Abstract—Building business solutions may require combining multiple existing enterprise services. In SOA paradigm this can be accomplished using composite services; a key feature in this paradigm. Composite services can be in turn recursively composed with other services into higher level solutions allowing to build new business services from existing ones. However, determining the optimal composition in terms of its short- and long-term profitability and building and expanding client base for service composers is a challenging goal especially in competitive business service environments. This paper addresses how to create profitable customer-focused composite services in those environments through optimising pricing, cost and trustworthiness of those services.

Index Terms—composition, profitability, pricing, trustworthiness, optimisation

I. INTRODUCTION

Service Oriented Architecture (SOA) and Web Services are increasingly popular, with increased attention from industry. A key concept is that services can be dynamically or statically composed to create new services. Services are described, published, discovered, and assembled, providing loosely coupled distributed interoperable business processes. Service composition is a key concept in SOA. It is the process of assembling multiple basic services into a single value-added composite service. The resulting service may be used directly by a service consumer or be recursively incorporated in further service compositions.

The composition techniques also must be able to provide the most profitable and trustworthy composite services especially in the competitive service environment. Profitability over the short and long term is achieved through a number of factors such as cost efficiency, optimal pricing and trustworthiness of provided service. Cost efficiency improves profitability directly through a more flexible margin for profit and indirectly by affording to offer the service at lower prices and hence more effectively compete as well as maintain and expand client base.

Trust is a relationship between two or more entities that indicates the expectations from an entity towards another in relation to reliance in accomplishing a certain action at a certain quality. Trustworthiness of an entity is the level of trust that the trusting entity or its agent has in that entity. Maintaining service trustworthiness helps consumer confidence in relation to risk, helps make easier choice decisions and provides a safe environment for businesses to dynamically interact and carry out transactions. Attractiveness of the composite service relates to the matching between trustworthiness and price. This varies between customer market segments.

A service provider CR acts as a composer of component services. CR wants to provide an optimal composition for enhancing continuous profitability and maintaining and expanding its customer base. This may be called optimising the Business Quality of Composition (BQoC), so that the services provide best combination of profit for CR and best experience to users. CR has access to a pool of services of which many may provide the same functionality but with varying trustworthiness and cost.

As discussed in [1] businesses may emphasise customer satisfaction programs over strategies focusing on service quality as consumers do not necessarily buy highest quality service. Other considerations such as price may enhance satisfaction without affecting consumer’s perception of service quality or trustworthiness. Attractiveness of a composite service therefore should be based on customer market segments by providing multiple levels of trustworthiness and price targeting each segment.

The paper is structured as follows: Section II discusses the function and architecture of the composition optimisation and the trustworthiness monitoring module. Optimisation of the pricing of composite services is described in Section III. Section IV explains the mechanisms for the calculation of the cost and trustworthiness of a composite service based on its composition plan. The prediction of a service’s trustworthiness based on its monitoring is examined in Section V. Section VI discusses a simulation of the optimisation of service selection. It also describes experiments using simulations of the trustworthiness monitoring in service compositions. Related work is described in Section VII and conclusions and future work are discussed in section VIII.

II. COMPOSITION OPTIMISATION ARCHITECTURE

Trustworthiness monitoring is the process of checking that service contracts are fulfilled over time, particularly if changes can occur. Monitoring is also used to detect vulnerabilities and discover attacks on a service, e.g. by making use of intrusion detection systems or dynamic testing tools available in the environment. The trustworthiness monitoring component is responsible for monitoring of composite service trustworthiness based on a set of mechanisms and metrics to ensure contract compliance.
Figure 1 illustrates the basic operation of the optimised service compositions. A Service Composer is a service provider that is responsible for constructing service compositions and offering them to consumers. A service composer is notified of important changes in the trustworthiness of a composite service as a result of one of its components. A component service that is below the satisfactory trustworthiness level can be replaced with another component service offering the same capability but with better trustworthiness. The Cost and Pricing Engine determines the cost and price of the composite service as a result of the change. It can contain varying charging schemes that can include discounts offered in some cases such as multiple component services from single or partnering providers. The consideration of costs ensures that a balance is maintained between both trustworthiness and cost efficiency of the service. The Composition Optimiser is responsible for the service selection using mechanisms that optimise compositions for profitability and customer satisfaction through cost and trustworthiness. The Process Manager is responsible for coordinating the execution of component services in a service composition.

Figure 2 shows the architecture of the trustworthiness monitoring module. The events and metrics refer to the notifications received by the module from the Event Processing and QoS Monitoring components. They include QoS metrics, user ratings and alerts that indicate violations or adherence to the service contracts, threats or changes in the environment. In addition to the direct experience through those events and metrics, the trustworthiness monitoring can exchange recommendations with other online modules in relation to service trustworthiness. Incoming events and metrics are evaluated by the module’s rules engine to generate service ratings. The rules calculate the rating for the input and add other attributes including the recency value (when event happened) and the type of event. Ratings are then stored by the module and can be used for calculating the overall trustworthiness level of each service. Context configurations allow the customisation of the trust context by adjusting the significance of types of metrics and events e.g. security and performance. Policy configurations allow setting the trustworthiness thresholds and algorithmic constants such as the rating half life. The trust engine is responsible for providing prediction of the trustworthiness levels of services both component and composite.

Metrics that define service trustworthiness in one industry may be different in another e.g. Healthcare vs. financial services. Attributes defining trustworthiness in a particular service might not have the same significance depending on the type of service and the environment. For example, security attributes of a financial service may have more priority than performance attributes where a service having high security and medium performance is preferable to one with medium security and high performance which may be different in health services. Similar analogy applies to the variations of services environments e.g. high degree of maliciousness requires more security. Therefore, different types of events can be given different weights depending on those circumstances.

III. OPTIMISATION OF PRICE IN SERVICE COMPOSITIONS

Let’s assume that \( k = 1 \) similar service compositions are offered by competitors each is characterised by trustworthiness \( T_i \) and price \( P_i \) for \( i = 2, \ldots, k \). Customers can choose between available compositions. Consumers make an overall evaluation of each composition based on a set of attributes by using utility maximisation. Attributes may differ in types and expected values between customers depending on factors such as personal preferences. Authors in [2] classify utility into deterministic utility and stochastic utility; where deterministic utility is based on measurable choice such as in our case trustworthiness and price. Stochastic utility refers to independent factors such as incomplete information or errors in customer perception. The deterministic utility is given by function \( U(P, T, \alpha, \beta) \) where \( P \) is the price of the composite service, \( T \) is its trustworthiness, \( \alpha \) is the price response function [3], and \( \beta \) is the trustworthiness response function. The price response function determines how demand changes as a function of price and it indicates price sensitivity. Similarly, the trustworthiness response function indicates how a change in trustworthiness affects the probability of purchase. \( \alpha \) and \( \beta \) are unique for each service composition and market segment.

Price response function is based on assumptions about customer behaviour. One of the models for customer demand is
called *willingness to pay* or *reservation price* which indicates the maximum price a customer is willing to pay for a service at a particular trustworthiness. A customer who decides not to pay for a composite service will choose one from a competition or may not purchase any. Commonly a (negative) scalar is used to represent a price response function as in [2] however, the function is typically not linear and tends to take a shape of downward sloping curve as illustrated in Fig. 3. The slope of the curve at a given point may be affected by other factors in addition to the price change such as:

- the distribution of the customers’ willingness to pay,
- the trustworthiness of the composite service, and
- the availability of alternative composite services from competitors.

The two most common measures of price sensitivity as in [3] are:

- *slope* of the price response function measures how demand changes due to a price change. It is equal to the change in demand divided by the change in price. The quality of this measurement reduces with larger changes in price as it assumes linearity of the price response function.
- *elasticity* of the price response function is the ratio of percentage change in demand to the percentage change in price. Elasticity changes at different prices and tends to be highest around the market price.

Trustworthiness response function $\beta$ has the opposite effect to price response function on the decision of the customer i.e. an increase in trustworthiness is expected in general to result in an increased demand ($\beta > 0$). The function is also not expected to be linear due to the existence of a range of trustworthiness where a small change can result in proportional change in the probability of purchase. This range is where trustworthiness level is generally acceptable but a change at the same price can make a customer decide whether to choose a competitive service or not to purchase the service at all. Small changes at low trustworthiness is expected to not result in proportional change in demand as customers would consider the service not trustworthy enough.

To form a new composition $CR$ may first select a market segment which determines the price and the trustworthiness. Affordable trustworthiness and cost balance varies according to each segment. $CR$ then selects services based on this objective balance.

Calculation of the cost during optimisation also considers cost change due to service combinations. For instance a component service provider might provide a discount for the use of multiple services from it or jointly with a partnering provider.

The deterministic utility for a service composition $U_i$ is calculated as a product of the utility variables:

$$U_i = (T_i)^{\beta_i} \cdot (P_i)^{\alpha_i}$$

Multinomial Logit (MNL) is commonly used in economics as a customer choice model. Using MNL in our case, the probability $\rho$ of choosing a CR's service composition instead of that from its competitors is as follows:

$$\rho_1 = \frac{e^{U_i}}{\sum_{i=1}^{k} e^{U_i}}$$

Although ideally the formula would consider all competing service providers, in practice it may be satisfactory to only consider a limited number of major competitors. We will use the variables without the subscript “1” for CR’s composition.

In order to maximise profitability we have to balance demand and price-cost difference. The cost of the composite service is the aggregated cost of its constituent services as discussed in Section IV. The problem is maximisation of the following function:

$$(P - C)\rho$$

IV. COST AND TRUSTWORTHINESS OF A SERVICE COMPOSITION

We model the trustworthiness level of a composite service in general as a function $g$ of the trustworthiness of its components:

$$T_{cs} = g ((t_1, t_2, ..., t_m))$$

*Trustworthiness* of a service refers to the level of trust that the trusting entity (service consumer) has in a service. Maintaining service trustworthiness boosts consumer confidence about risk, helps make easier choice decisions and provides a safe environment for dynamic interaction between businesses.

The calculation of the cost and trustworthiness level depends on the structure of the business process. Selection of component services statically (during design time) or dynamically uses information from which cost and trustworthiness level of the composite service can be predicted. Selected services are executed in a business process. The process is viewed externally as the composite Web service. The calculation of the trustworthiness of the composite Web service depends on the way the abstract service is constructed. Component Web services may be invoked in a business process in one or more path constructs such as the following basic and commonly supported constructs:

- **Sequence**: services are invoked one after another.
- **Parallel with Synchronisation (AND split/AND join)**: two or more services are invoked in parallel and their outcome
is synchronised. All services must be executed successfully for the next task (service) to be executed.

- **Loop or Iteration**: a service is invoked in a loop until a condition is met. We assume that the number of iterations or its average is known at the time of composition.
- **Exclusive Choice (XOR Split/XOR join)**: a service is invoked instead of others if a condition is met. We assume that the likelihood of each alternative service is to be invoked is known at the time of composition.
- **Discriminator (AND split/OR join)**: two or more services are executed in parallel but no synchronisation of the outcome of their execution.
- **Unordered Sequence**: multiple services are executed sequentially but arbitrarily.

These and several other possible patterns varyingly supported by modelling languages and products are investigated in Workflow Patterns Initiative [4].

Table I shows our functions for calculating the trustworthiness level $T_{cn}$ and cost $C_{cn}$ per service construct. For *sequence*, *parallel* and *unordered sequence* constructs the cost in those cases is simple sum of the cost of the components. The trustworthiness is calculated as a product of that of constituent services $\{t_1, \ldots, t_n\}$:

$$T_{cn} = \prod_{i=1}^{n} (t_i)$$

The cost and trustworthiness of a *loop* construct containing $n$ iterations of a service $s$ is the same as a *sequence* construct of $n$ copies of $s$ i.e. $T_{cn} = (t_i)^n$ and $C_{cn} = n \cdot c_i$.

Each service $s$ in the alternative services in the *exclusive choice* set $S$ has a probability $p$ that it will be executed and $\sum_{i=1}^{n} p_{i} = 1$. The aggregation of cost and trustworthiness in the *exclusive choice* is the sum of that of each component service multiplied by its probability.

Since a *discriminator* construct only fails if all constituent services fail, its trustworthiness is as follows:

$$T_{cn} = 1 - \prod_{i=1}^{n} (1 - t_i)$$

However, the cost in this case is the sum of the cost of all services as they are all executed.

**V. TRUSTWORTHINESS PREDICTION OF A SERVICE**

Here we propose an algorithm that is more efficient than those proposed for multiagent systems in REGRET [9] and FIRE [10] since no need for recursively running through all the ratings with each new rating received. In this algorithm, the trustworthiness level is determined using moving averages that are updated with every new rating. Older ratings reduce in value over time. The comparison with those algorithms is further discussed in the evaluation.

Trustrworthiness level $T_s$ is determined by two values $T_s = (R_s, F_s)$ where $R_s$ is the trust score for service $s$ and $0 \leq R_s \leq 1$. $F_s$ is the confidence in the score and $0 \leq F_s \leq 1$. We calculate the trust score for service $s$, $R_s$ as a dynamically weighted moving average of the rating scores. When a new rating is received the trust score is updated. First we update the total weight of all received ratings. The weighting is based on weight $w_\gamma$ of the event type that triggered the rating as configured for the service ($0 \leq w_\gamma \leq 1$) and on the recency $w_t$ of the rating. Recency weight $w_t$ decays exponentially and $0 \leq w_t \leq 1$, as follows:

$$w_t = \left(\frac{1}{2}\right)^{\frac{\Delta t}{t_i}}$$

where $\Delta t$ is the age of the rating i.e. the difference between the current time and the time when the rating took place; and $t_i$ is the customisable half life of the rating.

The accumulated weight of the trust score $w_s$ ($w_s > 0$) is updated as follows:

$$w_s = w_s \cdot w_t + w_i$$

where $w_i$ is weight of the new rating $r_i$; $w_i = w_{t_i} \cdot w_{v_i}$ but for a rating that is generated at the time of calculation i.e. $\Delta t = 0$ and $w_{t_i} = 1$, the new accumulated weight ($w_s = w_s \cdot w_t + w_{v_i}$). To facilitate the recalculation of the trustworthiness level when new ratings are received, value of $w_s$ is stored after each update.

$$R_s = \frac{(w_s - w_i) \cdot R_s + w_i \cdot r_i}{w_s}$$

We calculate the confidence value of service $s$, $F_s$ as:

$$F_s = f_\eta \cdot f_\delta$$

where $f_\eta$ is the rating quantity confidence; and $f_\delta$ is the rating quality confidence. $f_\eta$ is calculated as follows:

$$f_\eta = 1 - e^{-\gamma \cdot w_s}$$

where $\gamma$ is a constant parameter that can be used to adjust the slope of the relationship between the sum of the ratings’ weights and the quantity confidence. The higher the value of $\gamma$ the faster the full confidence (i.e. 1) is reached. It can be set

<table>
<thead>
<tr>
<th>Construct</th>
<th>Trustworthiness ($T_{cn}$) Cost ($C_{cn}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence</td>
<td>$\prod_{i=1}^{n} t_i$, $\sum_{i=1}^{n} c_i$</td>
</tr>
<tr>
<td>Parallel</td>
<td>$\prod_{i=1}^{n} t_i$, $\sum_{i=1}^{n} c_i$</td>
</tr>
<tr>
<td>Loop</td>
<td>$(t_i)^n$, $n \cdot c_i$</td>
</tr>
<tr>
<td>Exclusive Choice</td>
<td>$\sum_{i=1}^{n} p_{i} \cdot t_i$, $\sum_{i=1}^{n} p_{i} \cdot c_i$</td>
</tr>
<tr>
<td>Discriminator</td>
<td>$1 - \prod_{i=1}^{n} (1 - t_i)$, $\sum_{i=1}^{n} c_i$</td>
</tr>
<tr>
<td>Unordered Sequence</td>
<td>$\prod_{i=1}^{n} t_i$, $\sum_{i=1}^{n} c_i$</td>
</tr>
</tbody>
</table>
to any positive value but for gradual increase in confidence it should typically be set to a value between 0 and 1. The confidence increases proportionate to the number of ratings and to the degree of their recency.

The quality confidence $c$ is calculated as:

$$d_s = \frac{(w_s - w_i) \cdot d_s + w_i \cdot |R_s - r_i|}{w_s}$$  \hspace{1cm} (10)

where $d_s$ is the deviation history of the service trust ratings around the trust score. To facilitate the recalculation of the trustworthiness level when new ratings are received, value of $d_s$ is stored after each update.

$$f_\delta = 1 - d_s$$  \hspace{1cm} (11)

where $|R_s - r_i|$ is the absolute difference between the service’s trust score and individual rating score. The quality confidence indicates the deviation of the rating scores around the overall trust score and ranges between 0 (highest deviations) and 1 (lowest deviations).

For optimal selection of a component service for service compositions, we maximise the following formula:

$$(T)^\beta \cdot (C)^\alpha$$  \hspace{1cm} (12)

where $C$ is the cost of the composite service $CS$ and $T$ is a representation of the trustworthiness calculated from trust ($R_{cs}$) and confidence ($F_{cs}$) scores as in (13). $\alpha$ and $\beta$ are defined in Section III

$$T_{cs} = R_{cs} \cdot F_{cs}$$  \hspace{1cm} (13)

VI. ILLUSTRATION OF SERVICE SELECTION OPTIMISATION

To optimise service selection allowing to choose among the best component services as per equation (12), an optimisation solutions is needed. Since the trustworthiness levels and costs of component services have discrete values and because of the non-linearity of those attributes, linear programming and other solutions that require continuous variables and/or linearity are not suitable. Additionally, the number of services to select from may be large making heuristic methods a better option to provide fast and good results. Genetic algorithms (GA) are well-suited to these kind of problems. A custom GA is required to suit the characteristics of the problem of service composition.

We developed a custom GA in MATLAB that uses equation (12) as the fitness function. As illustrated in Figure 4 in which services correspond to those in Figure 1, the genome is represented by a binary matrix where each row represents an ordered set of (concrete) services belonging to a single service type (abstract service). A selected service is represented by “1” and an unselected by “0”. Therefore, each row must have only single “1” as only one service can be selected from each type as a component of the composite service. Since the number of available services may be different for each type, the number of columns in a matrix equals the number of services in the largest set of a service type $S$ where we have $m$ service types i.e. $\max_{i=1}^m(size(S_i))$. Empty elements are filled with Not-a-Number (NaN).

A set of matrices (using MATLAB cell array) are created as an initial population. The custom crossover function takes the parents as cell arrays, and returns the children that result from the crossover by reversing the order of randomly selected sections in the parents’ matrices. The custom mutation function randomly selects two elements in a row of a parent and swaps their values. Since all elements but one are set to “0”, the mutation may have an effect only if the value of one of the affected elements equals “1”. The number of generations can be fixed to a constant number or set to be proportionate to the number of service types and number of services. Figure 5 shows the improvement of the score of best composition over 50 generations for a simulation of services. Note that the problem is converted to a minimisation one. In the simulation there are 10 types of services with each type having between 5 and 10 concrete services. The cost and trustworthiness of the services are randomly assigned.

Both of the two scores we use to indicate a trustworthiness level (i.e. trust and confidence scores) are important in indicating the status of a composite and component services. Reduction of the trust score signifies receiving consistent bad ratings of the service while reduction in confidence indicates either low number of ratings received recently or significant fluctuations in the rating scores. Those fluctuations may indicate that a service is not scalable enough to meet demands during peak times. Hence a service composer might be in a situation where it has to choose between for example two component services; $s_1$ with medium trust score and low confidence score (caused...
by deviations in the ratings) and \( s_2 \) with medium trust score and medium confidence score. Despite the same trust score, the overall trustworthiness of each of the services is different as \( s_2 \) will have higher trustworthiness because of its better scalability and performance over peak times.

Figure 6 shows the monitoring of the trustworthiness of services and changes in the scores over time and with new ratings. Ratings values vary between services, the time of the rating and whether there is an (artificial) increased demand. The high demand is set to cause consistent low performance (resulting in mainly low trust score) or fluctuations in performance (resulting mainly in low confidence) in some of the services. A Gaussian random number generator is used to generate new ratings where the mean and the variance depend on the component service, its type (functionality), and the time of rating (e.g. high demand). In Figure 6 the peak demand time is between 600 and 800 which causes low and varying ratings for \( s_1 \) resulting in reduction of its confidence and to lower extent its trust score.

![Figure 6. Trustworthiness Changes in Component Services](image)

The simulation creates a composite service \( CS \) consisting of five abstract services and selects a component service for each abstract service from ten available component services based on cost and trustworthiness.

![Figure 7. Processing time in msec for FIRE and our algorithm](image)

FIRE [10] is a widely cited trust management model and algorithm for the assessment of the trustworthiness of agents in open multiagent systems. FIRE extends REGRET system developed by Sabater [9]. Unlike our approach of using a moving average, FIRE algorithm recursively runs through all the ratings with each new rating received. This results in an increasing delay in responding to requests for trustworthiness evaluation as ratings increase in quantity. Figure 7 compares the processing times of new ratings required by the algorithm described in this paper and that of FIRE. The figure clearly shows our algorithm is considerably more efficient as the number of ratings available for assessment increases.

The above experiments evaluate the trustworthiness and cost of \( CS \) based on the assumption that the component services are part of process constructs that calculate their trustworthiness as a product of the constituent services as in sequence, parallel constructs, etc. However, the trustworthiness and to lesser extent the cost of \( CS \) may be affected differently by changes in trustworthiness and cost of its components if they are part of constructs requiring different approaches for the calculation. For example, consider the effect of the decline in the trust and confidence scores of one of component services constituting a discriminator construct. This decline does not affect \( CS \) as the calculation method suggests as long as other services in the construct maintain their trustworthiness. The trustworthiness of \( CS \) here is less than that of component services because of other sequence component services that are part of \( CS \). Worthy of notice in this construct is that cost of a component plays more significant role than its trustworthiness since it is the sum of that of the component services.

Likewise, in the case of exclusive choice construct the trustworthiness of \( CS \) is only partially affected by decline of the trustworthiness of one of the construct services depending on its probability of execution. On other hand, a moderate decline in the trustworthiness of a component service executed in a loop results in a significant decline. Also note that in this case trustworthiness values are more significant than cost of a component service because of the exponential effect of changes in its trustworthiness.

VII. RELATED WORK

Service composition is similar to product bundling by companies serving their customers with heterogeneous preferences. Bundling has been studied in terms of consumer behaviour, economics and marketing. The literature on bundling focuses on reasons and contexts adequate for bundling rather than on the choice of components and price optimisation. Bitran and Ferrer in [2] address the problem of determining the composition and price of a bundle to maximise the total expected profit in a competitive environment. The study uses a scalar to represent price response function which is not realistic as demand changes differently at different price levels. It uses attractiveness as an attribute of a product in addition to price. Unlike price the study doesn’t suggest a response curve to changes in attractiveness. Chung and Rao [5] developed a model to find market segments for bundles with heterogeneous
products to estimate willingness to pay for bundles and to determine optimal prices for different market segments. Authors in [6] discuss a dynamic charging approach for service compositions including duration of usage and discounts but do not specifically address the role of cost and trust in service selection. Our paper examines strengthening composite service providers profitability through both optimisation of the composition trustworthiness and costs and through pricing optimisation.

In the past decade there has been large amount of activity in the area of computational trust and reputation, with applications in security, multi-agent systems, game theory, and spam filtering [7]–[12]. The terms trust and trust models are also used in Web service standards – e.g. WS-Trust [13] – but they are limited to the context of being able to trust the identity of the service [14]. However, establishing a Web service’s identity does not mean that the service itself is trustworthy. For instance, such an authenticated service could be temporarily unreliable or unavailable.

There is also work on trust and reputation specific to the Web services domain. In [15] the authors present an agent based trust model for Web service reputation that enables rating of individual services as well as providers. The model has shortfalls, such as the need for human intervention. A framework for reputation based service selection is proposed in [16]. Service consumers submit their ratings to a “Reputation Manager Service” which computes the service’s reputation based on those ratings. Singh [17] discusses the challenges of trustworthy service composition. He states that current approaches fail to adequately address the challenges for trust in service oriented computing. In [18] the authors introduce a framework for establishing trust in service oriented environments named RATEWeb. The framework operates by aggregating reputation ratings from consumers in a P2P fashion. It aims to support the use of trust in service selection and composition. However, unlike the work described in this paper, it does not consider the computation of trustworthiness in a composite service. It also does not provide mechanisms for responding to dynamic changes in the environment that for instance may affect a service’s trustworthiness.

**VIII. CONCLUSION**

In this paper we presented an approach to optimise short- and long-term profitability of composite services and maintaining as well as expanding customer base in a competitive service environment. A consumer may purchase the composite service that maximises their utility. The selection of component services based on cost and trustworthiness and the pricing of the composite service are made in view of compositions it will be competing with in the market. A *composition optimiser* and a *cost and pricing engine* determine the optimal cost and price for a service composition. A *trustworthiness monitoring* module monitors the adherence of the Web services to their contracts and receives metrics and alerts relating to the QoS and security events. Service ratings are generated using a rules engine and stored in the module’s rating’s store. The calculation of trustworthiness and cost depends on the construction of the composite service. The selection of composite services takes into consideration the customer value or attractiveness of the service and their expected response to changes rather than focusing merely on cost, price or trustworthiness.

Ongoing and future work includes the development of novel optimisation mechanisms for dynamic service composition and adaptation to provide robust algorithms that take into account multiple objectives including trustworthiness, cost and pricing.

**Acknowledgement:** The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant no 257930 [19].

**REFERENCES**

[8] L. Xiong, L. Liu, “PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities”, IEEE Transactions on Knowledge and Data Engineering v.16 n.7 p.843-857, July 2004
[19] Aniketos (Secure and Trustworthy Composite Services), http://www.aniketos.eu