Agreement Computing: Opportunities and Challenges

Juan Carlos NIEVES
Department of Computing Science
Umeå University

What is an agreement?

A dictionary definition of an agreement is:

: an arrangement, contract, etc., by which people agree about what is to be done
We live in a society of agreements

There are agreements between people, countries, etc.

We usually express agreements in terms of contracts, standards, etc.
Implications of an agreement

Basically, we assume **OBLIGATIONS** **PERMISSIONS** between implicated parties.

**Some agreements in our daily life:**

- Agreements with our mobile provider.
- Agreements with FACEBOOK, email providers.
- *Cloud computing providers,* e.g., Dropbox.
- *Accepting Cookies when we are surfing!!*
Do you really agree?

A classical message about cookies:

**GOOGLE**: Cookies help us deliver our services. By using our services, you agree to our use cookies.

Do you agree that google tracks your:

- searches,
- clicks,
- personal data,
- send personalized advertisement,
- etc.?
How do we know if a service provider is complying a committed agreement?

• Agreements with our mobile provider.
• Agreements with FACEBOOK, email providers.
• Agreements with cloud computing providers, e.g., Dropbox.
Software components willing to participate in open systems will therefore require to include extra capabilities to explicitly represent and generate agreements, on top of the simpler capacity to interoperate, once the agreements are set.

Agreements should become the basic run-time structures that determine whether a certain interaction is correct, in a similar way as type-checking currently determines if the values in a call to a procedure are correct.
Agreement Computing

**Agreement-checking** is a run-time analysis of whether a particular interaction between two entities satisfies an agreement.

This view requires that **the interaction between two components starts** by the generation (or perhaps selection) of the **interoperation agreement** and then a subsequent phase in which the actual interoperation of the parties involved.

**These capacities** will make software components **“interaction-aware”** and can produce a significant step forward in software design and run-time verification **if a number of scientific challenges are solved.**
Agreement Computing
Challenges

Semantic interpretation of agreements:

Although standards are in place for ontology representation (e.g. OWL) there is currently no scalable solution to establish agreements between software entities on the alignment of their semantics.
Challenges

Representation of Norms

**Agreements** are explicit and declarative, and thus they open the door to use **model checking and logic based approaches**, like BDI. These techniques may make open entities **norm-aware** and endow them with the capacity to build **cognitive models of their environment**.
Challenges

Organisations (e.g., electronic institutions):

How an organisation that regulates the co-operation and set the rules of the interaction is generated on-the-fly is an unsolved question.
Challenges

The way agents and services interact depends on two types of dynamics.

1. First, **open systems evolve**, new entities appear and disappear, and thus **new agreements have to be set**.

2. Second, the rules of the game that regulate the **interaction between two entities might change** due to **the establishment of new agreements** between them and due to agreements with third parties.

**These dynamics are true challenges** as many traditional research solutions are based on a static world view (e.g. game theory, classic planning).
Conclusions

Open distributed systems are going to be the norm in the software development industry of the future and the interoperation of the software entities will need to rely on a declarative concept of agreement that is autonomously signed and executed by the entities themselves.

The generation of agreements between entities will need to integrate semantic, normative, organisation, negotiation and trust techniques.
References