To achieve this goal we have set up a cluster of XEN servers that can support 6 Virtual Machines each. In VMs the linux (no GUI) version of the client will be executed, while in some other PCs of the laboratory the windows GUI version will be up and running. The Overlay Management server side component will be hosted in a separate linux PC.

During the UoP Campus experiments, the students will be invited to download, install and run the windows P2PClient. We intend to announce the transmission of live streaming video at specific dates and monitor the whole procedure.

4.1.2 Tests

The Laboratory experiments will be performed in three phases:

- Phase 1: Use of up to 100 VMs in Linux PCs.

  The purpose of this test is to involve a great amount of clients in the overlay in order to gather statistics related to scalability, bandwidth utilization, flow control, bandwidth waste and scheduler performance.
- Phase 2: Use of 10 Windows Clients.

  The purpose of this test is just to ensure the desired functionality of the windows version of the P2PClient.
- Phase 3: Mix of VMs and Windows Clients (up to 100).

  This is the overall experiment in the laboratory that combines both operating systems and as much clients as we can support in our premises.

The Real users' test will be the final and most demanding phase of the evaluation tests:

- Phase 4: Invite students to download, install and run the windows P2P Client.

  In the context of this test we plan to broadcast live streaming video (webcam, channel) as well as to broadcast some video files of different quality and different format.

The above tests are ideal for evaluation and measurement of some crucial parameters of the P2P streaming system.

More specifically, taking into consideration the underlying network during experimentation, we assume that the average upload capacity of the peers can support the service rate of the transmitted stream. In order to demonstrate the performance of the overlay management algorithms (neighbour allocation, transport mechanisms, control messages etc) we form a randomly created overlay where every peer obtains up to 8 neighbours and executes the above algorithms every second. We execute the same experiment by first inserting randomly the peers and then applying our algorithms. Then we can form the CDF of the energy of the participating peers and contrast the results with the graphs of simulations in OPNET as stated in the previous deliverables.

The evaluation of the scheduler performance is tested under the different scenarios and demonstrates its effectiveness by means of CDF of successful block receptions from peers. This graph depicts the bandwidth utilization that the P2P system can achieve. The CDF can also be formed by means of the duplicate packets that were received by peers. The percentage of the duplicates, which is translated to wasted bandwidth, along with the control overhead of our system, give us an upper bound on the maximum achievable service rate for a given distribution of upload capacities. Other performance parameters that can be measured are the percentage of the peer bandwidth that is exploited for control message exchanges, the time that a peer doesn’t contribute to each neighbours (idle time) and the number of block requests that a peer received which shows the efficiency of the token generation and the proactive block request algorithms that manage to distribute the task of block propagation evenly to every peer according to its uploading capacity.

Scalability and stability of the system is measured by demonstrating its behaviour for different number of participating peers. The expected behaviour has to be independent of the number of participants indicating a potential asymptotic behaviour.

Finally we evaluate the performance of our system under dynamic conditions (frequent arrivals and departures of peers during the transmission of the stream). The estimation is about 3% degradation of the provided services in comparison with static overlay conditions, which fact

may lead to the conclusion that the system is almost immune to these dynamic conditions due to the fast reorganization of our overlay and our dynamic neighbour selection function.

4.1.3 Tests

The capabilities of the proposed P2P aware DVB/IP networking environment will be validated by experimental tests that will be conducted under real transmission/reception conditions. The experiments will be carried out in two phases.

- Phase 1: The first experiment will focus on evaluating the system performance under typical Client/Server applications, i.e. when no P2P overlay is formed during a content distribution procedure.
- Phase 2: In the second experimental scenario, the P2P engine will be activated in the regenerative DVB-T side and the corresponding VITAL++P2P-client application at the end user's side.

Building upon existing system architectures for interactive broadcasting implementations, we create a VITAL++ P2P based network overlay in a regenerative DVB-T constellation. The P2P technology will be exploited for enhancing the system scalability and performance during personalised and on-demand services access.

5 Vital++ and PII integration

The Vital++/PII integration regards the process of preparing and installing the outcomes of Vital++ (at least those that constitute the core of the platform) as testbed offerings. The term testbed in the previous sentence regards the operation of an aggregation of equipment and hardware/software resources under the principles and techniques defined in the context of the PII platform.

5.1.1 Aims and Scope

Vital++ has produced a platform consisting of various software components installed in a number of testbeds offered by partners. All the components are brought together over a virtual network realised by the adoption of VPN techniques. Each component has been developed, prepared and maintained by one responsible partner. Among the components there are certain relationships that have been clearly identified and documented so that the proper integration among them was possible. The Vital++ platform therefore can be installed on any available testbed facilities. These characteristics have revealed the need for the existence of a setup framework that can install the platform on top of existing computing resources in order to ease the process of experimentation with and also demonstration of the produced outcomes. Key element in this remains the opportunity to test and collect feedback from the possible Vital++ deployment with respect to the capabilities of the P2P algorithms and techniques. In this essence, PII platform provides a very good approach for deploying instances of the Vital++ platform utilising probably existing resources such as IMS installation offering user accounts.

Having such an option available at the end of the project, Vital++ will be able to offer its outcomes to greater audience that may be already familiar with the PII methodologies. This will allow for demonstrating the Vital++ concept to people that may express such interest without the involvement of the Vital++ consortium. On the other hand the documented and defined associations among Vital++ components may give the opportunity to other people to develop their own solution and by using the PII installation of Vital++ validate the compliance of their work. Additionally, functionality offered by any of the Vital++ components might be of interest for be integrated in other service scenarios on top of or in parallel to Vital++.

5.1.2 Technical Details

The latest demonstration of Vital++ had involved a significant number of components and network architectures for realizing an extensive set of use cases. An indicative list follows:

- OpenVPN based VPN interconnections among testbeds including routing and DNS services offered by TID OpenVPN Server installation
- OpenIMS installation at Fokus in the role of the home domain for all subscribers
- OpenIMS installations at UoP, TA, TID in the role of visited domains
- P2PA-SA at Fokus
- CP-SA at WIT
- CI-SA and OM-SA at UoP
- Monster and BCT clients launched on demand in the demo room and UoP testbed
- Satellite and VLC equipped CTRC testbed for generation of media streams

- OpenSwan VPN tunnel between CTRC and UoP for delivering the media streams to the Vital++ platform
- Satellite Relay entity at UoP for publication and adaptation of the CTRC content in the Vital++ platform
- Mobile Relay entity at VG for adaptation of P2P acquired media content to regular SIP subscribers
- SoftMix Application Server at RBB offering metadata and media files
- SuperPeer at UoP for publication and adaptation of the content offered by RBB for delivery in the Vital++ platform
- Transcoding Service at TID for preparation of a number of alternatives of the RBB offered media

In the context of this task not all the above listed components will be made available via PII methodology, although the overall approach can be possible to be applied to any of the excluded components. The included ones are mentioned in the following paragraphs.

Network Setup and Server Side Components

It was decided not to cover the establishment of the VPN connections but to offer all the components via public Internet connections, so that the exact VPN establishment approach (if it is mandatory) is not limited to the techniques Vital++ has adopted.

The same applies to the foreign/home IMS domains approach that was used in the latest demonstration. There is no evident reason for reproducing the same architecture via PII methodology since the actual IMS topology should be left open and the focus should be put on the process of building on top and enhancing existing IMS installations.

Therefore the server side components are the four subarchitectures and one OpenIMS installation. The OpenIMS installation should be treated in a way that does not impose any specific requirements on the rest of the components so that other alternatives can be also selected.

Client Side Components

The selection of the client side components to be installed via the PII methodology depends on the actual use cases. It was decided that LiveStreaming and SoftMix should be the two scenarios to be presented.

Therefore there is need for a number of clients scattered around testbeds. Since the client is a piece of software designed for desktop and not background usage there is need for adapting it so that it can be launched and operate in an autonomous manner without requiring user intervention. This will be offered by a GUI-less mockup of the BCT client.

For SoftMix the presence of the SuperPeer is necessary. The SoftMix AS will be used from its default location and not be considered as a PII installable component.

Identification of Resources and Resource Adapters

Resources can be installed on demand or be preexisting installations for which RAs have been activated to cater for controlling the resource under the commands of the PII infrastructure.

In the context of this integration task the following approach is envisaged:

1. OpenIMS will be used as an existing resource offered by one of the testbed. An RA will be developed for it in order to allow for associating user accounts with client instances. This means that at least the realm and port number of P-CSCF should be reported by this RA.
2. P2PA-SA can be considered as a preexisting resource with an RA reporting its access details (SIP URI)

3. CP-SA can be installable via a Virtual Image on a VMware server. There is need for an RA for the VMWare server to handle the create command for the CP-SA. The Server will be preexisting resource reporting only the IP address of the installation. An RA is required for the CP-SA to expose the SIP URI of it to client instances. The IP address will be inherited from or allocated by the Server.
4. CI-SA and OM-SA can be bundled in a Virtual Machine that can be installed on a preexisting XEN or VMware server. CI-SA needs to know the CP-SA URI as well as a valid account on the OpenIMS. RA for Virtual Server, RA for CI-SA, RA for OM-SA.
5. The SuperPeer can be also bundled in Virtual Machine. An RA is required for configuring the Super Peer with the User Account as well as with the access details of the RBB SoftMix AS. Considering the fact that the SoftMix AS will be used from its current installation, the possibility to present this as a PII provisioned resource as well should be evaluated. In this case an RA will be required to cater for replying with the access details of the SoftMix AS in the relevant queries.
6. A number of GUI-less BCT clients acting according to a predefined scenario need to be installed in various places in the testbeds involved in this scenario. An RA will be required for this client instances as well. The client installation can be also done via Virtual Images, however, other solutions are also evaluated.

5.1.3 Validation Tests

Name	IMS_OT_1
Purpose	Validate that all involved entities register with the IMS included in the setup
Prerequisites	IMS RA, Client RAs
Testing Procedure	Launch a set of clients associated with the IMS and ensure that they register properly
Expected results, if any	Correct distribution of credentials and settings
Any other involved topics, if any	Proper routing and networking setup

Name	CI_CP_OT_1
Purpose	Validate that clients can publish media offerings along with the licensing information
Prerequisites	Client RAs, CI-RA, CP-RA
Testing Procedure	Verify that publication are performed correctly
Expected results, if any	Correct distribution of CI and CP contact information, proper publication
Any other involved topics, if any	Proper routing and networking setup

Name	CI_CP_OT_2
Purpose	Validate that clients can search for media offerings and acquire licensing material

Prerequisites	Client RAs, CI-RA, CP-RA
Testing Procedure	Verify that discovery of media and licensing is performed correctly
Expected results, if any	Correct distribution of CI and CP contact information, proper content indexing and licensing
Any other involved topics, if any	Proper routing and networking setup

Name	CI_OM_OT_1
Purpose	Validate that clients can join overlays
Prerequisites	Client RAs, CI-RA, OM-RA
Testing Procedure	Verify that CI interacts properly with OM and provides proper bootstrapping information to clients
Expected results, if any	Correct distribution of CI and OM contact information
Any other involved topics, if any	Proper routing and networking setup

Name	P2P_OT_1
Purpose	Validate that clients acquire content through the overlays
Prerequisites	Client Ras
Testing Procedure	Verify that clients can join the overlays and exchanged data through these
Expected results, if any	Proper bootstrapping and operation of P2P Engines
Any other involved topics, if any	Proper routing and networking setup

6 Summary

The aim of this deliverable was twofold. On one hand, to describe which services were deployed on Vital++ infrastructure, building on the uses case defined in D5.2, and highlight how they are used to demonstrate the validity of Vital++ concept. On the other hand, to define the tests required to achieve these validation aims.

The results of the defined tests and the conclusions ensuing from them fall outside the scope of this deliverable and will be presented and analised in D5.3.

The first section has been allocated to the description of the services deployed on the infrastructure described in D5.1.

Then, specific tests designed for the validation/assessment of each SA/architectural element implemented for Vital++ have been

A third section deals with experiments involving real users, which will help to assess the scalability/stability/performance of Vital++.

Finally, it has been laid out how Vital++ and PII are going to be integrated, to what extent this target will be pursued and which tests have been defined to validate that this integration has been achieved.

- End of document -