

What is OpenRate?

Open, Flexible, Free
Rating and Billing for the 21st Century

OpenRate placement

Orientation

Overview

- OpenRate is a mediation, rating and pre-billing Framework, based on the concept of “pipeline processing”
- A Framework in this context is a collection of medium to high-level processing functions, which can be used to tackle a complex task
- The framework guarantees the basic functions (e.g. moving records around, handling transactions), and lets you concentrate on the business level configuration
- Processing is performed by around 80-90% using standard modules, which are high level operations (e.g. rating, zoning). The other 10-20% will be created during the implementation
- OpenRate provides a “sweet spot” in the balance between flexibility and out of the box functionality. It helps you do most things, it stops you doing nothing.

OpenRate placement

Orientation

Overview

- You are at liberty to create new modules based on OpenRate's clear and open module structure, using standard, freely available tools,
- OpenRate offers high performance, usually in excess of 5000 CDR/second/CPU (batch) and 1000 CDR/second/CPU (real time)
- OpenRate offers short implementation time scales - solutions are direct, and workarounds (the bane of packaged software) are rare
- OpenRate offers a truly convergent processing model (both service convergence, real time and batch convergence)

OpenRate placement

Orientation

Target Clients

- OpenRate is aimed at IT-savvy organisations, who wish to use their IT resources to tackle business problems, rather than IT problems
- Generally our customers are one or more of the following:
 - Organisations who are making the switch from entirely “home-grown” solutions during IT consolidation after periods of growth
 - Start ups and other rapidly growing sectors, who need agility and speed
 - Organisations who have reached the limits of their current systems
 - Cost conscious and results oriented
- OpenRate is particularly useful in situations where innovative market offerings are required. A framework allows you to adapt what you already have, whereas a traditional product does not

OpenRate placement

Orientation

What it's not

- OpenRate is not currently a billing solution - it handles mediation, rating and pre-billing (usage statement) calculations.
- Integrations with many billing platforms are available
- In spring 2009 we are launching a project to create a carrier grade billing framework to twin with OpenRate
- OpenRate is not a rigid “product” - it is a tool kit which can be rapidly configured to handle any reasonable processing scenario
- OpenRate is not a closed product - it is entirely Open Source
- It's not a free for all - all changes to the framework are controlled by the core team, and checked for coherence with the roadmap: Governance is key to OpenRate

OpenRate Key Concepts

Features and Key Concepts

OpenRate feature overview

Concepts

Framework

- OpenRate is a mediation, rating and pre-billing **Framework**, based on the concept of “pipeline processing”
- **Pipelines** are chains of processing **plug-in** modules
- Each pipeline has an **input plug-in**, some **processing plug-ins** and some **output plug-ins**

Rationale:

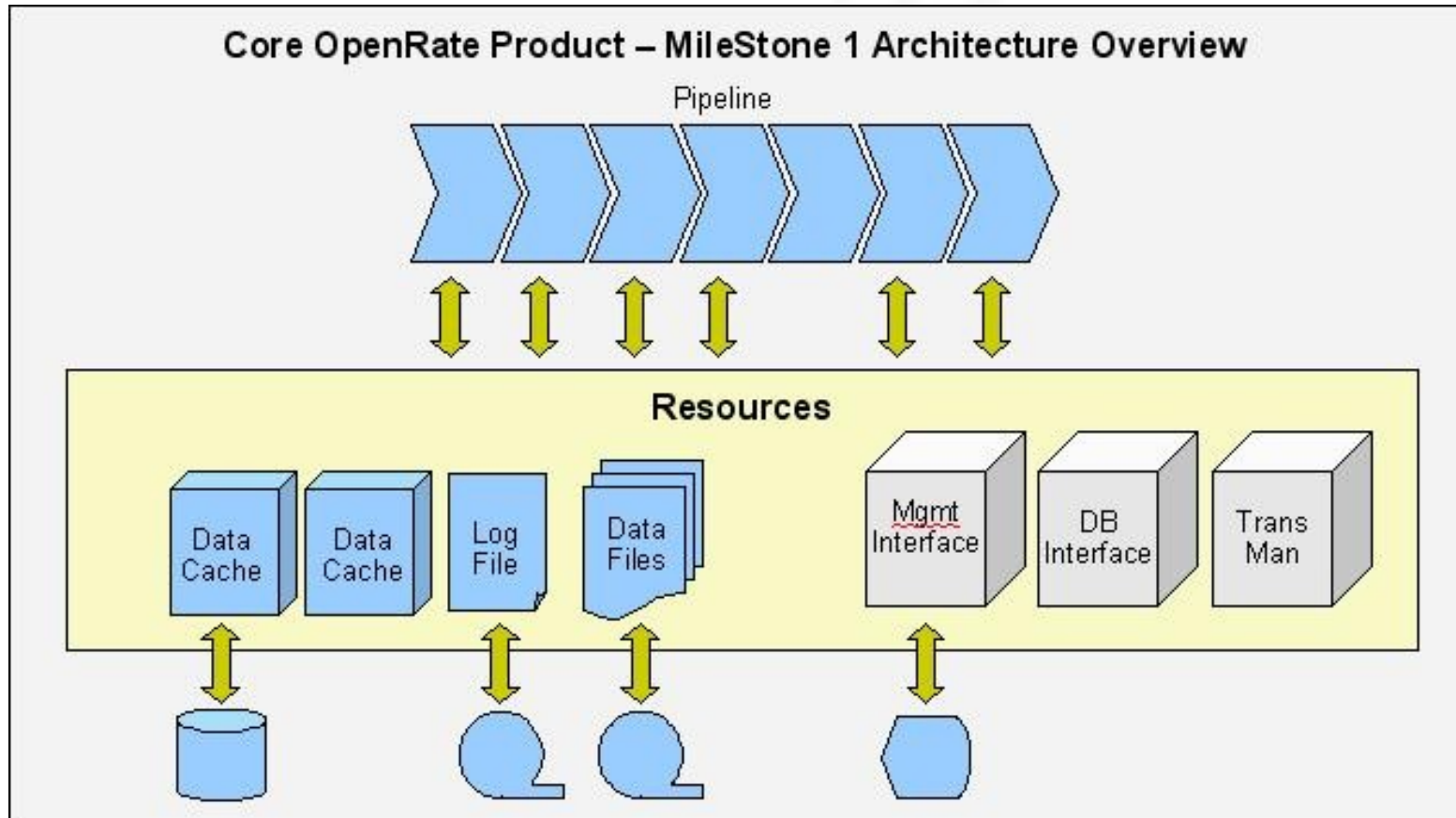
- Most innovative processing solutions mostly require new combinations of existing techniques, plus a few new techniques
- Expressing problems in terms of high level building blocks (plug-in functions) greatly aids the analysis of complex processing

OpenRate feature overview

Concepts

Framework

OpenRate Framework, showing the key elements



OpenRate feature overview

Concepts

Records

- **Records** pass through the pipelines, each plug-in either creating (input), transforming (processing) or persisting (output) records.
- A record is a fully-fledged object in OpenRate. It has an identity and encapsulates all information needed for processing. It knows nothing about the record that preceded it or the record following it
- Information about the state of the processing is provided by **Framework Resources**
- Resources hold data that can be used by records (**Data Resources**) and system services such as logging (**Service Resources**)

OpenRate feature overview

Concepts

Resources

- Data Resources are either **lookup** resources (read only) or **modifiable** resources (read-write)
- All lookup resources can be loaded with information on framework start-up
- Usually modifiable resources will be saved on Framework shut-down
- All resources can be managed at run time, e.g. reloading new configurations or saving current results
- Resources can be loaded from/saved to files or databases

OpenRate feature overview

Concepts

Transactions

- All processing is controlled in **transactions**. Transactions enclose a complete processing stream (files or the contents of a table)
- Each pipeline in the framework has its own **Transaction Manager**.
- The Transaction Manager is responsible for checking the processing status of each plug-in in the pipeline. If any plug-in reports a critical processing error, the whole transaction (e.g. File) is rolled back.
- Only when all plug-ins report that the stream has been closed without error, will a commit happen

OpenRate feature overview

Concepts

Errors

- Non-critical errors are handled by attaching **Errors** to Records.
- The record is marked as invalid (“Errored”) if it has at least one error attached to it.
- Invalid records pass through a parallel stream in the pipeline - they normally do not pass through the main processing path
- However, they can be recovered at any point in the processing, if this should be required
- You can decide at any point to abort a whole transaction

OpenRate feature overview

Concepts

Threads

- Each plug-in in each pipe is created as a separate thread group in the Framework. Only input plug-ins are attached to the Framework thread, all others are free-running threads
- This means that the OpenRate Framework is heavily threaded, and easily exploits exotic hardware
- Any plug-in can be configured to use any number of threads. By default 1 thread is used for each plug-in
- A system of monitors is used to wake threads up when records arrive to be processed
- In this way, records are “sucked” through the pipe

OpenRate feature overview

Concepts

Performance

- To enable the high speed processing, a FIFO (First In First Out) buffer is placed between each plug-in
- Using these techniques, OpenRate reaches extremely high levels of performance
- Even on single processor machines, OpenRate routinely manages to rate records using non-trivial processing at around 5000 records/second (18Million/hour) PER CPU.
- OpenRate therefore provides a world class scalable high performance rating engine. Even expensive commercial products have extreme difficulty reaching this level of sustained performance.

OpenRate feature overview

Concepts

Plug ins

- Each Plug-in is based on the concept of a “processing stack”
- Each layer in the stack performs a well defined function
- A new plug-in can be made, “branching off” from any level of the stack, preserving all basic functionality lower down the stack
- The stack is organised in such a way that the basic functions of the processing (e.g. moving records around) are at bottom of the stack
- Only high-level, business relevant functions are exposed at the top of the stack

Result

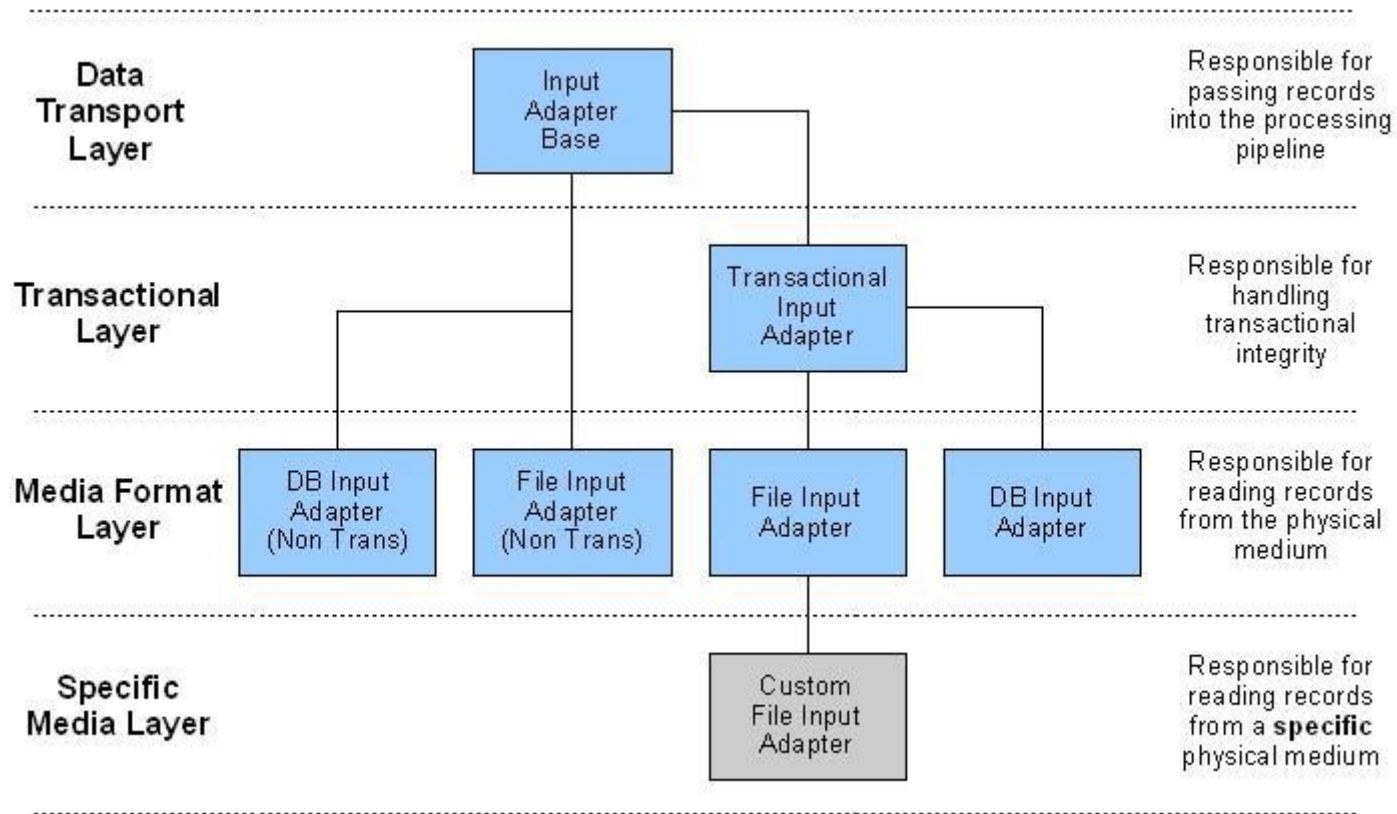
- You concentrate on your business challenges, not code

OpenRate feature overview

Concepts

Module stack

Example: Input Adapter Family



OpenRate feature overview

Concepts

Module stack

- The “Data Transport Layer” handles moving records into the pipe
- The “Transaction Layer” handles the transaction, opening a new transaction when a new stream starts, and committing at the end
- The “Media Format Layer” tells the adapter how to read **any** file, or **any** DB
- The “Specific Media Layer” adds the information about **your specific** file or DB
- Normally you will never have to go into the first three layers

OpenRate feature overview

Concepts

Logging

- The OpenRate framework has advanced logging built in, using the extremely efficient “log4j” library
- There is a log for the framework, one for each pipe and one for statistics and performance information
- Logging levels are “ERROR”, “WARNING”, “INFO” and “DEBUG”
- Logging levels can be set for each log, or even *each plug-in* separately
- Logs can be rotated/appended/overwritten at will

OpenRate feature overview

Concepts

Run time

- The OpenRate framework can be managed during run time using a dedicated interface, over telnet or via a GUI
- Some (but not all) of the configuration options can be changed while processing is occurring (“Dynamic”)
- Changes to configurations will either happen immediately, or between transactions, depending on the type
- Data caches can be reloaded - in this case, the loading is done between transactions (“synchronised”)

OpenRate feature overview

Concepts

Configuration

- Configuration is performed using an XML file that defines all of the modules to include in the environment, and how to build the pipelines
- A configuration GUI is in creation at the moment, that will support the whole configuration process
- The configuration GUI will allow also the entry, import and export of configuration data (e.g. zoning data)

OpenRate feature overview

Concepts

Database

- The framework provides pooled connection handling for virtually any sort of database using JDBC, e.g. we have direct experience in the following DBMSs:
 - Oracle
 - MySQL
 - Postgres
- The framework opens and manages the connections transparently, so you only have to make reference to the DB name
- Connection pooling provides high performance

OpenRate feature overview

Concepts

Hardware

- OpenRate can run on any hardware supported by a JRE (and that is virtually every contemporary hardware)
- Tested on multi-processor and 64bit architectures (Solaris, HPUX, Linux) up to 3 million customers
- Because of the multi-threading built into the framework, scales linearly onto multi-CPU machines

OpenRate Functional View

A walk through of an example application

OpenRate - Rating Calculation

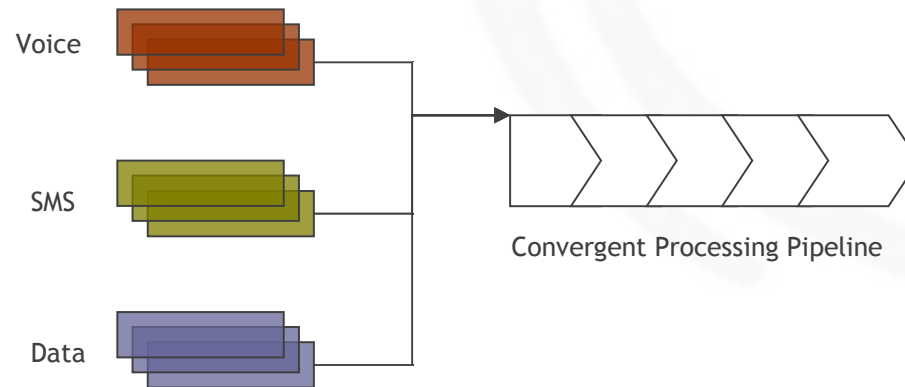
CDR Input Structure

Rated CDR Structure

Rating Process Monitoring

Features

- OpenRate CDR input processing is flexible and can handle input from any file or Database
- Independent record mapping based on record type/scenario, optionally normalizing the data into a single internal format
- Records can be categorized for mapping using flexible recognition on any field or field groups
- Handles sub-records (for logical records that cover more than a single item of an input stream)
- Allows easy and efficient mapping of convergent data flows.
- E.g. in the current implementation performs convergent handling of Voice, SMS and Data



OpenRate - Rating Calculation

CDR Input Structure

Rated CDR Structure

Rating Process Monitoring

Examples: Input Voice File containing 2 record types

```
00;20070810184900;OpenRate;1
01;;;;;3101500000000001;00393405158129;;;;4;4;00912342323;;;;11;17;20070810220000;120;;;;4;;;;;
02;;;;;00912342323;3101500000000001;;00393405158129;;11;17;20070810220000;120;;;;4;4;;;;;
99;20070810184900;OpenRate;1
```

- “00;” is a header record
- “01;” is a voice originating (MOC) record
- “02;” is a voice terminating (MTC) record
- “99;” is a trailer record
- All these record types are normalised by the mapping into a single internal format

OpenRate - Rating Calculation

CDR Input Structure

Rated CDR Structure

Rating Process Monitoring

Internal Structure

- The internal structure of the record provides extremely rich information, and can be extended at will to include any additional information that is necessary
- The “engine room” functionality are provided in parent class implementations that are usually not part of the implementation effort, and are therefore not changed
- The custom implementation happens on user level inherited classes, thereby providing simplest possible interface for implementation, without risking adverse impact on the base functionality
- Nearly all important information is already exposed in the internal structure (e.g. rating breakdowns, zoning information, timing information, interconnection and airtime costs)
- Any further information required can be added with little effort
- Key elements in the standard rated CDR structure are:
 - Subscriber information
 - Products that the subscriber owns, including validity dates
 - “Charge Packets”, containing a detail breakdown of each products contribution to the rated total
 - “Balance Impacts”, which roll up the rated effects on a given resource (rating can impact any counter in the system, not just currency)
 - Error lists, with detailed error information and error location
 - Output lists, which provide information about the output locations to which the CDR should be sent

OpenRate - Rating Calculation

CDR Input Structure

Rated CDR Structure

Rating Process Monitoring

Example

- An example of the information for a rated SMS record:

===== DETAIL RECORD =====

```
Record Number   = <0>
Record Type     = <MOC>
A Number       = <00393405158129>
B Number       = <00912342323>
CDR Start Date = <1186776000>
Duration       = <120.0>
Events         = <0.0>
Volume        = <0.0>
IMSI          = <3101500000000001>
--- Scenario Attributes ---
Service        = <VOICE_ORIG_INTL>
Direction     = <Originating>
RoamingType   = <Domestic>
PreZoneType   = <International>
```

Basic CDR Information

```
Charge Packets = <1>
----- Charge Packets -----
Rate Plan Name = <P.MOC.INTL.1>
Packet Type    = <A>
Service        = <VOICE_ORIG_INTL>
Priority       = <0>
RUM            = <DUR>
RUM Value      = <120.0>
Time Model     = <FLAT>
Time Result    = <FLAT>
Zone Model     = <BZM.1>
Zone Result    = <BWorld>
Price Group    = <75C_60s>
Price Model    = <75C_60s>
Cost           = <1.5>
-----
```

Charge Breakdown Information

```
----- Customer Attributes -----
Guiding Key    = <3101500000000001>
MSN           = <00012012>
SubscriptionID = <SUB1>

Rate Plans    = <1>
----- Rate Plans -----
Prod Name     = <P.MOC.INTL.1>
Subscription = <SUB1>
Service       = <VOICE_ORIG_INTL>
Valid From   = <1199142000>
Valid To     = <1230721201>
-----
```

Customer & Product Information

```
Bal Impacts = <1>
----- Balance Impacts -----
Rating Impact = <EUR>
Impact        = <1.5>
Counter ID    = <978>
Counter Rec ID = <0>
-----
```

Charge Breakdown Information

```
Errors = <0>
```

Error Information

OpenRate - Rating Calculation

CDR Input Structure

Rated CDR Structure

Rating Process Monitoring

OpenRate offers a sophisticated run time monitoring and control interface

- Necessary as OpenRate is designed as an “always on” framework
- Can be used either as a Command Line interface via Telnet (often preferred by operations staff), or a GUI
- Allows monitoring of throughput, thread allocations, loading, performance bottlenecks
- Allows the modification at run time of configuration information, e.g. to temporarily halt a pipeline, while leaving others running
- Allows synchronised (to ensure transactional integrity) loading of configuration information
- Allows purging or writing back of in memory information
- Allows enquiry of any configuration option in a running environment
- Provides a script interface to allow the integration with existing environments and operations
- Provides immediate responses, even a high load times

OpenRate also has Detailed logging

- Logging is performed for
 - the Framework
 - Each pipeline
 - Overall statistics
- It uses various levels of logging:
 - Debug (developer information)
 - Info (generally interesting stuff)
 - Warning (non fatal errors)
 - Error (unrecoverable errors)

OpenRate - Rating Calculation

CDR Input Structure

Rated CDR Structure

Rating Process Monitoring

Example

- An example of a simple interaction session via telnet

```
-----
OpenRate Admin Console, Release 0.1, 17-03-2006
Copyright Tiger Shore Management Ltd, 2006
-----
openrate> c
OpenRate command listing:
-----+-----+-----+-----+
| Command                | M | D | S |
-----+-----+-----+-----+
|ServiceCache
| - ServiceCache:GroupCount      |   | X |   |
| - ServiceCache:ObjectCount     |   | X |   |
| - ServiceCache:Reload          |   | X | X |
|RatePlanSelCache
| - RatePlanSelCache:GroupCount   |   | X |   |
| - RatePlanSelCache:ObjectCount  |   | X |   |
| - RatePlanSelCache:Reload      |   | X | X |
|AZoneLookup
| - AZoneLookup:Threads           |   |   |   |
| - AZoneLookup:BufferSize        | X |   |   |
| - AZoneLookup:BatchSize         | X |   |   |
| - AZoneLookup:StatsReset       |   | X |   |
| - AZoneLookup:Stats             |   |   |   |
...
-----+-----+-----+-----+
openrate> ServiceCache:ObjectCount
10
openrate> AZoneLookup:BatchSize
5000
openrate>
```

OpenRate - Rating Calculation

CDR Input Structure

Rated CDR Structure

Rating Process Monitoring

Example

- An example of a simple interaction session via the GUI, used to monitor performance statistics.
- The “Current Value” shows the performance information:
CDR processed:ms processing time:files processed:CDR/s average:input buffer:buffer limits hit

Open Rate GUI - System Monitoring

File Actions Configuration Help

Module Name	Class
APNCache	OpenRate.cache.RegexMatchCa
APNLookup	PatniRatingScenarios.APNLooku
AZoneCache	OpenRate.cache.BestMatchCac
AZoneLookup	PatniRatingScenarios.AZoneLoo
BLInpAdapter	PatniRatingScenarios.BalInputAc
BOOutAdapter	PatniRatingScenarios.BalanceO
BZoneCache	OpenRate.cache.BestMatchCac
BZoneLookup	PatniRatingScenarios.BZoneLoo
BallLoader.TM	OpenRate.transaction.Transacti
BallLoader	OpenRate.Pipeline
CacheFactory	OpenRate.cache.CacheFactory
CustomerCache	OpenRate.cache.CustomerCac
CustomerLookup	PatniRatingScenarios.CustLooku
DBOutputAdapter	PatniRatingScenarios.DBOutput

Module Name	Command Na	Current Value	Mandator	Dynamic
GatherImpacts	Stats	20536:156:604:131641.0:0:0		
AZoneLookup	Stats	20536:127:604:161700.0:0:0		
TimeLookup	Stats	20536:265:604:77494.0:0:0		
BOOutAdapter	Stats	20536:1481:604:13866.0:0:0		
BLInpAdapter	Stats	10746:5443:0:1974.0:0:0		
TPInpAdapter	Stats	20536:44038:0:466.0:0:0		
BZoneLookup	Stats	20536:155:604:132490.0:0:0		
RatingPrep	Stats	20536:444:604:46252.0:0:0		
CustomerLookup	Stats	20536:2022:604:10156.0:0:0		
RatePlanSel	Stats	20536:1488:604:13801.0:0:0		
PBZoneLookup	Stats	20536:133:604:154406.0:0:0		
TPOutAdapter	Stats	20536:1339:604:15336.0:0:0		
DumpFirst	Stats	20536:2250:604:9127.0:0:0		
VZoneLookup	Stats	20536:122:604:168950.0:0:0		

The framework is up

OpenRate - Rating Calculation

CDR Input Structure

Rated CDR Structure

Rating Process Monitoring

Example

•An example of a Framework log file

```

2007-12-06 22:26:56,033 DEBUG - Registered Client <LogFactory>
2007-12-06 22:26:56,035 DEBUG - Registered Client Service <Reload>
2007-12-06 22:26:56,039 DEBUG - Command Port handled
2007-12-06 22:26:56,039 DEBUG - Command MaxConnection handled
2007-12-06 22:26:56,114 DEBUG - Creating new DataSource.
2007-12-06 22:26:56,114 INFO - Creating DataSource <TestDB> using driver <com.mysql.jdbc.Driver> from URL
<jdbc:mysql://localhost/PatniTestDB>
2007-12-06 22:26:56,166 DEBUG - jdbc driver loaded. name = <com.mysql.jdbc.Driver>
2007-12-06 22:26:56,180 DEBUG - poolableConnectionFactory built = org.apache.commons.dbcp.PoolableConnectionFactory
2007-12-06 22:26:56,181 INFO - Successfully created DataSource <TestDB>
2007-12-06 22:26:56,186 INFO - Starting CacheFactory initialisation
2007-12-06 22:26:56,193 INFO - Created Cacheable Class <ServiceCache>
2007-12-06 22:26:56,194 INFO - Starting cache loading for <ServiceCache>
2007-12-06 22:26:56,194 DEBUG - Found Cache Data DB <TestDB> for cache <ServiceCache>

```

•An example of a statistics log file

```

2007-12-06 23:00:45,010 INFO - Transaction <1236> closed
2007-12-06 23:00:45,010 INFO - Statistics: Records <87061>
2007-12-06 23:00:45,010 INFO -           Duration <35723> ms
2007-12-06 23:00:45,010 INFO -           Speed <2437> records /sec
2007-12-06 23:01:00,286 INFO - Transaction <1237> closed
2007-12-06 23:01:00,286 INFO - Statistics: Records <48032>
2007-12-06 23:01:00,287 INFO -           Duration <14232> ms
2007-12-06 23:01:00,287 INFO -           Speed <3374> records /sec

```

OpenRate - Road Map

OpenRate Mission

Roadmap

Objectives

- Flexibility
- Performance
- Service model, not license model
- Cooperative developer network
- Fully featured open-source mediation/rating and billing platform
- Part of a family of BSS projects

Flexibility

Concepts

Flexibility

- **OpenRate is Commercial Open Source.** We do not believe in vendor lock-in, and OpenRate is a clear, open, architecturally mature platform that aims to have no secrets. This means that your IT is in your control. Of course, we provide a full set of training and support services which mean you are never left out in the cold.
- **OpenRate is extensible and modular.** Most software evolution makes use of 80-90% of existing functionality, modifying only 10-20%. The modular OpenRate architecture allows you to maximise re-use, shortening development times and time to market, and reducing risk.
- **OpenRate uses only standard tools and objects.** OpenRate uses standard tools and is based on open code and libraries. This means that your IT organisation can easily introduce and use OpenRate, and quickly be productive with it.
- **Great Support.** We provide a range of services (consulting, training, implementation, Service Agreements) to exploit all of the flexibility that is built into OpenRate. You'll never be left out in the cold.

Performance

- Compared to the nearest comparable commercial system:
 - OpenRate gives you double the speed
 - OpenRate uses half the CPU
 - The OpenRate architecture does not force you to do workarounds, and that saves performance
 - Massively parallel threading model to extract all the CPU available

License Model

Concepts

License

- The OpenRate business model does not rely on license fees, instead our tried and tested business model (we are a company with 10 years experience) aims to supply support, training and implementations to make money.
- The license guarantees that your OpenRate agreement will not change status
- We provide expert support to you and your clients to ensure the success of your project
- Consultancy and services are on an “opt-in” basis
- The license is innovative, and lets you tailor it to your needs.

Cooperative Developer Network

Concepts

Community

- We provide expert product support using a network of high calibre professionals
- The support team are paid on results, not on chair warming
- You will have full access to the network
- You will have a say in the core module promotion process and road map

OpenRate Road Map

Concepts

Roadmap

- ~~MileStone 1 (V0.5) “As good as Integrate”~~
- ~~MileStone 2 (V0.6) “Operability”, “Complete Rating”~~
- ~~MileStone 3 (V0.7) “Pre-paid convergent”~~
- ~~MileStone 4 (V0.8) “Billing”~~
- ~~Milestone 5 (V0.9) “Integration”~~
- ~~Milestone 6 (V1.0) “Fully Convergent Billing Solution”~~

(Strike through indicates milestone reached)

Roadmap - MileStone 1

- MileStone 1 (V0.1-V0.5, Jan 2006-Sept 2008)
- Build and test the framework
- “To be as good as Integrate”
- To provide additional performance
- To provide additional flexibility
- To be provided under a free of cost license

Realised in OpenRate V0.5

Roadmap - MileStone 2

- “Operability” (V0.6, Mar 2007 - Oct 2008):
- Provide operations and monitoring GUI to allow fine control of the application
- Provide prototype configuration GUI to allow easy configuration
- Web services/SOA interface



Realised in OpenRate V0.6
(Configuration GUI Ongoing)

Roadmap - MileStone 2

- “Complete rating” (V0.6, May 2007 - Oct 2008):
- Convergent rating for all postpaid scenarios (mobile, data, fixed line)
- Group discounting and fee roll-up
- Rich plug-in population for standard tasks (e.g. zoning, rating, aggregation)

Realised in OpenRate V0.6

Roadmap - MileStone 3

- “Real time Convergent” (V1.0, Jul 2008 - Dec 2008):
- Integration between real time and batch processing scenarios
- Provide configuration GUI to allow easy configuration



Realised in OpenRate V1.0

Roadmap - MileStone 3

- Generally Available Version 1.0 (Jan 2009):
- GA release of production tested software
- Production implementations in daily use since June 2008
- Proven maturity



Realised in OpenRate V1.0

Roadmap - MileStone 4

- “Billing Solution” (V2.0, Autumn 2009):
 - SOA Framework Component
 - Front End integration with best of breed CRM (e.g. Sugar CRM)
 - Subscription/Purchase fee management
 - PDF Invoice generation
 - Bill cycle management
 - Uses OpenRate as the core rating engine
- Where you want it to go

Roadmap - MileStone 5

- “Revenue Management” (V2.1, Spring 2010):
 - Taxation / General ledger
 - Accounts receivable
 - Interfaces to best of breed CRM and Financial tools
 - XML Invoice interface for print shop output
- Where you want it to go

Roadmap - MileStone 6

- “Fully Convergent Billing Solution” (V3.0, Date Unknown):
 - Real-time Balance Manager component
 - Triple-A and integration with best of breed AAA protocols (Diameter, Radius, IPDR)
 - Integration with trouble ticketing and provisioning systems
- Where you want it to go