Requirements Engineering for Social Computing @ RE 2011


M. Baldoni\textsuperscript{1}  C. Baroglio\textsuperscript{1}  E. Marengo\textsuperscript{1}  V. Patti\textsuperscript{1}  A. Ricci\textsuperscript{2}

\textsuperscript{1}\textsc{Dipartimento di Informatica, Università degli Studi di Torino}

\textsuperscript{2}\textsc{DEIS, Università degli Studi di Bologna}

Trento, August 29, 2011
Overview

1. Vision and motivation
2. An interaction-centric framework
3. Discussion and conclusions
B2B, cross-business, open environment systems

- Facilitate development of new products
- Cooperatively exploit resources
- Share best practices
- Integration
- Interaction based on agreed contracts
B2B, cross-business, open environment systems

- Usually, the integration is based on the classical notion of control flow of their software, even by means of orchestration languages.
- However, this does not help software reuse and modular development.
- Instead, this reality demands abstraction and models where the involved entities are fully autonomous.
An interaction-centric approach

What is it necessary?

- A notion of “coordination”, obtained by introducing *social dimension*
- A new *interaction-centric* approach

Interaction, coordination, and communication are all central issues to the area of MAS but the current platforms are too much content-centric.
A new equation?

Some former equations

- Algorithms + Data Structures = Programs
- Algorithm = Logic + Control

The new equation

- Social computing = Dependencies + Autonomous Control
A new equation?

Some former equations

- Algorithms + Data Structures = Programs
- Algorithm = Logic + Control

The new equation

- Social computing = Dependencies + Autonomous Control
Normative characterization of interaction

- Interaction creates *social expectations* and *bindings/dependencies*, but ...

- ... a normative characterization of coordination is needed [Castelfranchi, 1997, Singh, 1999], so that ...

- ... the publicly acceptance of the regulation allows reasoning about agents’ behavior [Conte et al., 1998]
Systems made of autonomous and heterogeneous components

Current platforms do not supply:

- agents the means for *observing* or *reasoning* about such meanings of interaction
- the designers the means to explicitly *express* and *characterize* them when developing an interaction model
Agents and Artifacts as abstractions

- A&A meta-model [Weyns et al., 2007, Omicini et al., 2008] provides abstractions for environments and artifacts, that can be acted upon, observed, perceived, notified, . . .

- From a SE point of view:
  - Abstraction
  - Modularity and encapsulation
  - Extensibility and adaptation
  - Reusability
Environments, artifacts can be perceived, acted upon, observed, ... All interactions among agents will be *indirect*, like in the real world. As Keil and Goldin observed, *indirect communication fosters the collaboration and the coordination inside open systems*. Environments, artifacts can be *general, programmable channels* of communication.
Agents and Artifacts as abstractions

Why do we use environments and artifacts?

- To reify regulations aimed at coordination
- Agents can examine them
- Agents can use them
- Agents can construct them
Agents and Artifacts as abstractions

Introducing a normative characterization

- *Commitment-based* approach
  [Castelfranchi, 1997, Singh, 1999]

- A semantics of interaction for *design time verification*

- Artifacts and environments for *runtime verification*

- Social computing = Dependencies + Autonomous Control
Agents and Artifacts as abstractions

Introducing a normative characterization

- Commitment-based approach
  [Castelfranchi, 1997, Singh, 1999]
- A semantics of interaction for *design time verification*
- Artifacts and environments for *runtime verification*
- Social computing = Dependencies + Autonomous Control
The schema of our proposal

- The **specification level** allows the designer to shape the interactions that will characterize the system by supplying adequate high-level abstractions.
- The **programming abstraction level** realizes at a programming language level the abstractions defined above.
- Our starting point for the **infrastructure level** is the CArtAgO framework [Ricci et al., 2009].
We propose to rely upon *commitment-based* approach and, in particular, *commitment-based protocols* [Singh, 1999, Singh, 2000, Yolum and Singh, 2001]

Standardization and regulation of interaction is a decisive factor in distributed and open systems, made of heterogeneous and changing parties

We rely on the proposal in [Baldoni et al., 2011, Marengo et al., 2011], that allows the representation of legal patterns of interaction by enriching commitment protocols with temporal regulations.
In particular, [Baldoni et al., 2011] proposes a decoupled approach that separates a *constitutive* and a *regulative specification*. The constitutive specification defines the meaning of actions based on their effects on the social state. The regulative specification reinforces the regulative nature of commitment by adding a set of behavioral rules, by means of *temporal constraints* among commitments. Defining the legal evolution of the social state, independently from the executed actions. Advantages: easier re-use of actions, easier customization, greater compositionality.
Incorporating interaction protocols based on commitments, patterns of interaction, forms of direct and indirect communication and coordination between agents (such as stigmergic coordination) inside the programmable environments envisaged by the A&A meta-model [Weyns et al., 2007, Omicini et al., 2008]

The act of using an artifact can be interpreted as a declaration of acceptance of the coordination rules.

This will generate social expectations about the agent’s behavior and this agrees with the characterization of norms in [Conte et al., 1998]
Incorporating interaction protocols based on commitments, patterns of interaction, forms of direct and indirect communication and coordination between agents (such as stigmergic coordination) inside the programmable environments envisaged by the A&A meta-model [Weyns et al., 2007, Omicini et al., 2008]

- The act of using an artifact can be interpreted as a *declaration of acceptance* of the coordination rules
- This will generate social expectations about the agent’s behavior and this agrees with the characterization of norms in [Conte et al., 1998]
Programming Abstractions Level

- Commitment-based protocols encoded into artifacts
- Programmable communication media, having a normative characterization

Software Engineering point of view

This conjugation helps providing protocol specifications good and important features: abstraction, modularity and encapsulation, extensibility and adaptation, reusability
Infrastructure Level

- Environments used as a computational support for the agents’ activities [Omicini et al., 2004]
- Artifacts used for realizing stigmergic coordination mechanisms [Ricci et al., 2007], organizational artifacts [Hubner et al., 2009, Piunti et al., 2009]
- Our starting point is the CArtAgO framework [Ricci et al., 2009]
  - a proper computational model, and
  - a programming model for the design and the development of the environments on the base of the A&A meta-model
Conclusion

- Interaction-centric
- Social meaning of interaction
- Programmable Communication channel with monitoring functionalities
- A normative value thanks to commitment-based approach
- Explicit acceptance of the regulamentations
- Flexibility and openness typical of MAS
- Modularity and compositionality typical of design and development methodologies


