Abstract—Trust of an enterprise distributed real-time and embedded (DRE) system indicates conformance of this system’s behavior to its published specification. As enterprise DRE systems migrate toward open-architectures that promote system design and implementation via (third-party) components and the composition of such components, it is becoming more critical to evaluate trust at multiple levels of granularity (e.g., individual components, component interactions, and component assemblies). This is because trust of an enterprise DRE system is not always equal to “summing” the trust of its individual parts.

This paper provides two contributions to research on evaluating trust in DRE systems. First, this paper presents a novel trust model, called TrDRES, which is based on subjective logic and supports evaluating trust of a DRE system at multiple levels of granularity. Secondly, it presents a case study that demonstrates an application of TrDRES on an enterprise DRE system from the domain of distributed tracking. The results of our case study highlight that trustworthiness of a composed DRE system could easily be evaluated using the proposed trust composition operators as described.

Keywords—trust, system composition, case study, enterprise DRE system

I. INTRODUCTION

Many enterprise distributed real-time and embedded (DRE) systems, such as unmanned aerial vehicles, air traffic control systems, and shipboard computer systems, are migrating towards open-architectures [18] due to the degree of flexibility and configurability open-architectures offer. Because of this migration, trust, i.e., the degree of confidence that a software service adheres to its specification, plays an important role in selecting software components (or services) that act as building blocks for the enterprise DRE system. This is because if the services are not performing as expected the overall system will fail to meet tight real-time and resource constraints in its operations.

In state-of-the-art approaches, trust metrics of evaluating enterprise DRE systems [13], [14], [20], [21] include only service users’ (i.e., external) perspective, such as ratings, and user feedback. For example, Limam et al. [14] present a model based on fuzzy cognitive maps that dynamically evaluate trust of software components/services and manage consistency of the trust of the system.

Although a Limam’s approach is feasible for evaluating trust, it has several limitations when applied to the enterprise DRE systems designed and implemented using open-architectures. First, the trust evaluation is done solely based on the history of the running system, and not based on the internal properties of the components such as the code quality, and design standards. Second, rules for composition of trust have not been indicated in deducing the trust of the overall system.

Trust, therefore, needs to be considered when designing, implementing, and composing individual software components of an enterprise DRE system (i.e., the entire software lifecycle of individual components and the composed system). To integrate trust into the entire software lifecycle of an enterprise DRE system, trust metrics have to be also considered from the service providers’ (i.e., the internal) perspective—in addition to the service users’ perspective. This is because the service users evaluations can be subjective (i.e., depends on the user requirements and the user environment) and incomplete (i.e., not all the execution scenarios can be tested), where as metrics from the internal perspective have a formal basis.

In prior work [4], we proposed a basic trust model that evaluates trust of individual software components based on the trust of individual artifacts associated with the corresponding software components. These artifacts are examined from both an internal and external perspective. The external perspective of trust is evaluated using the traditional trust metrics (e.g., user ratings and feedback); whereas, the internal perspective of trust is evaluated using following categories of trust metrics: source (i.e., who made it), elements (i.e., what it is made of), and process (i.e., how it is made). For example, in a system composed of services, trust metrics under the elements category may include the trust of individual services, and trust of the composition framework. This is because for a system to behave as expected, all the individual elements have to behave as expected.

This paper extends our prior work [4] on the trust model to support the evaluation of the trust of an enterprise DRE system at multiple levels of granularity (e.g., individual components, component interactions, and component assemblies). More specifically, the main contributions of this paper are as follows:

• It formally defines composition operators based on
  Hwang et al. [7], [9] to evaluate trust properties of a DRE system based on the trust of individual components used to realize that DRE system.
• It illustrates how the extended trust model that can be
used to evaluate trust properties of an enterprise DRE system using a case study, form the domain of distributed real-time tracking.

The results of our case study show that the evaluation of trust of a composed DRE system can be efficiently achieved using the trust operators for service interaction patterns.

**Paper organization.** The remainder of this paper is organized as follows: Section II introduces the distributed tracking system which the trust model is applied; Section III discusses our approach for composing a trusted DRE system; Section IV presents the case study in detail; Section V provides a brief survey of prevalent trust models in distributed systems; and Section VI presents concluding remarks and future research directions.

## II. Case Study: Trust of a Distributed Tracking System

This section introduces a case study from the domain of distributed object tracking. The system is called as Trusted Distributed Tracking System (TrDTS). It also motivates the need for improved models that evaluate the trust of an enterprise DRE system, which is implemented using an open-architecture.

### A. Overview of TrDTS

TrDTS is an existing distributed tracking system. It tracks objects containing patterns in an indoor setting. It is designed and implemented using jARToolKit [17] and Java 3D API [8] and is made up of approximately 10000 lines of java code. Figure 1 highlights the main software services in TrDTS.

![Fig. 1. High-level overview of the software components used to build TrDTS.](image)

As shown in this figure, TrDTS contains the following software services:

- **Sensor Services** – Sensor services provide approximations of an object’s position that is in its range of view. Examples of a sensor service include cameras, RFIDs and WiFi hubs. These services provide readings (e.g., video streams) in real-time, and often operate on limited resources. (e.g., low processing, memories, and battery power.)

- **Fusion Services** – Fusion services improve the accuracy of an object’s position by fusing position readings captured by different sensors. Example for fusion services are Kalman fusion service and Average fusion service.

TrDTS can be deployed in a health-care setting where it can be used to track equipments, people and interaction patterns between them. Due to its nature (i.e., real-time) and applicability, the TrDTS is an ideal choice to evaluate the proposed trust model.

### B. Challenges in Evaluating the Trust of TrDTS

The trust evaluation of TrDTS using prevalent approaches would be based on ratings, and third party recommendations. These metrics, due to the subjectivity, incompleteness, and lack of formality, are not adequate in evaluating trust of a systems, such as, TrDTS, which would be deployed in critical application settings. Hence, there is a need to develop a model that considers both the internal and external views and associated interacting patterns while evaluating the trust of a composed DRE system. The main challenge of such a model is formalizing the evaluation of trust by identifying trust rules associated with different composition patterns. In the following section, an enhanced trust model is proposed that address this challenge.

## III. Proposed Approach

### A. TrDRES Trust Model

Our preliminary trust model [4] evaluates the trust of a software service using different artifacts (i.e., multiple outcomes at each phase of the software life-cycle). Additionally, the model captures the dependencies between these artifacts. The artifact-based trust model is a measure of how a particular artifact meets its expected outcomes. The trust evaluation is different for each artifact and involves different stakeholders, who have certain views about the state and behavior of that artifact. The model captures these complex interactions and associated views as properties of artifacts ($p_a$). Figure 2 presents the outline of our model proposed in [4].

![Fig. 2. Basic Trust Model](image)

The internal trust view of an artifact property, $i_v(p_a)$, represents the software developers’ evaluations in realizing the artifact, (i.e., a collection of different developer experiences about the artifact). The external trust view of an artifact property, $x_v(p_a)$, represents the end-users’ measures of the
artifact, based on external evidences, (i.e., a collection of different user experiences of the artifact). This model is quantified using the subjective logic-based trust representation proposed by Jossang et al. [9], where trust is defined as a tuple: 
\[ Trust(T) = \{(B)elief, (D)isbelief, (U)ncertainty\} \]
This evaluation criteria was selected mainly, because it addresses uncertainty associated with trust. In summary, the model has the following characteristics:
- The trust of an internal view of an artifact property, \( T_{i\alpha} \), is defined as the collection of \((B, D, U)\) tuples of the internal properties.
- The trust of external view of an artifact property, \( T_{x\alpha} \), is defined as the collection of \((B, D, U)\) tuples of the external properties.
- The overall trust of the artifact, \( T_a \), of a particular phase is given by a functional aggregation of both \( T_{i\alpha} \) and \( T_{x\alpha} \), which defined as, \( T_a = f(T_{i\alpha}, T_{x\alpha}) \).
- At each iteration of the software life-cycle, the model evaluates the trust dynamically.

Hence, the trust of a software service \( S \), \( T_S \), is defined as:
\[ T_S = \forall a \in S \sum_n f(T_{i\alpha}, T_{x\alpha}) \]

The main challenges associated with this preliminary model are to quantify, aggregate and represent the trust values at each level.

B. Trust Composition Operators

Since distributed systems are made by composing services, the trustworthiness of the composite systems depends on the trustworthiness of individual participating services, and their interaction patterns. Therefore, it is necessary to build a framework that predicts the trustworthiness of the integrated system. This can be done using the information available form service specifications and identification of trust composition rules for each interaction patterns among the services in the composed system.

Before evaluating trust of a composed systems, trust of individual services needed to be specified. As indicated earlier, trust of a particular service is derived by a functional aggregation of trust attributes of service artifacts. Also, as mentioned earlier, trust attributes of a service are evaluated using subjective logic \((B,D,U)\) with respect to its functional and non-functional elements. In this paper we focus only on the non-functional (or QoS) attributes of a service.

There are many interesting research challenges in evaluating and representing trust in the service specifications. For example, \((B, D, U)\) for QoS attributes such as response time provided by a service can be deduced from both the developers confidence about delivering a particular value and the empirical testing provided by the users’ of that service. In this paper, however, it is assumed that the service specification contains its trust evaluations and these are used to derive trust of a composed system made out of these services. Trust of the composed service depends on individual service trust attributes and its composing patterns. For this purpose, the following basic interaction patterns [7] are considered.

The basic interaction patterns described in table I are:
- Sequence - Multiple services \( S_1, S_2, \ldots, S_n \) are connected sequentially.
- Parallel split/join (AND split/ AND join) - Multiple services \( S_1, S_2, \ldots, S_n \) are connected concurrently and join with synchronization.
- Exclusive choice (Exclusive split/Exclusive join) - From multiple services \( S_1, S_2, \ldots, S_n \) only one is executed at the split. (Taking \( w_{sr} \) as the probability of picking \( S_r \), where \( r=1,2,\ldots,n \))
- Discriminator (AND split/OR join) - From multiple services \( S_1, S_2 \), only one is considered at the join.
- Loop - The service is executed iteratively \( S_1 \) ‘n’ times

For example, a basic service composition could require a sequence of many services \( (S_1, S_2, \ldots, S_n) \). If response time is the quality attribute under review, then the composed system \( S \) has a response time of \( q_S = q_{S_1} + q_{S_2} + \ldots + q_{S_n} \). However, the trust of a composed service out of two sequence of individual services can not be evaluated by using simple addition operator, since it involves subjective logic and associated uncertainty. Hence, there is a need for trust composition rules for each of the interaction patterns.

Rules listed in Table I are consist of four parts, which represents the four column of the table. First column represents basic services interaction patterns, second column represents example quality attributes for each pattern, third column represents the composition operators for quality attributes [7]. It uses the operator ‘+’ for arithmetic addition, ‘∗’ for arithmetic multiplication. Fourth column represents the proposed trust composition operators for different quality attributes. Trust composition rules use two operators based on subjective logic, namely, conjunction (\( \land \)) and weighted average(\( \oplus \)). The weighted average is an extension of normal averaging into subjective logic and trust conjunction operator is inferred by conjunction operator proposed by Jossang et al. [9].

- a) Subjective Conjunction: : If trust of a binary statement ‘p’ to be true is \((b_p, d_p, u_p)\), and trust of another binary statement ‘q’ to be true is \((b_q, d_q, u_q)\), then the trust of both statements (‘p\&q’) to be true is defined as, \((b_{p\&q}, d_{p\&q}, u_{p\&q})\) where,
  \[ b_{p\&q} = b_p \land b_q \]
  \[ d_{p\&q} = d_p + d_q - d_{p\land q} \]
  \[ u_{p\&q} = b_p \land u_q + u_p \land b_q + u_p \land u_q \]

- b) Subjective Weighted Average: : If trust of a binary statement ‘p’ to be true is \((b_p, d_p, u_p)\), and trust of another binary statement ‘q’ to be true is \((b_q, d_q, u_q)\), then the statement ‘p’ is selected with a probability of \(w_p\), (the statement ‘q’ is selected with the probability of \(w_q\), \(w_q = 1 - w_p\)), then the trust of the selected statement (‘p \oplus w_p, q’) can be defined as,
  \((b_{p\oplus w_p \& q}, d_{p\oplus w_p \& q}, u_{p\oplus w_p \& q})\) where,
  \[ b_{p\oplus w_p \& q} = w_p \times b_p + w_q \times b_q \]
  \[ d_{p\oplus w_p \& q} = w_p \times d_p + w_q \times d_q \]
  \[ u_{p\oplus w_p \& q} = w_p \times u_p + w_q \times u_q \]
Both these operators could be easily extended to evaluate multiple binary statements. As these rules are for the binary statements, they could be applied for the quality related statements mentioned in the service specifications. For example, we can apply these rules to the statements about response time of service $S_1$ and service $S_2$, such as, "response time of the service $S_1$ is less than 30ms ($f_{S_1}$)", "response time of the service $S_2$ is less than 10ms ($f_{S_2}$)". Then the response time of the composed service should be a composite value of response time of $S_1$ and $S_2$, and can be evaluated based on their trustworthiness and the composition pattern as shown in Table I.

The rationals for the trust composition rules on response time for each composition pattern are described below.

**Sequence:** In a sequence of services, response time of the composed service would be the summation of response times of each service, as each service has to wait until the previous service has provided its output. Hence the calculated response time of the composed service depends on the response times of each of the individual services. if the response time of any service is slightly different than the one specified, the calculated response time would also be slightly different from the one evaluated by the summation rule. Therefore, the trust of the response time of the composed service would be the subjective conjunction of trust of response times as indicated in I.

**Parallel Split/Join:** The response time of a composite service with this pattern would mostly depend on a participant service having maximum response time. Hence the trust of the response time of the composed service is equal to the trust of the response time of the service with maximum response time.

**Exclusive choice:** In exclusive choice pattern, a probability is associated with the selection of a particular service. Therefore, the trust of the response time of the composed service depends on the response time of the chosen service. This can be expressed using the weighted average (where weight is the associated probability) of response times.

**Discriminator:** Similar to the ‘Parallel Split/Join’ pattern, the rational to the composition rule on response time for the Discriminator pattern is that the response time of the composed system depends only on the trust of response time of service with minimum response time.

**Loop:** Loop with ‘n’ iterations is equivalent to the sequence of ‘n’ services. Therefore, the trust rules for the loop composition pattern can be derived from ‘n’ application of the sequential composition rule.

### IV. Applying TrDRES to the Distributed Tracking System

The above mentioned operators are applied to TrDTS introduced in section II. Specifically, we focused on critical services in the TrDTS. These are various sensor services and the fusion services. Figure 3, 4 and 5 show a composition pattern of these services.

![Fig. 3. Composition Diagram For Distributed Tracking System](image-url)
are interested in, are response time \( q^t_4 \) and the error in the result \( q^e_4 \).

Figure 4 shows that the services \( S_1, S_2, \) and \( S_3 \) are composed using the parallel join and split pattern. Let \( S_4 \) denote the intermediate composed service. The trust of response time \( q^t_4 \), and error \( q^e_4 \) of \( S_4 \) are represented as \( (T(q^t_4)), \) and \( (T(q^e_4)) \). \( T(q^t_{S_1}) \) and \( T(q^e_{S_1}) \) are derived using the rules mentioned in the Table I, as shown below:

\[
q^t_{S_4} = \max(q^t_{S_1}, q^t_{S_2}, q^t_{S_3}) \\
T(q^t_{S_4}) = T(q^t_{S_1}) \\
T(q^e_{S_4}) = T(q^e_{S_1}) + T(q^e_{S_2}) + T(q^e_{S_3})/3
\]

For \( S_4 \) to get the above calculated value, the errors of the each of the composite services have to true. Therefore, the following subjective logic equation can be is used to compute the trust of \( q^t_4 \):

\[
T(q^t_{S_4}) = T(q^e_{S_1}) \land T(q^e_{S_2}) \land T(q^e_{S_3})
\]

The next step is to compose the sensor composed service \( S_4 \) and the fusion service \( S_5 \) to crate \( S_6 \) as shown in Figure 5.

![Figure 4](image-url)  
**Figure 4.** Sensor composed service \( (S_4) \) definition

![Figure 5](image-url)  
**Figure 5.** The complete object tracking service \( (S_6) \) definition

\[
T(S_6) = f_\ast \{T(q^t_{S_4}), T(q^e_{S_4})\}
\]

Hence, for the sake of simplicity the aggregating function \( f_\ast \) (refers to section III) is assumed to be the identify function, i.e.:

\[
T(S_6) = \{T(q^t_{S_6}), T(q^e_{S_6})\}
\]

V. Related works

This section compares our work on TrDRES to related work from the areas of Authentication/Authorization, E-Commerce Applications, QoS Composition, Peer2Peer Networks, Service Discovery and Composition and Real-Time and Embedded Systems.

**Authentication and authorization.** Josang et al. [9] propose a trust algebra for the domain of authentication-based applications (e.g., assessment of key-to-owner binding and trust-based navigation of network applications). Although the primary focus of their approaches is authentication issues, their concepts can be reused in trust-based applications in other domains. Our work is based on the subjective logic that they have proposed. However, we parameterize the use of trust metrics in evaluating the trust of DRE systems for both functional QoS angles.

**Qos Composition.** Hwang et al. [7] have proposed various interaction pattern related to QoS features in the context of composition of web services. We have used these basic patterns in this paper to compose individual features to predict the trust attributes of compose DRE systems.

**E-commerce applications.** Wang et al. [19] distinguish between two forms of trust in e-commerce applications: initial trust (i.e., the type of trust between two unfamiliar parties) and ongoing trust (i.e., confidence built towards the other party after mutual interactions).

**Peer-to-peer networks.** Kamvar et al. [11] introduce the notion of local and global trust and present a mathematical model to evaluate both these forms of trust.

**Service discovery, composition, and consumption.** In this category, trustworthiness of services are measured using different metrics. Some of these metrics are availability [1],
Reliability [5], [12], performance [15], security [3], [21], and cost [2], [16]. Trust values depend on metrics which are inferred not only from service provider’s side, but also from service consumer’s side (e.g., user’s functional and non-functional requirements, operating context, and willingness to be vulnerable [6], [10]).

Real-time and embedded systems. Alagar et al. [1] defined a formal method of ensuring trustworthiness of real-time system by preserving two kind of properties: safety (i.e., the quality of the operational behavior of the system) and security (i.e., the acceptable quality of the system before, during and after every operation).

All these above approaches (E-Commerce, P2P, Service Discovery & Real-time Embedded systems) do not consider the use of subjective logic to model the uncertainty associated with software services. Our model not only uses the (B,D,U) tuple to include the uncertainty but also considers but internal and external views associated with a service.

VI. CONCLUDING REMARKS

More and more DRE systems are composed out of available services. Hence, predicting the trust of such a composed ensemble is critical in the development of such DRE systems. In this paper, we have presented a model that uses subjective logic to predict trust attributes of a composed system. The proposed model is illustrated in the context of a distributed tracking system. Following are the lessons learnt from this study.

- The use of (B,D,U) to indicate trust of services provides a comprehensive mechanism that is suitable considering the open nature of DRE systems.
- Identification of interaction pattern and associated rules enable the qualification of the trust of an overall DRE system.

Likewise, the following is a list of future research directions:

- Investigation of trust composition rules at the function and service levels (Trust-by-Construction).
- Modeling of Trust using Tools such as GME.
- Empirical validation of the trust model by developing prototypical trust-aware DRE systems.

REFERENCES