Matching Cognitive Characteristics of Actors and Tasks

S.J. Overbeek[1], P. van Bommel[2], H.A. (Erik) Proper[2], and D.B.B. Rijsenbrij[2]

[1] e-office B.V., Duwboot 20, 3991 CD Houten, The Netherlands, EU
Sietse.Overbeek@e-office.com
[2] Institute for Computing and Information Sciences, Radboud University Nijmegen, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands, EU
{P.vanBommel,E.Proper,D.Rijsenbrij}@cs.ru.nl

Abstract. Acquisition, application and testing of knowledge by actors trying to fulfill knowledge intensive tasks is becoming increasingly important for organizations due to trends such as globalization, the emergence of virtual organizations and growing product complexity. An actor's management of basic cognitive functions, however, is at stake because of this increase in the need to acquire, apply and test knowledge during daily work. This paper specifically focusses on matchmaking between the cognitive characteristics supplied by an actor and the cognitive characteristics required to fulfill a certain knowledge intensive task. This is based on a categorization and characterization of actors and knowledge intensive tasks. A framework for a cognitive matchmaker system is introduced to compute actual match values and to be able to reason about the suitability of a specific actor to fulfill a task of a certain type.

1 Introduction

The importance of an actor's abilities to acquire, apply and test already applied knowledge increases due to e.g. growing product complexity, the move toward globalization, the emergence of virtual organizations and the increase in focus on customer orientation [1]. A knowledge intensive task is a task for which acquisition, application or testing of knowledge is necessary in order to successfully fulfill the task. When the pressure to acquire, apply and test more knowledge increases, actors struggle to manage their basic cognitive functions like e.g. the willpower to fulfill a task or maintaining awareness of the requirements to fulfill a task. These cognitive functions are also referred to as *volition* and *sentience* respectively in cognitive literature [2,3]. Difficulties to control basic cognitive functions influences practice and potentially threatens the success of task fulfillment [4]. Research in cognitive psychology has demonstrated that individual knowledge processing is negatively influenced when experiencing an overload of knowledge that needs to be processed. A burden of knowledge processing events may cause actors to underrate the rate of events [5] and to be overconfident [6].

In [7] we have discussed several types of knowledge intensive tasks, each characterized by their characteristics. These task types consist of an *acquisition* task,

R. Meersman and Z. Tari et al. (Eds.): OTM 2007, Part I, LNCS 4803, pp. 371–380, 2007.

a *synthesis* task and a *testing* task. An acquisition task is related with the elicitation of knowledge. A synthesis task is related with the actual utilization of the acquired knowledge. Lastly, a testing task is related with the identification and application of knowledge in practice inducing an improvement of the specific knowledge applied. The characteristics belonging to each task type indicate the cognitive requirements necessary for an actor to successfully fulfill an instance of a specific task type. Based on this earlier work, the research reported in this paper is specifically concerned with the *matching* of cognitive characteristics required to fulfill a certain task instance with the cognitive characteristics actually *possessed* by an actor. The ambition of this paper, however, is not to come up with a tool that will be concerned with cognitive matchmaking. Instead, the emphasis is on developing a framework which includes the aspects of such a matchmaking process and to acquire insight in how these aspects can be tackled.

2 Cognitive Actor Settings

Before elaborating on matching cognitive characteristics possessed by an actor with the cognitive characteristics required when fulfilling a task instance, a characterization of possible actor types is needed.

2.1 Actor Types

Actor types may draw from a pool of basic cognitive characteristics an actor might possess, such as sentience, volition and causability [8]. No one actor type necessarily has all of these characteristics and some have more than others. Using a series of linguistic diagnostics, Dowty [8] has shown that each of these characteristics can be isolated from the others and so should be treated as distinct. The following characteristics can thus be distinguished that can be utilized to generate cognitive settings of possible different actor types.

The *volition* characteristic is concerned with an actor's willpower to fulfill some knowledge intensive task instance. *Sentience* expresses that an actor has complete awareness of required knowledge to fulfill some task instance. The *causability* characteristic expresses that an actor has the ability to exert an influence on state changes of knowledge involved during fulfillment of some task instance. During fulfillment of certain knowledge intensive task instances an actor should be able to improve its own cognitive abilities. This is indicated by the *improvability* characteristic. The *independency* characteristic is necessary to be able to determine if an actor is able to fulfill some task instance on its own.

Having determined possible cognitive characteristics an actor may have it is now appropriate to distinguish several actor types. The combination of an actor type with the cognitive characteristics belonging to a type forms a *cognitive actor setting*. This characterization is shown in table 1. The five distinguished actor types are based on a classification of knowledge worker types [9] and on linguistic literature [2]. The set of actor types can be represented as:

$$\{\texttt{experiencer}, \texttt{collaborator}, \texttt{expert}, \texttt{integrator}, \texttt{transactor}\} \subseteq \mathcal{AT} \quad (1)$$

Table 1. Cognitive actor settings characterized

	$\mathcal{CC}$				
$\mathcal{AT}$	Volition	Sentience	Causability	Improvability	Independency
Experiencer	–	×	–	–	–
Collaborator	×	–	×	×	–
Expert	×	×	×	×	×
Integrator	×	–	×	–	–
Transactor	×	×	–	–	×

The set of cognitive characteristics can be represented as:

$$\{\texttt{volition}, \texttt{sentience}, \texttt{causability}, \texttt{improvability}, \texttt{independency}\} \subseteq \mathcal{CC} \quad (2)$$

An important remark to make here is that the actor types as well as the cognitive characteristics are not limited to five actor types and five cognitive characteristics. However, in this paper we restrict ourselves to the above mutually independent cognitive actor settings. The actor types as shown in table 1 can now be introduced.

The *experiencer* actor type has the sentience characteristic only. An experiencer is thus only aware of all the knowledge requirements to fulfill some task instance. Consider for example the following sentence: *John thoroughly reads an article about balanced scorecards before joining a meeting about balanced scorecards*. This indicates that John, as an experiencer, probably understands that reading an article about balanced scorecards is enough to successfully prepare himself for a meeting about that topic. The *collaborator* actor type possesses the volition, causability and improvability characteristics. A collaborator has the ability to exert an influence on state changes of knowledge involved during fulfillment of a task instance. During fulfillment of a knowledge intensive task instance a collaborator is also able to improve its own cognitive abilities. However, a collaborator does not have complete awareness of all required knowledge to fulfill a task instance and requires others to fulfill a task instance. Consider the following example: *John works at a hospital and requires knowledge about a patient's history. Therefore, he acquires the most recent patient log from a colleague*. This indicates that John, as a collaborator, understands that in order to acquire knowledge about a patient's history he must collaborate with another actor. After that John is able to update the patient's log with recent changes. An *expert* possesses all characteristics depicted in table 1. Suppose that John is an assistant professor working at a university and he would like to solve a difficult mathematical problem when developing a theory. He then uses his own knowledge about mathematics to solve the problem. John is also able to combine and modify his own knowledge while solving the problem and he can also learn from that. An *integrator* is able to fulfill a knowledge intensive task instance by working together and is able to initiate state changes of knowledge involved during task instance fulfillment. An integrator primarily wishes to acquire and apply knowledge of the highest possible quality. An engineer contributing to the construction of a flood barrier is an example of an integrator. Volition, sentience and independency are the characteristics belonging to the *transactor* actor type. A transactor can fulfill a task instance without collaborating with others and is not required to cause modifications in the knowledge acquired and applied during task fulfillment. A customer support employee working at a software company is an example of a transactor.

A specific instantiation of an actor type is expressed by $\mathsf{AType} : \mathcal{AC} \rightarrow \mathcal{AT}$, where $\mathcal{AC}$ is a set of *actor instances*. The example $\mathsf{AType}(a) = \texttt{experiencer}$ for instance expresses that an actor a can be classified as an experiencer. We can view the actor that is specifically fulfilling a task instance $i \in \mathcal{TI}$ as a function $\mathsf{Fulfillment} : \mathcal{AC} \rightarrow \wp(\mathcal{TI})$. Here, $\mathcal{TI}$ is a set of task instances. An actor a that fulfills a task instance i can be expressed as $\mathsf{Fulfillment}(a) = \{i\}$. A specific instantiation of a task type is expressed by $\mathsf{TType} : \mathcal{TI} \rightarrow \mathcal{TT}$, where $\mathcal{TT}$ is a set of *task types* that can be instantiated by a specific task instance. The expression $\mathsf{TType}(i) = \texttt{acquisition}$ can be used to assert that a task instance i is characterized as an acquisition task.

3 Cognitive Matchmaker System

In this section a framework for a cognitive matchmaker system is introduced that is able to compute a match between cognitive characteristics required for a specific task type and cognitive characteristics that are provided by a specific actor type. As a running example, we use the matchmaker system to match the cognitive characteristics offered by the *transactor* actor type with the required cognitive characteristics of a *synthesis* task. Figure 1 shows the architecture of the system on a conceptual level, which is translated into the formalisms throughout this section. In section 2, a function $\mathsf{AChar}_j(a) = C$ indicated the cognitive

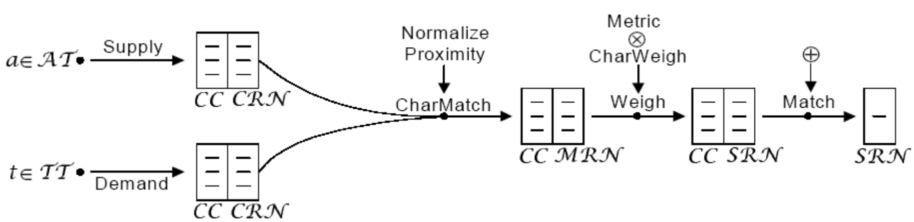

Fig. 1. Cognitive matchmaker system

characteristics that characterized an actor instance of a certain type, where j is a task type belonging to the set of task types $\mathcal{TT}$, a is an actor instance belonging to the set of actor instances $\mathcal{AC}$ and C is a set of cognitive characteristics that is a subset of or equal to $\mathcal{CC}$. Recall from section 2 that the corresponding actor type can be found by using the *actor type* function: $\mathsf{AType}(a) = j$. With this in mind, a *supply* function can be modeled that returns a value expressing to what extent an actor type offers a certain cognitive characteristic:

$$\mathsf{Supply} : \mathcal{AT} \rightarrow (\mathcal{CC} \rightarrow \mathcal{CRN}) \tag{3}$$

The expression $\mathsf{Supply}_{\mathtt{transactor}}(\mathsf{s}) = 10$ shows that an actor characterized by the transactor type offers the sentience characteristic and is at least capable to perform this characteristic at level 10. Note that for readability reasons the word 'sentience' has been abbreviated to the letter 's'. The resulting value '10' is part of a characteristic rank domain $\mathcal{CRN}$ which contains integer values within the range $[0, 10]$. The hard values as part of a domain of values can be found using the following function:

$$\mathsf{Numerical} : \wp(\mathcal{RN}) \rightarrow \mathbb{R} \tag{4}$$

Here, the set $\mathcal{RN}$ contains rank values and $\mathcal{CRN} \subseteq \mathcal{RN}$. Formally, the characteristic rank domain includes the following hard values: $\mathsf{Numerical}(CRN) = [0, 10]$. A value of 0 means that an actor is not able to offer a certain characteristic, a value of 5 means that an actor is able to offer a characteristic at an average level and a value of 10 means that an actor is able to offer a characteristic at the highest level. So, in the case of the example, the transactor is able to offer the sentience characteristic at the highest level.

Besides modeling a supply function, a demand function is needed that returns a value expressing to what extent a cognitive characteristic is *required* for a certain task type:

$$\mathsf{Demand} : \mathcal{TT} \rightarrow (\mathcal{CC} \rightarrow \mathcal{CRN}) \tag{5}$$

The expression $\mathsf{Demand}_{\mathtt{synthesis}}(\mathsf{s}) = 10$ indicates that a sentience characteristic is required at the highest level in order to fulfill a synthesis task. The supply and demand functions can now be utilized to compute the characteristic match.

3.1 Characteristic Match

In this section, a characteristic match function is defined to compare the resulting values from the supply and demand functions. This comparison provides insight in the way supply and demand of cognitive characteristics are matched with each other. In order to model a *characteristic match* function, an actor type as well as a task type are required as input, together with a cognitive characteristic:

$$\mathsf{CharMatch} : \mathcal{AT} \times \mathcal{TT} \rightarrow (\mathcal{CC} \rightarrow \mathcal{MRN}) \tag{6}$$

As can be seen in figure 1, the characteristic match function returns a value from the match rank domain, where $\mathcal{MRN} \subseteq \mathcal{RN}$. The match rank domain includes the following values: $\mathsf{Numerical}(MRN) = [0, 10]$.

To compute the actual characteristic match value, a *proximity* function is necessary to be able to define the characteristic match function. This proximity function computes the proximity of the level an actor offers a certain cognitive characteristic related to the level that is required in order to fulfill a task of a certain type. The values that are used as input for the proximity function are part of the characteristic rank domain. The resulting proximity value is then a value that is part of the match rank domain:

$$\mathsf{Proximity} : \mathcal{CRN} \times \mathcal{CRN} \rightarrow \mathcal{MRN} \tag{7}$$

A normalization function can be introduced that calculates the numerical proximity of demand and supply when a cognitive characteristic is concerned:

$$\mathsf{Normalize} : \mathbb{R} \rightarrow [0, 1] \tag{8}$$

The normalization function can be defined by using the supply and demand functions and two additional constants `min` and `max`:

$$\mathsf{Normalize}(\mathsf{Supply}_i(c) - \mathsf{Demand}_j(c)) \triangleq \frac{\mathsf{Supply}_i(c) - \mathsf{Demand}_j(c) + \mathtt{max} - \mathtt{min}}{2 \cdot (\mathtt{max} - \mathtt{min})} \tag{9}$$

Here, i is an actor type of the set $\mathcal{AT}$, j is a task type of the set $\mathcal{TT}$ and c is a cognitive characteristic of the set $\mathcal{CC}$. The values of the constants `min` and `max` can be determined by interpreting the minimum and the maximum values of a specific ranking domain. So, in the case of the running example $\mathtt{min} = 0$ and $\mathtt{max} = 10$ when the characteristic rank domain is concerned. The minimum value that can be returned by the normalization function is 0. This occurs if there is absolutely no supply (i.e. an incapable actor is concerned) but there is a maximum demand of a certain cognitive characteristic in order to fulfill a task of a certain type. This situation is depicted below:

$$\mathsf{Normalize}(0 - 10) = \frac{0 - 10 + \mathtt{max} - \mathtt{min}}{2 \cdot (\mathtt{max} - \mathtt{min})} = 0$$

In the case of an overqualified actor that is more capable to perform a cognitive characteristic than is required, the normalization function returns 1:

$$\mathsf{Normalize}(10 - 0) = \frac{10 - 0 + \mathtt{max} - \mathtt{min}}{2 \cdot (\mathtt{max} - \mathtt{min})} = 1$$

This means that the normalization function normalizes the proximity of supply and demand between 0 and 1. Using the normalization function, the proximity function can now be defined as follows:

$$\mathsf{Proximity}(\mathsf{Supply}_i(c), \mathsf{Demand}_j(c)) \triangleq \mathsf{Normalize}(\mathsf{Supply}_i(c) - \mathsf{Demand}_j(c)) \tag{10}$$

For the running example the proximity function as defined above results in:

$$\mathsf{Proximity}(10, 10) = \mathsf{Normalize}(10 - 10) = 0.5$$

Now with the introduction of a proximity function the characteristic match can be defined by computing the proximity of demand and supply in the context of a given characteristic:

$$\mathsf{CharMatch}(i, j) \triangleq \lambda_{c \in \mathcal{CC}} \cdot \mathsf{Proximity}(\mathsf{Supply}_i(c), \mathsf{Demand}_j(c)) \tag{11}$$

Recall from section 3 that an actor of the transactor type is able to perform the sentience characteristic at level 10, which equals the level to what extent a sentience characteristic should be offered for a synthesis task type. In the case of our example the characteristic match results in:

$$\begin{aligned}&\mathsf{CharMatch}(\mathtt{transactor}, \mathtt{synthesis}) = \\ &\quad \lambda_{\mathsf{s} \in \mathcal{CC}} \cdot \mathsf{Proximity}(\mathsf{Supply}_{\mathtt{transactor}}(\mathsf{s}), \mathsf{Demand}_{\mathtt{synthesis}}(\mathsf{s})) = \\ &\quad \mathsf{Proximity}(10, 10) = 0.5\end{aligned}$$

This example shows that for the transactor / synthesis task combination the eventual *proximity value* is 0.5. However, this proximity value is only related to the demand and supply of one specific cognitive characteristic. To compute a total match of the required cognitive characteristics for a task type and the characteristics offered, a *weighed suitability match* can now be introduced.

3.2 Weighed Suitability Match

The cognitive matchmaker system is completed by introducing a weighed suitability match, as is shown in the rightmost part of figure 1:

$$\mathsf{Match} : \mathcal{AT} \times \mathcal{TT} \rightarrow \mathcal{SRN} \tag{12}$$

This function returns a value from the suitability rank domain, where $\mathcal{SRN} \subseteq \mathcal{RN}$. The suitability rank domain includes the following values: $\mathsf{Numerical}(\mathcal{SRN}) = [0, 10]$. This means that an actor of a certain type can have suitability levels ranging from 0 to 10. To determine the suitability of the transactor fulfilling the synthesis task, the calculated proximity of demand and supply of a cognitive characteristic $c \in \mathcal{CC}$ can be weighed:

$$\mathsf{Weigh} : (\mathcal{CC} \rightarrow \mathcal{MRN}) \rightarrow (\mathcal{CC} \rightarrow \mathcal{SRN}) \tag{13}$$

To define the weigh function several other functions are necessary, though. As can be seen in figure 1 the weigh function uses the input from the characteristic match function and returns a value from the suitability rank domain as output. To construct the weigh function, a function is needed that has a match rank metric (i.e. the proximity value) as its input and a suitability rank metric as its output:

$$\mathsf{Metric} : \mathcal{MRN} \rightarrow \mathcal{SRN} \tag{14}$$

For instance, $\mathsf{Metric}(0.5) = 0.5$ shows that the value 0.5, which is the proximity value, equals the value 0.5 which is a suitability rank metric. A characteristic weigh function is needed to actually weigh the importance of a certain cognitive characteristic to fulfill a task of a certain type:

$$\mathsf{CharWeigh} : \mathcal{CC} \rightarrow \mathcal{SRN} \tag{15}$$

So, $\mathsf{CharWeigh}(\mathtt{s}) = 1.5$ means that a weigh factor of 1.5 is given to indicate the importance to offer the sentience cognitive characteristic (for a certain task). Finally, the $\otimes$ operator is also needed to define a definite weigh function:

$$\otimes : \mathcal{SRN} \times \mathcal{SRN} \rightarrow \mathcal{SRN} \tag{16}$$

The $\otimes$ operator is necessary to multiply the metric value with the characteristic weigh value. Multiplying the values mentioned above results in: $0.5 \otimes 1.5 = 0.75$. The weigh function can now be defined as:

$$\mathsf{Weigh}(c, \mathsf{CharMatch}(i, j)) \triangleq \lambda_{c \in \mathcal{CC}} \cdot \mathsf{Metric}(\mathsf{CharMatch}(i, j)) \otimes \mathsf{CharWeigh}(c) \tag{17}$$

Here, $c \in \mathcal{CC}, i \in \mathcal{AT}$ and $j \in \mathcal{TT}$. Continuing the running example, we would like to calculate the suitability of the transactor that is fulfilling a synthesis task. Considering the sentience characteristic only, this can be computed as follows:

$$\begin{aligned} &\mathsf{Weigh}(\mathtt{s}, \mathsf{CharMatch}(\mathtt{transactor}, \mathtt{synthesis})) = \\ &\lambda_{\mathtt{s} \in \mathcal{CC}} \cdot \mathsf{Metric}(0.5) \otimes \mathsf{CharWeigh}(\mathtt{s}) = \\ &0.5 \otimes 1.5 = 0.75 \end{aligned}$$

In order to calculate the suitability match of the transactor related to the synthesis task, it is mandatory to determine the cognitive characteristics supplied by the actor and demanded by the task. The transactor actor type supplies the *volition*, *sentience* and *independency* characteristics as is shown in table 1. The synthesis task type can be characterized by the *applicability* and *correctness* characteristics [7]. These characteristics are explained as follows. An actor should provide the applicability characteristic to be able to apply knowledge during task fulfillment and to make sure that the applied knowledge has a useful effect on successfully completing the task. An actor should provide the correctness characteristic to be able to judge the usefulness of applied knowledge in a task and to be sure that applied knowledge meets its requirements.

The set $\mathcal{CC}$ contains the following characteristics in the case of the running example: $\{\texttt{volition}, \texttt{sentience}, \texttt{independency}, \texttt{applicability}, \texttt{correctness}\} \subseteq \mathcal{CC}$. For all these characteristics a weigh value needs to be determined as in the example expression of function 17. This is necessary to compute a final *suitability match* resulting in one suitability rank value. Assume that the actual characteristic weigh values (each assigned to one cognitive characteristic as part of the set $\mathcal{CC}$) are: 2, 1.5, 0.5, 3 and 3. Note that these characteristic weigh values always summate to one and the same total value. In the case of our example the characteristic weigh values summate to 10. Thus, no matter how the weigh values are divided across the cognitive characteristics, they should always summate to a total of 10.

The results of the weighed characteristic matches have to be summated to generate a single *suitability match* value. To summate these values a $\oplus$ operator is required:

$$\oplus : \mathcal{SRN} \times \mathcal{SRN} \rightarrow \mathcal{SRN} \tag{18}$$

Now the final match function can be defined using the aforementioned functions:

$$\mathsf{Match}(i, j) \triangleq \bigoplus_{c \in \mathcal{CC}} \mathsf{Weigh}(c, \mathsf{CharMatch}(i, j)) \tag{19}$$

In the match function $i \in \mathcal{AT}, c \in \mathcal{CC}$ and $j \in \mathcal{TT}$. For the running example this means that the suitability match value of the transactor fulfilling a task instance of the synthesis type is computed as follows:

$$\mathsf{Match}(\texttt{transactor}, \texttt{synthesis}) = 1 \oplus 0.75 \oplus 0.5 \oplus 0.75 \oplus 0.75 = \mathbf{3.75}$$

As a result of the suitability match it can be concluded that the suitability of an actor characterized by the transactor type fulfilling a task instance of the synthesis type is 3.75. Remember that the lowest suitability value is 0 and the highest suitability value that can be reached is 10. The lowest value is reached if the supply of every characteristic is 0 and the demand of every characteristic is 10. The highest value is reached in the case of complete overqualification, i.e. if the supply of every characteristic is 10 and the demand of every characteristic is 0. At this point a decision can be made whether or not the specific actor is suitable enough to fulfill this task or if another actor is present who should be more suitable, i.e. has a better suitability match value. The suitability of

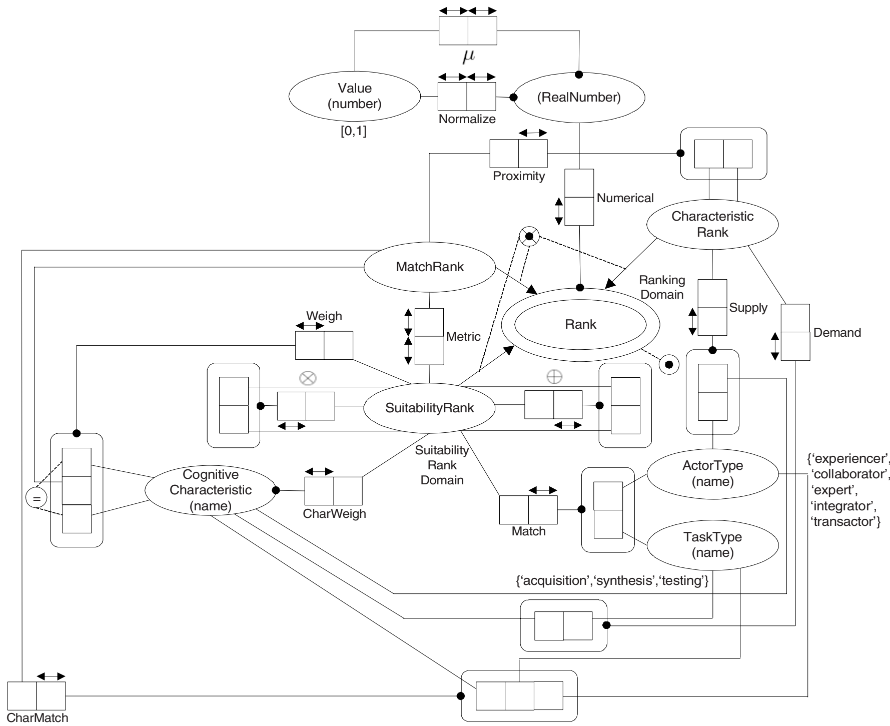

Fig. 2. Object-Role Modeling (ORM) model of the cognitive matchmaker system

an actor to fulfill a certain task is best if the resulting suitability value is 5. Underqualification as well as overqualification are both considered undesirable.

A certainty function can now be introduced to make sure how certain it is that an actor is suitable to fulfill a task:

$$\mu : \mathbb{R} \rightarrow [0, 1] \tag{20}$$

A linear certainty function can be defined as follows:

$$\mu(u) \triangleq \begin{cases} \frac{2}{\texttt{min}+\texttt{max}} \cdot u & \texttt{min} \leq u \leq \frac{\texttt{min}+\texttt{max}}{2} \\ \frac{-2}{\texttt{min}+\texttt{max}} \cdot u + 2 & \frac{\texttt{min}+\texttt{max}}{2} \leq u \leq \texttt{max} \end{cases} \tag{21}$$

For the running example, where $\texttt{min} = 0$ and $\texttt{max} = 10$, the following expression shows that the certainty that the transactor is suitable to fulfill the synthesis task is 0.75:

$$\mu(3.75) = \frac{2}{0 + 10} \cdot 3.75 = 0.75$$

This can be interpreted as being 75% sure that the transactor is suitable enough to fulfill the synthesis task. It might be a good choice to let the transactor fulfill the synthesis task, unless an available actor characterized by another type provides a better match. In order to also have a graphical representation of the discussed definitions throughout section 3, an Object-Role Modeling (ORM) model is presented in figure 2. For details on Object-Role Modeling, see e.g. [10].

4 Conclusion

This paper describes a categorization and characterization of actors that are able to fulfill knowledge intensive tasks, illustrated by cognitive characteristics indicating actor abilities for task fulfillment. Proceeding from these characteristics a running example, in which a match is determined of an actor characterized by the transactor type wishing to fulfill a synthesis task, shows how the theory can be materialized.

References

1. Staab, S., Studer, R., Schnurr, H.P., Sure, Y.: Knowledge processes and ontologies. IEEE Intelligent Systems 16(1), 26–34 (2001)
2. Kako, E.: Thematic role properties of subjects and objects. Cognition 101(1), 1–42 (2006)
3. Weir, C.R., Nebeker, J., Bret, L., Campo, R., Drews, F., LeBar, B.: A cognitive task analysis of information management strategies in a computerized provider order entry environment. Journal of the American Medical Informatics Association 14(1), 65–75 (2007)
4. Meiran, N.: Modeling cognitive control in task-switching. Psychological Research 63(3–4), 234–249 (2000)
5. Hertwig, R., Barron, G., Weber, E., Erev, I.: The role of information sampling in risky choice. In: Fiedler, K., Juslin, P. (eds.) Information Sampling and Adaptive Cognition, pp. 72–91. Cambridge University Press, New York, NY, USA (2006)
6. Koehler, D.: Explanation, imagination, and confidence in judgment. Psychological Bulletin 110(3), 499–519 (1991)
7. Overbeek, S., van Bommel, P., Proper, H., Rijsenbrij, D.: Characterizing knowledge intensive tasks indicating cognitive requirements - Scenarios in methods for specific tasks. In: Ralyt, J., Brinkkemper, S., Henderson-Sellers, B. (eds.) Proceedings of the IFIP TC8/WG8.1 Working Conference on Situational Method Engineering: Fundamentals and Experiences, Geneva, Switzerland. IFIP, vol. 244, pp. 100–114. Springer, Boston, USA (2007)
8. Dowty, D.: Thematic proto-roles and argument selection. Language 67(3), 547–619 (1991)
9. Davenport, T.: Thinking for a Living – How to get Better Performances and Results from Knowledge Workers. Harvard Business School Press, Boston, MA, USA (2005)
10. Halpin, T.: Information Modeling and Relational Databases, from Conceptual Analysis to Logical Design. Morgan Kaufmann, San Mateo, CA, USA (2001)