Cooperative Artefacts — A Framework for Embedding Knowledge in Real World Objects

Martin Strohbach, Gerd Kortuem, and Hans Gellersen

Lancaster University, Lancaster LA1 4YR, UK,
{strohbach, kortuem, hwg}@comp.lancs.ac.uk,
WWW home page: http://www.comp.lancs.ac.uk/~{strohbach, kortuem, hwg}

Abstract. In this position paper we introduce *Cooperative Artefacts*, physical objects that embed sensing, communication, computation and actuation in physical objects. In contrast to many other approaches, Cooperative Artefacts do not require any external infrastructure but cooperate by sharing knowledge. They are programmable with application rules abstracting from low level system aspects. We present an instance of our framework in connection with a scenario from the chemicals industry in which appropriate storage of chemicals is critical for safety reasons. We conclude this paper by discussing potential future research directions for Smart Object Systems.

1 Introduction

Many ubiquitous computing systems and applications rely on knowledge about activity and changes in their physical environment, which they use as context for adaptation of their behaviour. How systems acquire, maintain, and react to models of their changing environment has become one of the central research challenges in the field. Approaches to address this challenge are generally based on instrumentation of locations, user devices, and physical artefacts. Specifically, instrumentation of otherwise non-computational artefacts has an important role, as many applications are directly concerned with artefacts in the real world (e.g. tracking of valuable goods [1–3]), or otherwise concerned with activity in the real world that can be inferred from observation of artefacts (e.g. tracking of personal artefacts to infer people's activity [4]).

Typically, artefacts are instrumented to support their identification, tracking, and sensing of internal state [2, 5, 3, 6]. Complementary system intelligence such as perception, reasoning and decision-making is allocated in backend infrastructure [7, 8] or user devices [9]. This means, only those tasks that could not be provided as easily by external devices are embedded with the artefacts (e.g. unambiguous identification), whereas all other tasks are allocated to the environment which can generally be assumed to be more resourceful (in terms of energy, CPU power, memory, etc). However, this makes artefacts reliant on supporting infrastructure, and ties applications to instrumented environments.

In this paper we argue for the need of smart object systems that must not rely on any external infrastructure. We motivate this requirement with an application scenario from the chemicals industry and describe our solution in which we implement chemical containers as Cooperative Artefacts. Cooperative Artefacts model their situation on the basis of domain knowledge, observation of the world, and sharing of knowledge with other artefacts. World knowledge associated with artefacts thus becomes integral with the artefact itself and no external infrastructure is required to assess situations in a physical environment. The first part of this paper summarizes our results from [10]. In the second part we outline some of our ongoing work and outline some further research directions that may be of general interest for Smart Object Systems.

2 Handling and Storage of Chemicals

Jointly with the R&D unit of a large petrochemicals company, we are studying issues surrounding handling and storage of chemicals in the specific context of a chemicals plant in Hull, UK. Correct handling and storage of chemicals is critical to ensure protection of the environment and safety in the workplace. To guard against potential hazards, manual processes are clearly defined, and staff are trained with the aim to prevent any inappropriate handling or storage of chemicals. However the manual processes are not always foolproof, which can lead to accidents, sometimes of disastrous proportion.

In several consultation meetings we have derived a set of potentially hazardous situations that a system must be able to detect and react to. For the purposes of this presentation we will focus on a single scenario. The full set of identified scenarios is described in [10].

> *Incompatible materials*, i.e. chemicals that are reactive with each other, must be not be stored in close proximity to each other.

There are a number of important observations to be made with respect to the identified hazardous situations. First, the identified situations can occur in different environments: at the chemicals plant, in external storage (e.g. with distributors or customers), or in transit (e.g. when containers are temporarily stored together during transport). Most notably, the environments in which hazardous situations can occur are not under uniform control but involve diverse ownership (e.g. producer, distributors, consumer, logistics). This makes it unrealistic to consider a solution that would depend on instrumentation of the environment with complete and consistent coverage.

Second, the hazardous situations are defined by a combination of pre-defined domain knowledge (compatibility of materials, safety distances, etc) and real-time observations (detection of other materials, determination of proximity, etc). A generic sensor data collection approach, e.g with wireless sensor networks [11], would not be sufficient to model such situations. It is required that observations are associated with specific domain knowledge.

The described situations involve a combination of knowledge of the state of individual artefacts, and knowledge about their spatial, temporal, and semantic relationships. As a consequence, detection of situations requires reasoning across all artefacts present in a particular situation. This level reasoning is typically centralized and provided by backend infrastructure. To overcome dependency on backend services and reduce communication costs, reasoning about artefacts relationships needs to be allocated with the artefacts in a distributed and decentralized fashion.

3 Architecture

Figure 1 depicts the generic architecture for Cooperative Artefacts. Cooperative Artefacts include sensor devices to make observations of phenomena in the physical world. Sensor measurements are processed by the perception component that associates sensor data with meaning, producing observational knowledge that is meaningful in terms of the applications domain. For example, a chemical container will need to be able to recognize whether other containers are in proximity. Observations are stored and maintained in a knowledge base that reflects the current knowledge of the artefact about its world. The inference component infers further knowledge taking knowledge of nearby artefact into account and reasons about actions that should be taken based on inferred situations of artefacts in the system, e.g. using attached actuators.

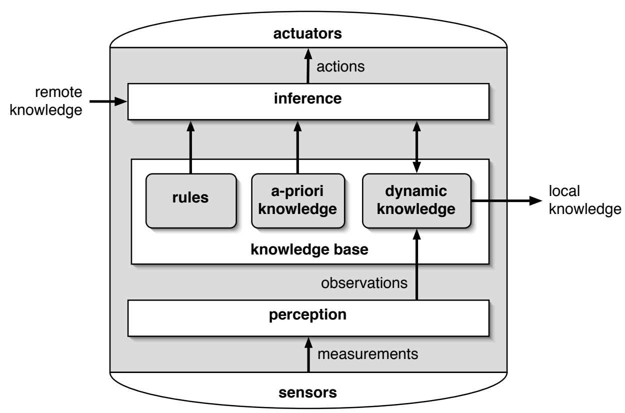

Fig. 1. Architecture of a Cooperative Artefact

It is a defining property of our approach that world knowledge associated with artefacts is stored and processed within the artefact itself. An artefact's knowledge base is structured into facts and into rules. Facts are the foundation for any decision-making and action-taking within the artefact. In addition to observations the artefact manages domain knowledge. Domain knowledge are facts that describe built-in knowledge about the application domain or the physical nature of the artefact. For example a chemical container will need to know its content and a list of incompatible materials to detect a nearby container with a reactive chemical (cf. Section 2). Regular rules allow to infer further knowledge based on facts and other rules. Special actuator rules describe the behaviour of an artefact in response to their environment.

3.1 Container Model

Table 1 depicts the knowledge base of a chemical container. We use a Prolog style notation. The literal me refers to the artefact storing the knowledge.

Table 1. Knowledge base of a chemical container.

Domain Knowledge	reactive(<chemical>,<chemical>) content(me,<chemical>)
Observational Knowledge	proximity(<container>,<container>)
Inference Rule	hazard_incompatible :- content(me,CH1), proximity(me,C), content(C,CH2), reactive(CH1,CH2).
Actuator Rule	alert_on :- hazard_incompatible.

4 Cooperative Reasoning

The ultimate goal of Cooperative Artefacts is to model all relevant aspects of a physical environment. Individual artefacts can only make limited observations of their environment, mainly due to intrinsic limitation of attached sensors and available perception algorithms. This means that knowledge is effectively distributed among artefacts. As a consequence artefacts need to cooperate to reason about changes in the environment.

Our cooperation model is based on knowledge-sharing and cross-artefact reasoning. Artefacts are individual, autonomous entities, each monitoring its own aspects of the environment. Rules in the artefact describe knowledge dependencies between facts. For instance, a chemical container assesses whether he is involved in a hazard by using the rule in Table 1 that describes that the conclusion hazard depends on the premises, i.e. the conditional part of the rule. As knowledge is distributed these dependencies may involve several other artefacts. It is therefore a key decision for the inference component to decide which facts can be obtained from which artefact.

The inference component uses a backward-chaining algorithm with choice points and meta-information about predicates to decide which artefacts should cooperate. Certain arguments can represent artefact identifiers that may provide the fact. In Table 1 the first argument of proximity and content is an artefact identifier. As proximity(me,C) and content(me,CH1) contain the literal `me`, the inference component is able to decide that no knowledge sharing with other artefacts is required. If variables are used to refer to an artefact like in `content(C,CH2)` this decision depends on the current variable binding. If the variable is bound to a value the artefact would ask the corresponding artefact to share this fact. Otherwise the local knowledge base will be used and in case of a negative result, the fact would first be searched in the local knowledge base and then in the knowledge based of nearby artefacts. While the latter case could imply a drastic increase of communication between artefacts, this situation will not occur in our scenario as variable `C` is always bound to an artefact in proximity.

Actuator rules are treated in the same way as regular inference rules with the exception that side effects can be defined for the conclusions that change the state of attached actuators.

5 Implementing a Cooperative Artefact Application

In this section we will briefly show how applications with Cooperative Artefacts can be developed. We will illustrate the process in connection with our chemicals scenario. Applications development involves 3 steps.

1. Build or instrument physical artefacts with wireless sensor nodes
2. Develop a perception module for each observation
3. Programme the artefact with rules and facts

Development for Cooperative Artefacts is supported by the arteFACT platform. The arteFACT platform provides Tools and APIs for implementing the Cooperative Artefact Architecture. Currently the arteFACT platform supports the Particle Smart-its[1] and .NET as targets.

We instrumented chemical containers as shown in Figure 2. Each container uses two individual boards. The Relate board is responsible for distance measurements using ultrasound measurements [12] and the arteFACT board implements the Cooperative Artefact architecture. As part of the measuring process Relate boards broadcast their measurements via RF.

arteFACT boards listen on the measurement broadcasts using them in a proximity module that abstracts observations with which the knowledge base is updated. Containers are in proximity if their distance falls below a pre-determined threshold. In our demonstrators this threshold is set to 20cm. This distance is hardwired in the perception algorithm.

The particles are programmed using a C API that can be used to write perception algorithms, initial rules and domain knowledge. Later changes to domain knowledge or rules can easily be made by sending simple messages to the artefacts. However, changes in the perception algorithms, requires the particles to be reprogrammed. However, if changes are anticipated at design time, they can be factored out in the knowledge base as domain knowledge, e.g. using `proximity_threshold(20)` as parameters to the proximity perception.

[1] http://particle.teco.edu

Fig. 2. Chemical container instrumented with Particle Smart-its.

Fig. 3. Testbed setup for chemical containers application

A developer of a systems of Cooperative Artefacts only has to care about the perception algorithms, rules and facts. By writing rules and facts he only describes relationships between artefacts allowing them to assess certain situations in the environment. All low level details including communication and sharing of knowledge is handled by the inference component which is part of the C API.

6 Future Research Directions

In this section we will shortly describe potential future research directions that have emerged from the work with Cooperative Artefacts:

- How is knowledge distributed among artefacts?
- How can knowledge be shared among artefacts efficiently?
- What is an appropriate declarative language for Cooperative Artefacts ?
- How can artefacts reason about temporal and spatial aspects?
- How can reasoning be limited to relevant artefacts?
- How can consistency among artefacts be achieved?
- How can activity models and memory be incorporated in physical objects?

In contrast to typical wireless sensor networks the core idea of our work is that nodes can make local decisions based on their domain knowledge, i.e. it is a crucial property that Cooperative Artefacts are autonomous entities with processing capabilities and that they always represent a physical object and not individual sensor nodes. Our framework provides the capability to distribute knowledge across artefacts and a mechanism to cooperatively reason about knowledge distributed among artefacts. However it is a decision of the designer of an artefact where knowledge is stored and maintained.

Our question is if there are general guidelines that indicate which knowledged should be assigned to which artefact. The container scenario seems to imply that artefacts should store knowledge that is related to themselves, e.g. containers should know about their content, reactive chemicals and containers in proximity. But this decision is not always easy to make, as observations can be made by artefacts that relate to other artefacts. For instance in [13] we used a table that recognized the location and weight of table top objects on its surface. Does this information only relate to the object that recognized the situation or do other involved artefacts need to know about the table observation as well? This decision may be related to the fact that artefacts are able to move independently from each other. Thus it may be necessary to replicate observations and inferred situations. We believe that this question is application specific and that further investigation into possible application scenarios will help to define guidelines on artefact modelling and knowledge distribution.

The questions about knowledge distribution is closely related to the concrete mechanisms of knowledge sharing. For instance if knowledge is replicated among several artefacts less communication is required. In our current implementation cooperative reasoning is implemented with a query/reply protocol. Whenever the knowledge base is updated, actuator rules are re-evaluated. This may result in queries sent to other artefacts. The specific rule set in our container scenario resulted in an communication efficient behaviour: only when new containers enter the proximity area queries are sent to asses the hazards. In the general case this might lead to an unnecessary communication overhead. For instance, a desk lamp could store a rule to switch on its light when a chair is occupied. This would require to periodically re-evaluate rules resulting in transmissions of queries. Communication traffic could therefore dramatically be reduced if events are supported.

Our current language does not specifically support temporal or spatial constructs. However the chemical container application underlines that spatial relationships are an important aspect of Cooperative Artefacts. We are currently extending our language to write rules that can use distance information between artefacts. For further work it will also be necessary to add temporal constructs to the language. For instance we used co-occurrence of events to infer situations in [13].

In general the expressiveness of the language must be carefully extended so that the reasoning can actually be implemented on extremely resource-constraint devices. For example, we restricted ourselves to distance information between artefacts which allows to implement our chemical hazard scenario. We have already developed a general low complexity algorithm for resource constraint embedded devices that maintains distance information between nodes.

Spatial information is also useful to scope knowledge to relevant artefacts. We are currently investigating how distance constraints in rules can be used to limit reasoning to artefacts that are within a certain range.

As individual artefacts do not have a knowledge about the rules of artefacts they cooperate with, there is also the question of consistency. Two artefacts may come to different conclusions about a situation as they may make different observations of the same physical phenomenon, they could be affected by communication failures or they could have conflicting rule sets or domain knowledge. In the current prototype this is rarely the problem, as the Relate technology delivers mostly consistent observations, retransmission of queries avoids deadlocks and communication problems and the knowledge bases do not conflict. In the general case it may however be important to exchange goals of artefacts and provide resolution mechanisms to resolve inconsistencies.

Finally Cooperative Artefacts may need to embed knowledge about their past and current activities. For example it is important to monitor how long workers are using certain tools to comply with health and safety guidelines. The tools may proactively switch themselves off if there is no valid use certificate, if the worker does not wear the required clothing or if a co-worker of supervisor is not in proximity. Most importantly this information must be captured for later retrieval to assure compliance with health and safety guidelines. Artefacts that remember their past activities can also be beneficial in production line scenarios. Here it is important that they have knowledge about their workflow process and the ability to track their progress.

7 Conclusion

In this paper we have introduced the Cooperative Artefact framework. Cooperative Artefacts are autonomous physical objects that have the capability to assess situations in their environment by cooperatively reasoning about their collective knowledge. They do not require any external infrastructure and can be programmed with a high level declarative programming language. We have presented an implementation of our framework for safe storage and handling of chemical containers. Based on our experiences with the framework and its implementation we discussed some future research directions for Smart Objects in general and Cooperative Artefacts in particular.

Acknowledgements

We would like to thank the Relate project for providing the ultrasound positioning technology. We especially thank Mike Hazas for his invaluable support in integrating the Relate hard- and software in the arteFACT platform. Cooperative Artefacts research is carried out within the EU project CoBIs and EPSRC project NEMO.

References

1. Decker, C., Beigle, M., Krohn, A., Robinson, P., Kubach, U.: eSeal – a system for enhanced electronic assertion of authenticity and integrity. In: Pervasive 2004. (2004) Vienna, Austria.

2. Lampe, M., Strassner, M.: The potential of RFID for moveable asset management. Workshop on Ubiquitous Commerce at Ubicomp 2003 (2003)
3. Siegemund, F., Flörkemeier, C.: Interaction in pervasive computing settings using bluetooth-enabled active tags and passive RFID technology together with mobile phones. In: IEEE PerCom 2003, IEEE Computer Society (2003)
4. Lamming, M., Bohm, D.: SPECs: Another approach to human context and activity sensing research. In Dey, A.K., Schmidt, A., McCarthy, J.F., eds.: Ubicomp 2003: Ubiquitous Computing. Volume 2864 of Lecture Notes in Computer Science (LNCS)., Seattle, WA, USA, Springer (2003) 192–199
5. Rekimoto, J., Ayatsuka, Y.: Designing augmented reality environments with visual tags. In: DARE 2000. (2000)
6. Beigl, M., Gellersen, H.W., Schmidt, A.: Mediacups: Experience with design and use of computer-augmented everyday artefacts. Computer Networks **35** (2001) 401–409
7. Addlesee, M., Curwen, R., Hodges, S., Newman, J., Steggles, P., Ward, A., Hopper, A.: Implementing a sentient computing system. Computer **34** (2001) 50–56
8. Kindberg, T., Barton, J., Morgan, J., Becker, G., Caswell, D., Debaty, P., Gopal, G., Frid, M., Krishnan, V., Morris, H., Schettino, J., Serra, B., Spasojevic, M.: People, places, things: Web presence for the real world. Mobile Networks and Applications **7** (2002) 365–376
9. Schmidt, A., Aidoo, K.A., Takaluoma, A., Tuomela, U., Laerhoven, K.V., de Velde, W.V.: Advanced interaction in context. In: HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing, London, UK, Springer-Verlag (1999) 89–101
10. Strohbach, M., Gellersen, H.W., Kortuem, G., Kray, C.: Cooperative artefacts: Assessing real world situations with embedded technology. In Davies, N., Mynatt, E., Siio, I., eds.: UbiComp 2004: Ubiquitous Computing. Volume LNCS 3205 of Lecture Notes in Computer Science., Heidelberg, SpringerVerlag (2004) 250–267 Nottingham, UK.
11. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: A survey. Computer Networks **38** (2002) 393–422
12. Hazas, M., Kray, C., Gellersen, H., Agbota, H., Kortuem, G., Krohn, A.: A relative positioning system for co-located mobile devices. In: MobiSys '05: Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services, New York, NY, USA, ACM Press (2005) 177–190
13. Strohbach, M., Kortuem, G., Gellersen, H.W., Kray, C.: Using cooperative artefacts as basis for activity recognition. In Davies, N., Mynatt, E., Siio, I., eds.: Proceedings of Second European Symposium of Ambient Intelligence, EUSAI 2004. Volume LNCS 3295 of Lecture Notes in Computer Science., Heidelberg, Springer Verlag (2004) 49–60