



Mathematical modelling, detection and classification.

---

# *DETECT*

*Detection and classification of fraud*

## Product Description ( Credit Card Fraud )

**Oscar Kilo Ltd**

T: +44 (0)1237 451327

E: [enquires@oscarkilo.net](mailto:enquires@oscarkilo.net)

W: <http://www.oscarkilo.net>

## 1 Introduction

Credit Card fraud is a rare event. The incidence varies from bank to bank and country to country but on average it represents less than 0.1% (EU average is about 0.08%) of all transactions and accounts for about 0.2% of the total value of transactions.

But although rare, its impact is far from insignificant. In the UK alone, losses amount to £500M per year ( APACS, <http://www.apacs.org.uk> ).

The total value of Credit Card transactions per annum for even a small Issuer is often hundreds of millions of Euros. The low cost of ownership of *DETECT* means that, given an estimated loss of 0.2%, even the smallest bank would see benefit from *DETECT*, which would pay for itself very rapidly.

*DETECT* is a new product based on many years experience of classification and optimisation problems and the particular problem of credit-card fraud detection. Its risk-engine is based on a unique algorithm that delivers high-detection performance. As well as addressing the central problem of fraud detection, *DETECT* has been designed for ease of use, low care and maintenance, and as such provides a cost-effective , non-invasive solution, with a remarkably lost cost of ownership.

*DETECT* can be deployed to address fraud as experienced by Issuers, Acquirers, Payment Processors and Merchants.

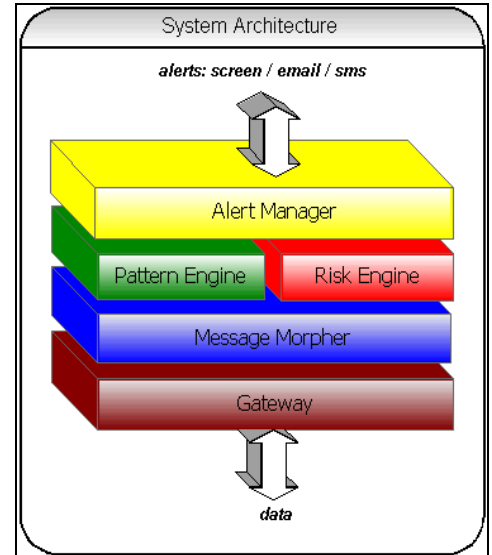
Some of the benefits of *DETECT* are:

- *Low Total Cost of Ownership* Fraud detection needs to pay for itself. The return on investment should be measured in weeks, not months and years, and the benefits should be immediate.
- *Alert Management* allows you to control many aspects of the fraud management process:
  - ✓ Alert rate                      to optimise use of staff or maximise value of alerts
  - ✓ Alert ranking                 fast access to the current highest risk cases
  - ✓ Risk-based routing          route alerts to different personnel via a choice of channels
  - ✓ Choice of risk metrics       financial risk exposure, probabilistic... to suit your risk policy
  - ✓ Transparent reporting      provides full explanation of alerts to aid case investigation
- *Case Management Tools* for risk assessment support
  - ✓ Account histories            single click access to full account history
  - ✓ Annotations                 record notes and comments for individual alerts or cases
  - ✓ Case allocation              allocate cases to users
  - ✓ Case sensitivity              increase or decrease risk level for individual cases
- *Immediate Results.* *DETECT* is provided with preconfigured fraud models that allows it to start detecting fraud immediately without the need for calibration or data-tagging.

## 2 System Architecture

*DETECT* is designed to be easy to install and does not rely on any third-party software that is not included with the software:

- It is a 100% Java application that includes its own Java Runtime Environment (JRE) so there is no need to worry about your currently installed versions of Java. It cannot interfere with other software and their supporting libraries.
- It contains its own embedded database (IBM's Cloudbase) for internal use.
- It contains its own web-server for the delivery of pages to the browser-based user interface. This can deliver pages to PC-based – and if required, hand-held devices.



Other benefits that result from the architecture of *DETECT* include:

- **Cross Platform** *DETECT* is a 100% Java application and will therefore run on any platform that supports Java 5. (Solaris, HP-UX, AIX, Linux, Windows Servers, ...)
- **Data Abstraction** *DETECT* incorporates a data abstraction layer that isolates the data sources from the internal logic. This means it can be easily configured to connect to and use almost any database.
- **Configuration** Configuration is very simple. In most circumstances even setting up the transaction protocol is simple as *DETECT* contains a highly configurable protocol handler and message morpher. Support is also provided for ISO8583.
- **Robust** *DETECT* has been designed so that it can recover from most types of failure of hardware and other software without the need for time-consuming, and therefore costly intervention by IT staff.
- **Scalability** The architecture of *DETECT* allows the system to scale as the transaction throughput increases. Very high transaction rates are possible even on low-range machines.

### 3 Transaction Delivery

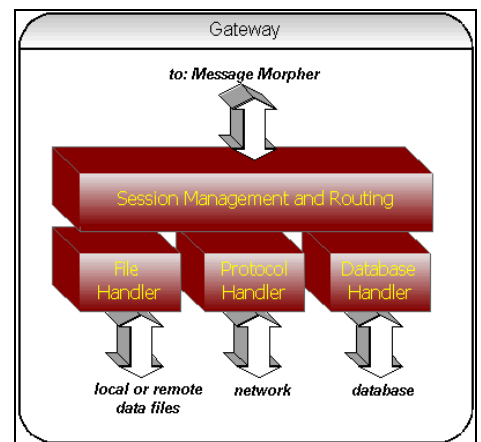
*DETECT* supports four modes of transaction delivery. This means that for the majority of installations the system can be configured to integrate into your current IT infrastructure without any impact on your current applications. It can be re-configured as your business processes evolve and change.

#### 3.1 Via TCP/IP

The transaction is sent over a LAN or WAN. Transactions can be sent and processed in batches.

There are four modes of message exchange:

1. *Pass Through Single*. A transaction is received, processed and then passed on with additional fields indicating its classification. (Synchronous)
2. *Pass Through Batch*. A set of transactions is received, processed and then passed on with additional fields indicating its classification as they are processed. (Asynchronous)
3. *Receive Response Single*. A transaction is received, processed and then details of the classification sent back as a response. (Synchronous)
4. *Receive Response Batch*. A set of transactions is received, processed and then details of the classifications sent back as a single response. (Asynchronous)



#### 3.2 Via Connection to a host database

A transaction is read from a database. The database table can be polled at a configurable interval or the availability of a transaction signalled via a bespoke mechanism (such as a Java stored procedure and database trigger).

#### 3.3 Via a flat file

A set of transactions is read from a file. The directory is polled at a configurable interval. The directory can be either local or remote. When it is remote the file is FTP'ed back to the local server and deleted from the remote server. Similarly, results are FTP'ed to the remote server.

Three file formats are supported as standard:

1. CSV – Comma Separated Variable
2. XML – Xtended Markup Language
3. FWC – Fixed Width Column

Each of the above formats can be configured to accommodate most requirements and can contain any number of transactions.

#### 3.4 Via Remote Method Invocation

This allows you to integrate *DETECT* into your existing applications

## 4 Message Morphing

Each message that is received passes through the Message Morpher that can completely change the structure of a message, if required. This means that you do not have to worry about the format of the messages sent to *DETECT*. By setting up a simple configuration file, messages can be transformed into the required format. The Message Morpher provides mapping down to the bit-level.

The Message Morpher is also used by the system to compress the data received, which it then stores in a cache. Data caching is so efficient that millions of transactions can be stored in memory, allowing users unprecedented rapid access to the transaction history. This is especially useful for pattern detection (see next section).

The core functionality of the Message Morpher can be enhanced with the use of *adaptors*. These are small light-weight Java classes written to perform a particular mapping task. Adaptors can be used to add extra information to the basic transaction. Some simple examples would be:

- Country groupings or Regions
- Latitude and Longitude
- Local Time-of-Day
- Local Day-of-Week
- MCC Groups
- Card-Limits

Other information can be added to the transaction via interfaces to host systems like for instance the Card Management System.

## 5 Detection Patterns

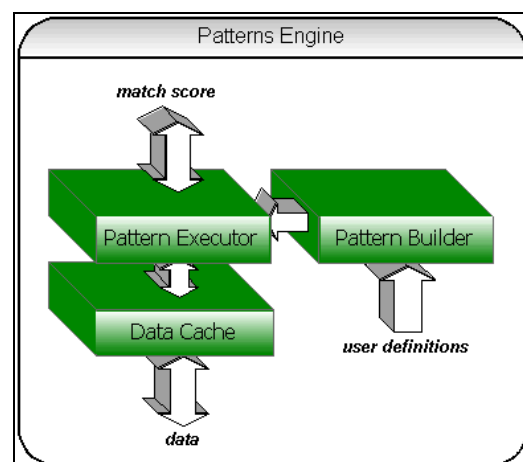
Detection Patterns are a powerful tool that allows the use of your expertise and understanding of the types of fraud experienced to capture fraud as early as possible.

Some simple examples of Patterns are:

- all the transactions on a particular merchant above €1,000.
- count the number of restaurants visited by cardholders in the last 24 hours.

The first example may be because of information regarding a merchant and a wish alerted whenever there is a transaction.

The second illustrates how patterns can be used to detect 'abnormal behaviour'. This also illustrates patterns can be used over a 'window' of time. As mentioned in the section above on the Message Morpher, the system uses a data-cache so that it can perform complex calculations based on the transaction history for the current account.



to be  
to  
how

The Patterns Engine includes a comprehensive set of built-in functions for calculating statistical values over a window or simple functions for manipulating data. The system includes a mathematical expression parser that allows the user to combine these functions and expressions.

Patterns can also be lists, which means you can use them to detect such things as whether a card is referenced in a hot-card list or the post-code is in a black-spot.

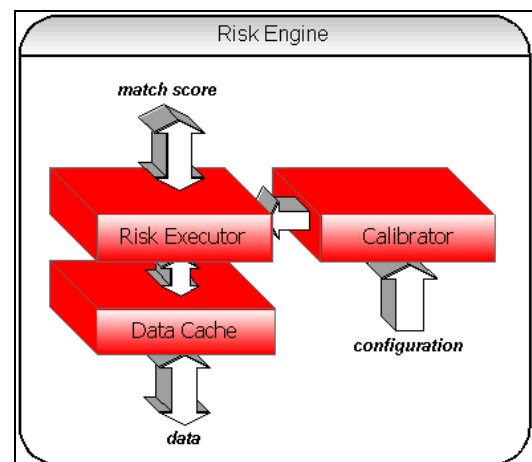
The flexibility of patterns allows you to detect very subtle features in the event stream, and while patterns are very powerful the user interface has been designed to be intuitive and easy to use. (Some example screenshots are shown at the end of this document.) The system is provided with a set of patterns to get you started and which adapted or built on. This enables the system to detect fraud as soon as it is installed.

## 6 Risk Engine

The Risk Engine computes the risk of fraud for each transaction based on historic data. It uses a sophisticated algorithm based on many years' research in both fraud detection and related areas that require the detection of rare-events.

The rarity of the fraud presents two particular problems:

1. How to deal with very sparse data, where there may be only one or two exemplars of fraud in a particular class of transaction. (Rare-event detection is a problem that crops up in many disciplines from nuclear reactors to pathology test results.)
2. How to combine the output of each test, rule or variable to produce an overall assessment of the risk of fraud. (Combining the output of different sensors is a problem that occurs in robotics, and is a particular issue associated with hybrid systems.)



Both of these problems are known to be technically difficult and different systems attempt to address these issues by using a wide variety of techniques, but usually with only partial success. The Risk Engine of *DETECT* uses techniques based on research in mathematics, robotics and artificial intelligence that exploit the rarity of the event.

Such systems require training (also known as calibration), which can involve a great deal of time and effort. The *DETECT* system uses incremental calibration, which allows it to automatically roll the calibration forward without any operator intervention.

This means that once the system is installed and configured there is little need for further day-to-day management, other than the processing of the alerts. Data tagging is important for the Risk Engine to be effective; this can be aided by the SMS Customer Alerting facility (see next section).

The Risk Engine derives transaction variables from the current and historic data. Just a few examples are:

- Rate of Spend – value per time period
- Implied Speed – time and motion between transactions

- Repeat transactions
- Transaction Rate – number per time period

It uses these and many other variables and the thousands of possible combinations to identify rare fraud events from the large volumes of data. The *Detect* Algorithm chooses the best differentiators to identify these rare fraud events

The powerful algorithm used by *DETECT* can compute several different measures of risk. Each measure has its merits but *DETECT* allows the user to choose.

### 6.1 Accumulated Risk

The risk of fraud for a particular transaction is calculated using models derived from the data. This underpins all types of risk assessment. However, the probability alone provides a simple and intuitive risk value. The risk is accumulated over a time period so that a rapid succession of high-risk transactions will produce a higher overall risk. Conversely, an anomalous high risk will decay away if not reinforced.

### 6.2 Accumulated Expected Loss

In its simplest form, the Expected Loss is the value of the transaction multiplied by the probability of fraud. This is a much more natural measure of risk exposure. The Accumulated Expected Loss is a more complex calculation that reveals the true exposure for the account and not just the risk exposure the current transaction. There are many benefits that flow from this calculation:

- Identification of highest financial risk over time
- Effective targeting of resources
- Prioritisation of Risk

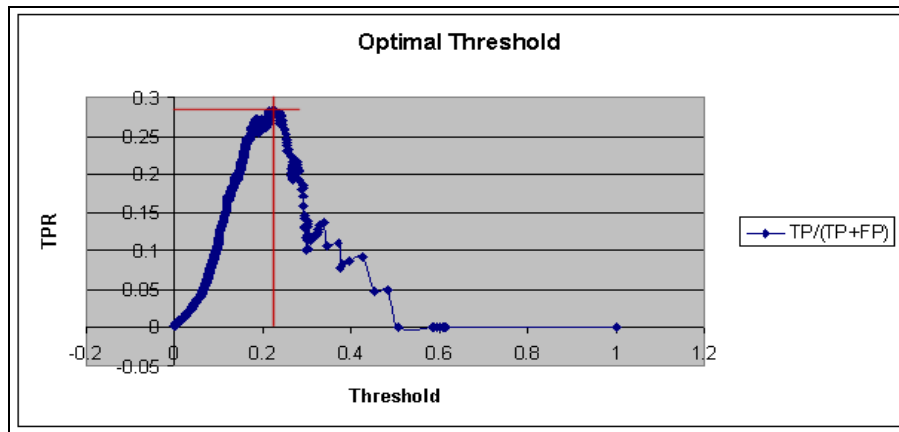
### 6.3 Likelihood Ratio

This is the ratio of the two probabilities: the probability of getting a transaction of this type assuming fraud and the probability of getting a transaction of this type assuming legality. It is a method of comparing the two statistical hypotheses; the transaction is drawn from the fraud data, and the transaction is drawn from the legal data.

## 7 Alerts

Alerts are generated when the selected risk exceeds a threshold. By adjusting the threshold the number of alerts is controlled. As the threshold is lowered, the number of alerts increases. At the extreme, if the threshold is reduced to zero then every transaction will be alerted and although a 100% detection rate could be claimed there will also be an overwhelming number of False Positives. Choosing the threshold that provides the best detection-rate with an acceptable number of alerts is often difficult.

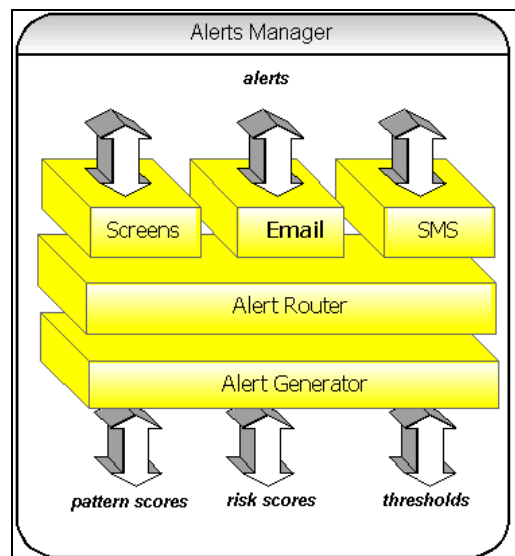
The graph below illustrates this:



There is a threshold where the percentage of alerts that are correct (the True Positive Ratio or TPR) is at a maximum. Below this threshold fraud is missed (False Negatives). Above this threshold fraud is caught, but at the cost of more false alerts (False Positives).

But at the maximum the most fraud can be caught for the least cost and effort. By tuning the alert-rate it is possible to maximise the detection of fraud while minimising the Cost of Ownership in terms of staffing levels. *DETECT* provides unique tools that allow this.

This is Alerts in general. *DETECT* actually supports several different types of alert and each three priority levels: high, medium and low. This enables thresholds for medium level alerts to be the optimal value, with independent thresholds the high level alerts to catch high-risk incidents the low-level alerts to sweep up the low-risk. By differentiating alerts levels in this way users target resources and opt to be informed by email SMS of particular alert types and levels.



has  
set to  
for  
and  
ones.  
can  
or

## 7.1 System Alerts

System alerts are generated by the Risk Engine. The computation of the risk measures is based on *DETECT*'s built in algorithms. Each measure generates alerts. Enabling immediate visibility when the expected loss on an account exceeds a threshold.

## 7.2 Pattern Alerts

The user of *DETECT* can set up patterns that experience has taught to be good detectors of fraud or to capture short term situations such as transactions from a particular merchant. Patterns allow the user to use their specialist knowledge of their particular client-base. Further, variables derived from the raw transaction data are also exposed (for instance, Rate of Spend) and can be included in patterns.

## 7.3 Customer Alerts

A separate risk threshold (the customer risk threshold) can be enabled so that customers can be sent an SMS message alerting them that their card has been used and detailing certain aspects of the transaction (amount, time, merchant, etc). The customer only needs to reply to this message if they wish to confirm that the transaction is fraudulent. A window in which a customer must respond is configurable. After this time the transaction is assumed to have been confirmed.

There are many advantages to this mechanism:

1. *Better detection.* Not all transactions will result in a message to customers, only those that exceed the customer risk threshold. This threshold can be set to be lower than the system risk threshold allowing a greater number of false-positives. These alerts do not impose an operational burden on the bank but are filtered by customer responses, allowing a higher overall detection rate.
2. *Immediate Feedback.* The immediate feedback from customers allows all subsequent transactions on the account to be blocked and causes the system to produce high-level alerts.
3. *Performance.* The information is available for immediate use by the system for calibration, which again improves system performance.

SMS alert messages need to be sent via a Short Message Service Centre (SMSC) who typically charge about €0.05 per message. So a medium-sized bank might expect to send about 1,000 messages per day at a cost of approximately €50.

To use customer-alerting the system needs access to a list that pairs mobile-phone numbers with account numbers. This list can be imported into *DETECT* or *DETECT* can access this remotely. Maintenance of such a list can be a costly burden if the bank does not have an online means of capturing this information, hence *DETECT* can provide a means whereby a customer can send an SMS message to the system quoting their account number. *DETECT* will record the mobile number against the account number but does not enable customer-alerting until an operator has checked with the customer and explicitly enabled it.

## 8 Alert Notification Management

Alerts can be controlled by thresholds, and Alert Notification Management provides control over how the system provides alerts so that it can respond according to the bank's own working practices. There is no need to adapt to the system; the system can adapt to the particular requirement.

### 8.1 Alert Notification and Routing

Alerts can be sent via SMS to selected bank staff based on level of risk or other patterns such as country, merchant category, etc.

### 8.2 Alert Escalation Policy

If an alert remains unread for a period of time, the alert can be escalated according to a configurable policy. For instance, if a high-risk alert remains unread for 30 minutes, an email/SMS message can be sent to senior members of staff.

### 8.3 Case-based alerts

Once an alert has been raised on an account it is often regarded as unnecessary to raise any further alerts while the particular case is being investigated. *DETECT* provides this option and although it will assess the risk of each subsequent transaction on this account and record it, it will not raise further alerts until the block is cleared.

### 8.4 Alert Audit Trail

When an alert is read, the fact that it has been closed, when and by whom are recorded.

## 9 Measuring Detection Performance

*DETECT* provides several means of measuring performance of both the Risk Engine and Patterns:

- 1.The percentage of fraud transactions detected for a given Alert Rate.
- 2.The percentage by value of fraud detected for a given Alert Rate.
- 3.The percentage of fraud cases detected for a given Alert Rate.
- 4.The false-positive ratio (FPR): the ratio of false to true positives. The FPR measures the ratio of incorrect alerts to correct alerts. From an operational point of view we want the FPR to be as low as possible.
- 5.ROC curves and Precision-Recall curves
- 6.Gini - the area above the diagonal of a ROC graph.

## 10 Interface

The interface to *DETECT* is very simple to use. It uses a standard web browser allowing it to be accessed from any authorised workstation and/or user (see section 12).

*Multi-lingual.* The language used by the interface is very easy to change and can support any local language, including non-Latin scripts.

*Configurable.* The look and feel can be configured to suit.

## 11 Report Generation

*DETECT* provides several screens for summarising alerts over a period. However, it is often the case that users wish to have access to tables to generate their own tailored reports.

For security and speed *DETECT* uses an embedded database that cannot be connected to from outside the application. *DETECT* therefore provides an export mechanism whereby data can be copied to an external database. The export mechanism also allows the data to be reformatted so it is in a convenient form for generating reports.

*DETECT* can also export to any JDBC compliant database such as Oracle, Sybase, DB2, MS SQL Server, etc. It can also export information to a simple CSV file for importing into spreadsheets, etc.

## 12 Security

All data held by the system is kept in an embedded database that cannot be connected to from outside of the application. All account specific information is anonymised and encrypted.

Access to the system is controlled by a 3-tier authentication system that allows the bank to control access to information.

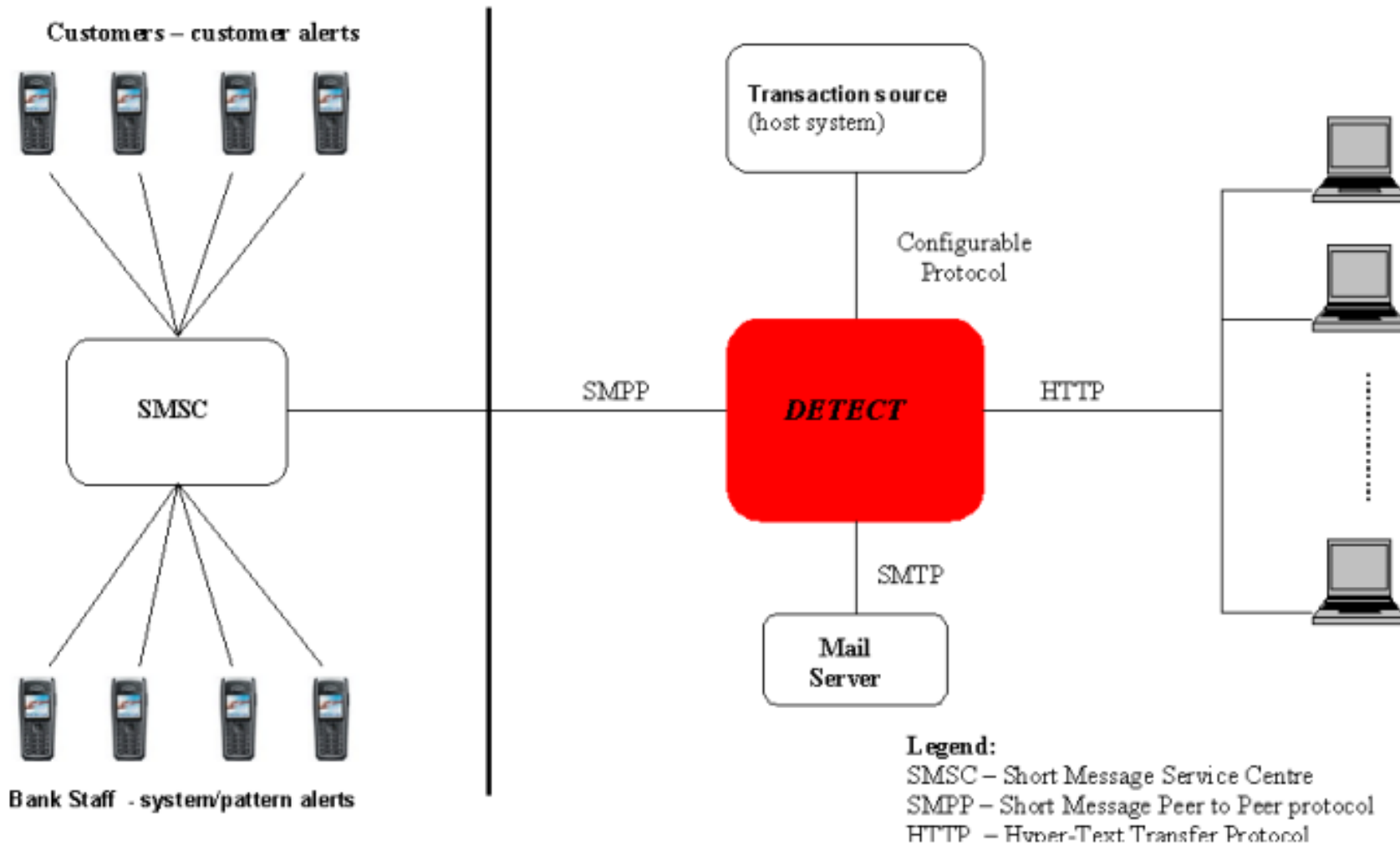
Users and workstations are subject to authentication. All user activity is logged.

The system requires few privileges to run. It can therefore be effectively isolated, as required, to conform with security policies.

## 13 Summary

- ✓ Maximum detection with minimum cost
- ✓ Minimum staff costs
- ✓ Ease of Implementation - weeks not months
- ✓ Platform Independence
- ✓ Low Cost of Ownership

### Network overview of *DETECT* working together





## Alert Screen – displaying alerts by transaction

High alerts are coloured red; medium alerts are yellow; low alerts are blue.

sts	alertid	account	alerttime	eventdate	value	threshold	risk
	634	*****534269	2005-11-16 16:52	2005-07-25 05:15	52.89	0.09	0.099
	630	*****145620	2005-11-16 16:52	2005-07-25 05:06	765.83	0.03	0.031
	627	*****388778	2005-11-16 16:52	2005-07-25 04:52	405.65	0.03	0.034
	625	*****366692	2005-11-16 16:52	2005-07-25 04:42	209.00	0.07	0.073
	624	*****366692	2005-11-16 16:52	2005-07-25 04:41	304.00	0.03	0.057
	622	*****366692	2005-11-16 16:52	2005-07-25 04:40	408.00	0.03	0.038
	620	*****862834	2005-11-16 16:52	2005-07-25 04:27	3442.69	0.03	0.034
	618	*****573448	2005-11-16 16:52	2005-07-25 04:20	15.00	0.03	0.032
	615	*****654235	2005-11-16 16:52	2005-07-25 04:08	39.57	0.03	0.042
	613	*****175612	2005-11-16 16:52	2005-07-25 04:02	564.00	0.03	0.049
	611	*****475127	2005-11-16 16:52	2005-07-25 04:01	1606.39	0.03	0.037
	610	*****866587	2005-11-16 16:52	2005-07-25 03:59	49.00	0.07	0.073
	609	*****221542	2005-11-16 16:51	2005-07-25 03:51	20.00	0.03	0.048
	608	*****221542	2005-11-16 16:51	2005-07-25 03:50	20.00	0.03	0.039
	607	*****866587	2005-11-16 16:51	2005-07-25 03:33	49.00	0.03	0.049
	604	*****397884	2005-11-16 16:51	2005-07-25 03:05	209.00	0.09	0.109

15:28:53

alert received

DETECT



## Highest Risk Alerts by Account

sts	alertid	account	alerttime	eventdate	value	threshold	risk
604		*****397884	2005-11-16 16:51	2005-07-25 03:05	209.00	0.09	0.109
634		*****534269	2005-11-16 16:52	2005-07-25 05:15	52.89	0.09	0.099
593		*****952153	2005-11-16 16:51	2005-07-25 02:40	23.38	0.09	0.095
572		*****600730	2005-11-16 16:50	2005-07-25 00:34	1068.41	0.07	0.08
610		*****866587	2005-11-16 16:52	2005-07-25 03:59	49.00	0.07	0.073
625		*****366692	2005-11-16 16:52	2005-07-25 04:42	209.00	0.07	0.073
575		*****327107	2005-11-16 16:50	2005-07-25 00:49	45.90	0.03	0.066
592		*****217551	2005-11-16 16:51	2005-07-25 02:40	514.00	0.03	0.062
613		*****175612	2005-11-16 16:52	2005-07-25 04:02	564.00	0.03	0.049
609		*****221542	2005-11-16 16:51	2005-07-25 03:51	20.00	0.03	0.048
615		*****654235	2005-11-16 16:52	2005-07-25 04:08	39.57	0.03	0.042
567		*****723098	2005-11-16 16:49	2005-07-25 00:21	675.90	0.03	0.039
570		*****845627	2005-11-16 16:50	2005-07-25 00:28	344.92	0.03	0.039
611		*****475127	2005-11-16 16:52	2005-07-25 04:01	1606.39	0.03	0.037
582		*****823835	2005-11-16 16:50	2005-07-25 01:34	135.18	0.03	0.037
589		*****440271	2005-11-16 16:51	2005-07-25 02:24	3126.42	0.03	0.035

DETECT



## Highest Expected Loss Alerts by Account

sts	alertid	account	alerttime	eventdate	value	threshold	risk
	<a href="#">572</a>	*****600730	2005-11-16 16:50	2005-07-25 00:34	1068.41	50.0	161.019
	<a href="#">589</a>	*****440271	2005-11-16 16:51	2005-07-25 02:24	3126.42	50.0	145.02
	<a href="#">620</a>	*****862834	2005-11-16 16:52	2005-07-25 04:27	3442.69	50.0	118.284
	<a href="#">635</a>	*****552009	2005-11-16 16:52	2005-07-25 05:15	7169.87	50.0	83.101
	<a href="#">565</a>	*****198041	2005-11-16 16:49	2005-07-25 00:15	2471.24	50.0	74.996
	<a href="#">611</a>	*****475127	2005-11-16 16:52	2005-07-25 04:01	1606.39	50.0	60
	<a href="#">580</a>	*****504046	2005-11-16 16:50	2005-07-25 01:29	2987.57	25.0	43.006
	<a href="#">628</a>	*****482409	2005-11-16 16:52	2005-07-25 05:02	3181.00	25.0	42.639
	<a href="#">604</a>	*****397884	2005-11-16 16:51	2005-07-25 03:05	209.00	25.0	42.613
	<a href="#">595</a>	*****232991	2005-11-16 16:51	2005-07-25 02:48	3269.42	25.0	41.051
	<a href="#">606</a>	*****969676	2005-11-16 16:51	2005-07-25 03:15	2821.15	25.0	39.614
	<a href="#">613</a>	*****175612	2005-11-16 16:52	2005-07-25 04:02	564.00	25.0	36.937
	<a href="#">619</a>	*****994306	2005-11-16 16:52	2005-07-25 04:20	2582.08	25.0	36.09
	<a href="#">564</a>	*****938109	2005-11-16 16:49	2005-07-25 00:14	2427.50	25.0	31.773
	<a href="#">592</a>	*****217551	2005-11-16 16:51	2005-07-25 02:40	514.00	25.0	27.079
	<a href="#">590</a>	*****603909	2005-11-16 16:51	2005-07-25 02:29	2340.31	25.0	26.858

DETECT



## Alert Details Screen

user recent alerts accum risk by event accum risk by channel expected cost by event expected cost by channel alert user submit >>

This form is for viewing the details of an alert and to set its status to being 'acknowledged'. You can add notes to the alert if required. The alert may have been raised for a number of reasons. By examining the contributors to the overall risk you can see which factor was most important. For more information please see the help.

acknowledge   
 alert time 2005-11-16 16:52  
 risk 0.073  
 threshold 0.07  
 value 209.00

notes

name	value
eventid	1543
value	209.0
class_tag_groupid	0
procid	1
class_date	
class_tag_code	
org_code	
class_tag_userid	0
class_threshold	
class_tag_date	

group	individual	measure	risk
RepeatMCCG	Restaurants/Bars	meanSpend	0.001
RepeatMCCG	Restaurants/Bars	SpendRate	0.017

\*\*\*\*\*366692 15:28:53 alert received

15:28:52 alert received

## Detection Pattern Variable definition screen



grpadmin system patterns variable testvariable pattern testpattern grpadmin ?

name [dropdown] new name testxn is a list

reset save >

Variables are used by Patterns. They represent a feature which the pattern is then used to test for. Variables can be simply a field in a single event or they can be derived from many events. This form is used to define variables that can then be used when setting up patterns. For more detailed information on variables see the help-chapter on Patterns.

**variable**

COUNT(CMP(value,<10),24,H)

V F W

functions windowlength units

V = COUNT ( V1, 24, hours )

functions numeric ..f(z) oper vnumeric

V1 = CMP ( value, less than, 10 )

Notes

Variable that returns the number of events that have a value less than 10 in the last 24 hours for the current channel (account).