Towards Conceptual Foundations for Context-Aware Applications

Patrícia Dockhorn Costa[1], **João Paulo A. Almeida**[1,2],
Luís Ferreira Pires[1], **Giancarlo Guizzardi**[3,4], **Marten van Sinderen**[1]
[1]Centre for Telematics and Information Technology, University of Twente, The Netherlands
[2]Telematica Instituut, The Netherlands
[3]Computer Science Department, Federal University of Espírito Santo (UFES), Brazil
[4]Laboratory for Applied Ontology (LOA), ISTC-CNR, Trento, Italy
dockhorn@cs.utwente.nl, joaopaulo.almeida@telin.nl, pires@cs.utwente.nl, guizzardi@loa-cnr.it, sinderen@cs.utwente.nl

Abstract

In this paper we aim at defining conceptual foundations for context-aware applications. We argue that the concepts of *entity* and *context* should be separated in conceptual models for context-aware applications. Further, we propose a novel approach that characterizes context as either *intrinsic* or *relational*. The concepts we propose in this paper have been inspired by and aligned with conceptual theories from the fields of philosophy and cognitive sciences. Since we concentrate on conceptual modeling, understandability and clarity are given precedence over properties such as efficiency and tractability.

Introduction

Context-awareness has become an important and desirable feature in ubiquitous applications (Dockhorn Costa et al. 2005a). The users' demands for innovation and the interest of application providers to offer more attractive applications require context-aware application developers to anticipate unusual and surprising scenarios. As a result, new kinds of context are frequently incorporated into applications and more sophisticated context reasoning activities are used.

As applications become more complex and interconnected, there is an ultimate need for context modeling *abstractions* that are appropriate to (i) support common understanding, problem-solving, and communication among the various stakeholders involved in application development (Guizzardi 2005), and to (ii) represent context unambiguously. Context modeling abstractions should be able to adequately characterize the universe of discourse of context-aware applications. For example, a location-aware tourism application refers to concepts in a universe of discourse comprising elements such as hotels, restaurants, tourist attractions, the user's geographical location, the notion of distance between locations, etc. The quality of this application depends on the correct representation of its universe of discourse. In the tourism application, a user is expected to be informed about tourist attractions in his/her proximity. Therefore, a suitable notion of proximity should be captured in a context model. The same applies to other elements of the application's universe of discourse. The resulting context model is a conceptual model (in the sense of (Mylopoulos 1992)) of context. We argue that the definition of such context model should precede the detailed design of a context-aware application.

Current approaches towards context modeling assume that modeling languages such as OWL and UML offer the adequate conceptual foundations upon which their models can be based. However, although OWL and UML are currently being used for conceptual modeling, they have been originally designed for other purposes (OWL has been designed for computational efficiency in reasoning; UML has been initially designed to support software design and implementation). As a consequence, these languages fall short in offering suitable abstractions for constructing conceptual models, as defended extensively in (Guizzardi et al. 2002; Guizzardi 2005).

The main contribution of this paper is to provide basic conceptual foundations for context modeling, which allow designers of context-aware applications to represent relevant elements of a context-aware application's universe of discourse. These conceptual foundations should facilitate the specification of context models that are clearer and easier to understand. We do not aim at replacing current modeling languages, but instead we provide concepts and guidelines that can be used in combination with these languages to improve the quality of context models.

As a basic distinction, we propose the separation of the concepts of *entity* and *context*. Further, we propose that context should be characterized as either *intrinsic* or *relational*. We motivate these concepts in this paper, relating them to developments in foundational ontologies (Guizzardi 2005), which are in line with conceptual theories in the areas of philosophy and cognitive sciences. We also provide usage examples throughout the paper.

The remainder of this paper is structured as follows: The section on Characteristics of Context identifies relevant characteristics of context and introduces the context categorization scheme. Sections Intrinsic and Relational Context discuss the categorization scheme in more detail. Section on Formal Relations discusses the notion of formal relations, which are also useful for context modeling; and

This work is part of the Freeband AWARENESS and A-MUSE projects (http://awareness.freeband.nl and http://amuse.freedband.nl). Freeband is sponsored by the Dutch government under contract BSIK 03025.

the following sections discuss related work and present conclusions, respectively.

Characteristics of Context

Context can be defined as "the interrelated conditions in which something exists" (Merriam-Webster 2005). This definition reveals that context is only meaningful with respect to a thing (that "exists"), which we call here an *entity*.

The concept of *entity* is fundamentally different from the concept of *context:* context is what can be said about an entity, i.e., context does not exist by itself. Examples of entities are persons, computing devices and buildings. Examples of constituents of these entities' contexts are the location *of a* person, the available memory *of a* device or the temperature *of a* building. In the remainder of this paper, we use the term context to refer to constituents of the context of an entity. Together, these constituents form the entity's context.

The process of identifying relevant context consists of determining the "conditions" of entities in the application's universe of discourse (e.g., a user or its environment) that are relevant for a context-aware application or a family of such applications. The representation of these relevant conditions or circumstances is called here a *context model*. We define a context model as a conceptual model (in the sense of (Mylopoulos 1992)) of context.

We have observed that conceptual modeling of context shares a great deal of commonalities with conceptual modeling in general. We have drawn a parallel between the concepts proposed here for context and those defined elsewhere for foundational conceptual models (Guizzardi 2005; Masolo et al. 2003; Mulligan and Smith 1986). This allows us to reuse results from conceptual modeling.

We avoid in this paper a discussion on possible alternative definitions of context (as, e.g., is done in (Bucur et al. 2005)). For our work, it suffices to observe that application designers use the notion of context and context-awareness meaningfully to design successful applications to fulfill user's demands. Further, we strive to model context in terms of the state-of-affairs of an application's universe of discourse and therefore we do not to distinguish whether and how context is sensed, provided, learned, produced and/or used.

Basic Ontological Foundations

Universals and *individuals* are fundamental categories that have been considered in our modeling abstractions. Universals are predicative terms that can possibly be applied to a multitude of individuals (Guizzardi 2005). Intuitively, individuals refer to instances, while universals refer to types. We focus here on context models that capture the general aspects of context, and therefore, we only represent universals. We define a universal for entities and a universal for context, namely, `Entity` and `Context`, respectively. For example, the `Entity` type `Person` and the `Context` type `Location` are universals, while John and his actual location are individuals (instances of these universals), respectively.

Universals can be categorized as *substantial* or *moment* (Mulligan and Smith 1986). A moment is an individual that *existentially depends* on other individuals to exist, named its *bearers*. In addition, a moment should also *inhere* on its bearer(s), the way mood inheres in a person and a smile on a face. Substantials are universals that do not inhere in other universals, i.e., which are not moments. Inherence is much stronger than a one-to-one relationship. It implies existential dependence between individuals. Figure 1 summarizes these concepts.

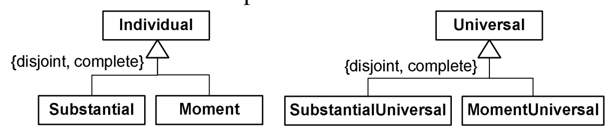

Figure 1 - Fragment of foundational concepts

Considering the fundamental categories mentioned above, we argue that `Entity` and `Context` types should be classified into *substantial universal* and *moment universal*, respectively. Since entities do not inhere in other entities, they cannot be moments, and therefore they should be classified as substantials. On the contrary, contexts always inhere in other entities, and therefore, they should be classified as moments.

Entities and Context Categories

Figure 2 depicts the relationship between the concepts of Context and Entity. We represent context models as UML class diagrams in this paper because of UML's widespread adoption. We use the stereotypes `<<SubstantialUniversal>>` and `<<MomentUniversal>>` to denote explicitly that `Entity` and `Context` are categorized as substantial and moment universals. Context types are depicted as shaded rectangles to facilitate readability.

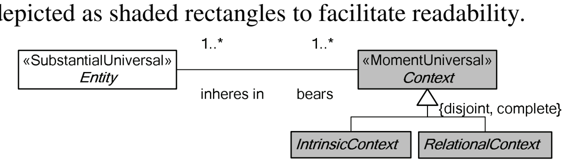

Figure 2 – Basic context modeling concepts

We distinguish two categories of context, namely intrinsic context (`IntrinsicContext`) and relational context (`RelationalContext`). Intrinsic context defines a type of context that belongs to the essential nature of a single entity and does not depend on the relationship with other entities. An example of intrinsic context is the location of a spatial entity, such as a person or a building. Relational context defines a type of context that depends on the relation between distinct entities. An example of relational context is `Containment`, which defines a containment relationship between entities, such as an entity building that contains a number of entity persons.

This categorization of context is analogous to the ontological categories of moment defined in (Guizzardi 2005), which classify moments into intrinsic or relational.

Similar to our definition, an intrinsic moment inheres in a single individual, while a relational moment inheres in a plurality of individuals.

Figure 3 depicts some examples of `Entity` types such as `SpatialEntity` and `IntangibleEntity`. Spatial entities represent tangible objects, such as a person, a device, a room or a building. Intangible entities represent intangible objects such as an application and a network. A particular type of spatial entity is the `ContainerEntity`, which is capable of physically containing other entities.

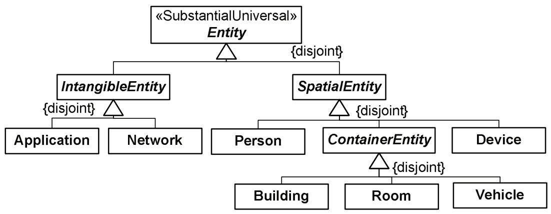

Figure 3 - Entity Hierarchy

Intrinsic Context

Figure 4 depicts examples of intrinsic context types. Geographic location (`GeoLocation`) is context that inheres in all spatial entities. Spatial entities are *bearers* of `GeoLocation`. Similarly, battery power (`BatteryPower`) inheres in a device. Analogous reasoning can be applied to other context types depicted in this figure.

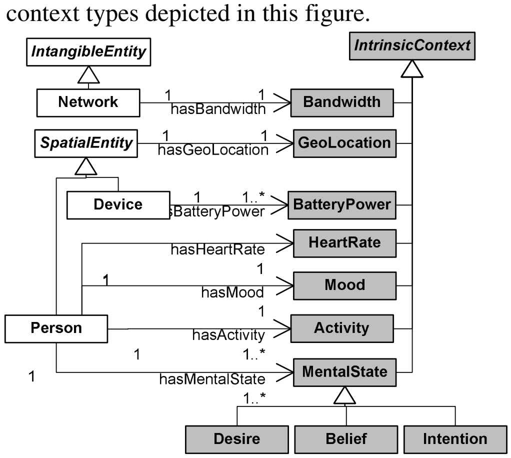

Figure 4 - Intrinsic Context Types

Intrinsic context types discussed in this paper are classified as the ontological notion of *quality universal*. Quality is an intrinsic moment that can be mapped to a value (*quale*) in a quality dimension (Guizzardi 2005). A quality dimension defines the possible set of values a quality type may be associated with. The geographical location of an entity is an example of quality, whose quality dimension is defined by all possible values in a geographical coordinate system.

The quality of an entity is an intrinsic objectified property of that entity, thus, even if two entities are co-located, they do not have the same location quality in the strong sense. Co-location depends on the granularity of associated quality dimension. For instance, take two different quality dimensions *Q*, *Q'* associated with the quality universal location such that *Q = {list of names of civil locations}*, *Q' = {precise GPS location value space}*. Under these circumstances, we can have that two entities are considered co-located in the quality space *Q* but not in *Q'*. In other words, the accuracy of our comparisons of entities' intrinsic properties depends on the precision of our quality dimensions. Quality dimensions could be represented as datatypes in our models. However, we omit datatypes here due to space limitations.

Figure 4 also presents examples of intrinsic context types of a person, such as the person's current activity, mood and mental state. These context types are quite subjective and difficult to measure. However, one could conceptualize an objective notion for these context types in a context-aware application, by enumerating the possible values (quality dimension) with which each of these types may possibly be associated. For example, we may say that the possible values of a person's mood are: "happy", "sad", "bored", "tired" and "moody"; and the possible values of a person's current activity are: "working", "dancing" or "attending a meeting".

Figure 5 shows how environmental characteristics can be modeled by using intrinsic context types associated with a `ContainerContext`. Examples of container context types are noise level and temperature of a room and humidity of a car. These context types are also *qualities*, and therefore, quality dimensions should be specified for each of them. The quality dimension of relative humidity, for example, comprises the values between 0 and 100 (percentage values).

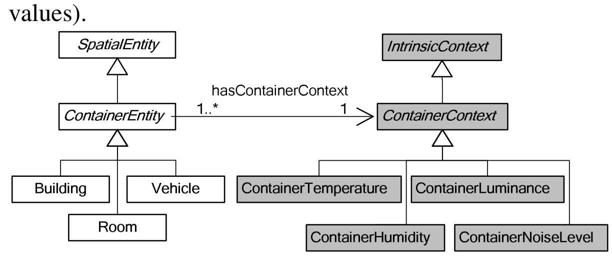

Figure 5 - Intrinsic Context Types for Persons

In some scenarios, depending on the modeling choices, context information may be classified as either intrinsic or relational. Take as an example the entity's civil location (country, street and house). It may be necessary to treat country, street and house as entities themselves, since one may be interested in properties of these entities, such as the number of persons in a house, the holidays of a country and the intensity of traffic on a street. In such scenarios, a civil location depends on the existence of a set of entities, and, therefore, is classified as relational context.

Relational Context

While intrinsic context information inheres in a single entity, relational context information inheres in a plurality of entities. Figures 6 and 7 show examples of relational context.

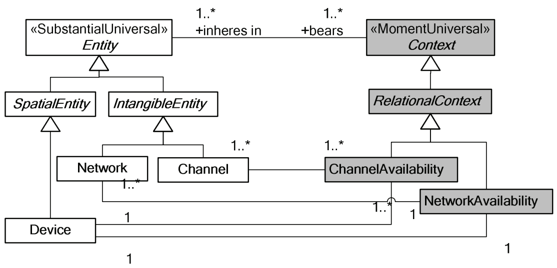

Figure 6 - Relational Context Types

Relational context may be used to relate an entity to the collection of entities that play a role in the entity's context. Examples of relational context are `DeviceAvailability`, `NetworkAvailability`, `SocialNetwork` and `ChannelAvailability`. The `DeviceAvailability` relational context relates a person to a collection of devices that are available to that person. `NetworkAvailability` relates a device to a collection of networks that are available through that device, `SocialNetwork` relates a person to the collection of persons interacting with that person by any communication channels, and `ChannelAvailability` relates a device to a collection of communication channels supported by that device (e.g., e-mail, voice and SMS).

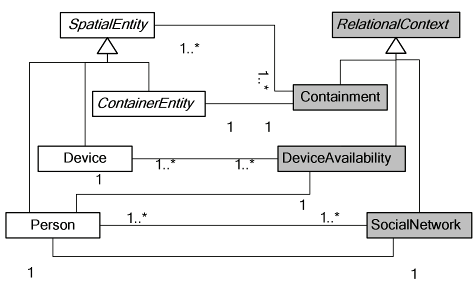

Figure 7 - Relational Context Types

Figure 7 depicts another example of relational context, the `Containment` context, which represents a direct containment relationship among spatial entities. More specifically, a `ContainerEntity` such as a building, a room or a vehicle may be associated with a containment relational context, which may in turn contain a set of spatial entities. A containment chain is created with the condition that every contained entity physically fits in its respective container entity.

Intuitively, relational context allows us to navigate the context model from an entity to the contexts of entities that are related through the relational context, still maintaining the separation of the concerns between entity and context. Consider the following example involving the entity types `Person`, `Device` and `Channel`. Let us suppose that John (of type `Person`) is related to his PDA and phone (of type `Device`) through `DeviceAvailability`. John's PDA is related to e-mail (of type `Channel`) though `ChannelAvailability`, and John's phone is related to a voice channel also through `ChannelAvailability`. Therefore, we can conclude that John is indirectly related to certain e-mail/voice channels.

We regard `RelationalContext` type as a relational moment universal in conceptual modeling. The relation that holds between bearers of a rela-tional moment is called a *material relation*. For example, the relation that holds between devices and channels through `ChannelAvailability` is a material relation.

Formal Relations

Material relations are not the only means by which one can establish the relation between entities. Conceptual modeling theories also define the notion of *formal relation.* Formal relations hold between two individuals directly, without any intervenient individual. Examples of formal relations are: *greater than*, *taller than, older than* and *subset of.* The immediate relata of such relations are qualities (Mulligan and Smith 1986), i.e., formal relations are defined in terms of their relata qualities.

Nearness is an example of formal relation useful in context modeling. The truth value of an expression such as "John is near Maria" ("nearness" being defined, for example, as within 1 km range) only depends on the values of John's and Maria's locations, which are qualities (intrinsic context). Another example of formal relation is *distance* (*Distance(x,y,z)),* which can be thought of as a logical construction from the intrinsic context a = *location(x)*, b = *location(y)*, such that z = |*valueof*(*a*)-*valueof*(*b*)| (Euclidian distance between a and b).

The distinction between material and formal relations are useful in our context models. On one hand, it is possible to derive or infer the truth value of a formal relation solely from the intrinsic context of entities related. On the other hand, for relational context, direct inference from intrinsic context is not sufficient to determine whether a material relation holds.

Relational context and formal relations may be interchanged, depending on the context model adopted. For example, one could adopt a model where the containment relational context is defined in terms of the spatial dimensions of a container and the location of a contained entity, being therefore a formal relation. A different approach is to adopt a model where the containment relational context exists on its own (for example, in a badge system). In such scenarios, there is no need to explicitly conceptualize the spatial dimensions of a container nor the location of a contained entity. In this scope, containment is categorized as relational context (and hence, is a material relation).

Related Work

Most approaches towards context modeling presented in the literature (e.g., (Chen et al. 2003; Henricksen and Indulska 2004; Wang et al. 2004; Bucur et al. 2005)) do not explore the benefits of conceptual modeling as the first phase in the design trajectory. Often these approaches consider technological issues already in the beginning of the design process, giving precedence to computational issues over human understandability. In addition, these approaches do not consider ontologically well-founded theories to support their modeling choices.

In (Henricksen and Indulska 2004), entities and intrinsic context types are represented as object types in ORM. Both relational and intrinsic context types as well as formal relations are represented as relationships between object types. Because of this modeling choice, the interpretation of context models in this technique requires one to inspect the attributes of the relationships between object types in order to distinguish intrinsic context and entities.

The work presented in (Wang et al. 2004) proposes a general context model (which they call an "upper level ontology") that supports domain-specific specializations. Although this approach uses a hierarchy, it does not distinguish context from entities, hindering the reuse of properties, since context and entities typically do not share common properties. Furthermore, the upper level ontology is not based on fundamental characterizations of context, making it difficult for distinct applications to agree upon the common model proposed.

Conclusions and Future Work

We have presented in this paper a novel context modeling approach based on conceptual modeling. As a basic distinction, we have proposed the separation of the concepts of *entity* and *context*. Further, we characterize context as either *intrinsic* or *relational*. We believe that conceptual modeling of context should precede the detailed design of context-aware applications, in a similar way as analysis should precede detailed design of an information system.

Since conceptual modeling focuses on supporting structuring and inferential facilities that are psychologically grounded (Mylopoulos 1992), the adequacy of our context modeling technique rests on how it contributes for common understanding of context among the stakeholders of a context-aware application (e.g., users and designers). Therefore, we have justified our modeling choices with results from foundational ontologies (Guizzardi 2005), which are in line with conceptual theories in philosophy and cognitive sciences.

As future work, we would like to provide support for bridging the gap between conceptual context models as proposed here and *context information models*. In the scope of context information models, we should refer to *context information* as opposed to *context*. Context information refers to the representation of (constituents of) context in an application, such that this representation can be manipulated and exchanged. Issues that become relevant for context information models relate to: (i) how context is sensed; (ii) how context information is produced, learned, inferred and used, and (iii) the validity and quality of context information.

In parallel, we are developing an approach based on conceptual graphs (Sowa 1984) to model *situations* as a complement to our structural conceptual models. Situations can be considered genuine ontological concepts to define state-of-affairs (Heller and Herre 2004). Further, we will extend our conceptual foundations to include temporal aspects of context. This would allow us to, e.g., model relevant events and changes in the state-of-affairs of a context-aware application's universe of discourse.

References

Bucur, O., Beaune, P., and Boissier, O. 2005. Defining and Modeling Context in a MAS Architecture for Decision Making. In *Proc. of the IJCAI-05 Workshop on Modeling and Retrieval of Context (MRC '05)*, Scotland.

Chen, H., Finin, T., and Joshi, A. 2003. An ontology for context-aware pervasive computing environments. *Knowledge Engineering Review, Special Issue on Ontologies for Distributed Systems.*

Dockhorn Costa, P., Ferreira Pires, L., and van Sinderen, M. 2005. Designing a Configurable Services Platform for Mobile Context-Aware Applications. *Int'l Journal of Pervasive Computing and Communications (JPCC)*, 1(1):13-25, Troubador Publishing.

Guizzardi, G., Herre, H., and Wagner, G. 2002. On the General Ontological Foundations of Conceptual Modeling. In *Proc. of the 21st Int'l Conference on Conceptual Modeling (ER-2002)*. Finland. LNCS 2503.

Guizzardi, G. 2005. *Ontological Foundations for Structural Conceptual Models*. PhD Thesis, University of Twente, The Netherlands. TI-FRS No. 15.

Heller, B., and Herre, H. 2004 *Ontological Categories in GOL*. Axiomathes 14:71-90 Kluwer Academic Publishers.

Henricksen, K., and Indulska, J. 2004. A software engineering framework for context-aware pervasive computing. In *Proc. of the 2nd IEEE Conf. on Pervasive Computing and Communications (Percom2004)*, USA.

Merriam-Webster, Inc., Merriam-Webster Online; http://m-w.com

Mylopoulos, J. 1992. Conceptual modeling and Telos, In P. Loucopoulos and R. Zicari, eds, *Conceptual modeling, databases, and CASE*, chap. 2, pages 49-68. Wiley.

Mulligan, K., and Smith, B. 1986. *A Relational theory of the Act.* Topoi (5/2), 115-30.

Masolo, C., Borgo, S., Gangemi, A., Guarino, N., and Oltramari, A. 2003. *Ontology Library*. WonderWeb D18.

Sowa, J.F. 1984. *Conceptual Structures: Information Processing in Mind and Machine*, Addison-Wesley, NY.

Hang Wang, X., Qing Zhang, D., Gu, T., and Keng Pung, H. 2004. Ontology-Based Context Modeling and Reasoning Using OWL. In *Proc. of the 2nd IEEE Conf. on Pervasive Computing and Communications Workshop (PERCOMW04)*,USA.