Building a virtual marketplace for software development tasks

Boris Kötting & Frank Maurer
University of Kaiserlautern & University of Calgary
koetting@informatik.uni-kl.de & maurer@cpsc.ucalgary.ca

Abstract

This short paper presents initial ideas on tool support for subcontracting software development tasks over the Internet. Using a scenario, we discuss a contract-net based protocol for negotiating software development tasks. We show how this negotiation process fits into the context of our software process support environment MILOS. We briefly illustrate the design of the virtual marketplace.

Introduction

This short paper presents our approach on building a virtual marketplace for software development tasks. We use our process enactment support system (MILOS) and enhance it by offering tasks of a project to other companies in a virtual cooperation or to other parts of a globally distributed company. The term virtual cooperation was widely used in the past decade, i.e. [DM92], [BB93]. The aim of our enhancement is to support task distribution over the Internet, whether in one globally distributed company or in a virtual enterprise consisting of several companies.

We will reach this goal by designing a virtual marketplace for software development tasks that allows offering tasks to (semi-automatic) bidding agents. These agents are able to search for potential development tasks, to make a bid and to (semi-automatically) support the contract negotiation process. We are implementing this approach using the JavaSpaces™ technology.

What is MILOS?

The focus of the MILOS approach is on providing flexibility in software process support by interleaving project planning with project enactment and on supporting changes by automatic change notifications.

The major components of the MILOS are

- Resource pool: The resource pool component manages roles, agents and agent properties. It allows representing the organizational structure of a company as well as hierarchical skill sets. Agents can be found by querying for their skills.

Skills can also be used in the definition of processes to specify requirements to perform the task.

- Experience base: The experience base component can be used as a construction kit for new project plans and to save useful project plan (fragments) for future projects.
- Project plan: The project plan management component allows customizing of existing process models from the experience base as well as starting a new project from scratch. It is also possible to import MS-Project-Plans and execute them with the workflow engine. Beside adding/removing tasks, planning also includes scheduling planned start and end times of processes and assigning agents to processes.
- Workflow engine: The workflow management component is responsible for enacting the project plan and managing products. It generates to-do lists for agents and maintains the current state of the project. The workflow engine is able to react dynamically to project plan changes during execution, without interrupting the execution.

Components are linked by an event propagation mechanism that sends notifications about changes to all observers, namely other components and users.

Modeling, planning and execution can be distributed to different companies and areas by running the clients on an arbitrary machine connected to the Internet. For a more detailed description see [MD 98] [Bendeck et al. 98].

A possible scenario

When a project is planned, there are lots of risk factors and uncertainties. A planner needs to estimate many environment conditions and often cannot be sure what resources (time, human resources and equipment) he will be able to use. Assume a planner who is responsible for a software application. He creates tasks and calculates time for application logic design, implementation, integration and testing. While there are not many good jobless computer scientists and the budget is limited, he fails in acquiring staff for the application persistency aspects. This is one point where our enhancement supports the planner: when there are no employees on the local job market, there is a virtual marketplace where you can offer your development tasks.

The planner can invoke an *offering-agent* GUI where she can insert additional information about the task for potential contract partners and check/change the information provided to the bidding agent including the skills required for performing the task according to a skill ontology defined in the resource pool. Furthermore, she can set the type of offering and a recipient list (the use of wildcards is possible).

Bidding agents from other parts of the (virtual) company have access to the virtual marketplace and use a bidding-agent GUI to support their responsible planners.

After accessing information about the task and checking the experience base, a planner who wants to accept the task for her company makes a bid for the task and writes it to the virtual marketplace.

The planner using the offering-agent decides which bidding-agent will receive the task by both checking the bid properties and the experience base for additional information like company rating and past experience with the bidding company. After granting the task to a bidding agent, that task will be marked as *sourced out* in the project. As a result, the winning bidding agent gets a connection to the workflow engine of the company to perform the task, accessing all necessary information about the task and related documents.

The concept

Negotiation protocol

Our design is based on the contract net protocol, which is a negotiation protocol proposed by [Smith 80]. It provides a model how agents can interact in negotiations. An agent with a task to offer, broadcasts a call for bids and waits for replies for some time. After this time elapsed, it awards a contract to the best bid (according to its own selection criteria). This protocol has been widely used and there are some expansions of that protocol like [SL 98]. We have expanded the protocol [KM99] and are implementing it based on JavaSpaces™.

Java Spaces and agent communication

In JavaSpaces™ context, a *space* is a shared, network-accessible repository that agents can use as a persistent object storage and information exchange mechanism. In our design, there is no direct interaction between agents. They communicate via objects in a space (the virtual marketplace). If an agent broadcasts an offer to other agents (as described in the Contract Net Protocol), he writes it to the virtual marketplace, addressed to everybody who is interested.

Agents communicate indirectly with other agents by exchanging objects via this space, direct communication is not necessary. When an agent wants to communicate with another one, he puts an object on the marketplace, filling the addressee slot of an object with that concerned agent (id).

Using a virtual marketplace offers a different way for agents to communicate: instead of using a standard agent communication language (ACL), we use a different protocol based on JavaSpaces-Technology and our semantics based on the object structure. Bidding and offering agents need to use interfaces to the human agent (graphical user interface), to the JavaSpace protocol provided by the virtual marketplace and to the experience base. The interface to the virtual marketplace is well defined by the JavaSpaces™ technology.

Object structure

The offer object is structured in three parts:

1. JavaSpaces™ specific

   The JavaSpaces™ specifics contain elements like the time an object resides on the marketplace.

2. Negotiation specific

   We put the negotiation logic into the objects that are put into and taken from the space. This information contain

   - the type of offering
   - the addressees (a list of identifiers, or a wildcard).
   - a sender-identifier
   - a task identifier

3. Task specific

   The task specific data is the core of the object. The planner has the possibility to decide which information will be inserted into the object that will be written to the marketplace.

   The data in the task specific part of the object contains:

   - Schedules for task and deliverables
   - Required skills for performing the task
   - Conditions for payment
   - Short description of deliverables of the task (e.g. code and documentation)
   - Type of equipment to be used (e.g. Hardware and Software)
   - Short description of given Inputs to perform the task
   - Standard or methods to be followed
   - Miscellaneous items (e.g. performance requirements)
   - Warranty
   - State of the contract (e.g. offered, committed)

The offering agent will propose most data entries, but it is the human agent who checks and (possibly) changes the information. I.e., the offering agent extracts the planned start and end dates from the project plan and shows a duration in the GUI, so the duration will be displayed to the planner, where she can change it or not.

The planner may also want to hide some information that is not relevant or sensitive to the company, like predecessor and successor tasks.

The tasks of offering agents and bidding agents

The tasks the agents need to perform are various: they need to parse objects from the marketplace as well as writing objects to it in a form that can be understood by other agents. Our work offers a concept and an implementation that enables agents to check if a task in the marketplace is relevant for them or not. An offer can be addressed to all agents, a group of agents or a specific agent in the virtual marketplace. The address field is also important for later phases in negotiation.

Hence, the task of a bidding agent in the context of this abstract is not only the visual representation of data from the marketplace. It is also a filtering of tasks based on available skills and resources of the company. Additionally, the experience base may provide information about the company offering the task. The experience base will provide standard artificial intelligence techniques like case based reasoning to support the retrieval of experiences.

Types of offerings

We will support different kinds of offering types. There can be closed (yes/no) offering types where a bidding agent can only accept or reject an offer. In an auction (best-bid offering) the payment is open and the bidding agent needs to name the amount he (the responsible human agent behind him) wants to receive when he accepts the task. In both cases it is in the responsibility of the offering agent to select the "winning" agent and to contact the bidding agent to confirm the acceptance of the offer. Sometimes, the best is not to choose the cheapest bidder or the first bidder accepting a closed offering. Information about former contracts with companies also performing on the marketplace are important criteria for the decision. Hence, an intelligent use of the experience base is of great importance. Information about other companies must be inserted into the experience base[1].

Another type of offering is the partial offering. This case is only possible with complex tasks that can be decomposed into several subtasks. A bidding agent receives a contract offer but can only handle some of the task that need to be performed. So, he puts a "partial bid" object on the virtual marketplace, where he can name the subtasks he wants to perform and the price he wants to collect. This case can lead to one of the offering forms described above, i.e. if the offering agent wants to accept the partial offering of the bidding agent but not for the price announced, he can put a closed offering on the marketplace, addressed to the bidder (the other agents will ignore this object when it is addressed to a special agent with a lower payment).

This information will be put into the marketplace object, after the human agent has checked and occasionally modified the data provided by the agent communicating with him. For the agent, it makes a difference if the task is a complex task –a part-of

[1] Building the experience base is not a task of the virtual marketplace component of MILOS. It only uses the experience base but does not insert data into it.

hierarchy of tasks- or if it is an atomic task. The information extraction and the information presentation on complex tasks is more complicated (e.g. loops in a task structure).

Integration of the virtual marketplace into the MILOS System

A first step is the use of JavaSpaces™ for the realization of a virtual marketplace and the creation of the two agents types (offering and bidding) for accessing the marketplace, including writing and reading of offer object data. In a next step, canonical GUIs have to be created for providing the parsed information to a user. The integration into MILOS will be done by expanding the (existing) GUI of the project planner with a possibility to write tasks to the marketplace.

Changes to the MILOS data model are new states for processes (e.g. “in negotiation” and “sourced out”). When the process is outsourced, the workflow engine needs to provide access to a process and its referenced products (documents) to a more or less unknown agent from another company.

State of implementation

We want to present a first version of the offering agent and the bidding agent at the workshop on ICSE-2000. Agents will be able to offer atomic tasks for closed and open offering.

The interface to the experience base will conceptually be developed in the second half of this year and implemented in early 2001.

Partial offerings of tasks will not been finished until the ICSE 2000, but is planned until September 2000.

One further step after the negotiation about tasks is the negotiation about resources. If a company has more resources than needed, it can offer them on the marketplace. Resource offering is planned until December 2000.

References

[Bendeck et al. 98] F. Bendeck, S. Goldmann, H. Holz, B. Kötting. Coordinating Management Activities in Distributed Software Development Projects, *IEEE Post-Proceedings of the 7th Intl. Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, Stanford (USA), 1998.

[MD 98] F. Maurer, B. Dellen: An Internet Based Software Process Management Environment. ICSE 98 workshop on “Software engineering over the Internet”, http://sern.cpsc.ucalgary.ca/~maurer/ICSE98WS/Submissions/Maurer/ICSE.html

[SL 98] T. Sandholm, V. Lesser. Issues in automated Negotiation and Electronic Commerce: Extending the Contract Net Framework. Readings in Agents, 1998.

[Smith 80] R.G.Smith. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver, *IEEE Trans. On Computers* C-29 (12), 1980.

[KM99]B. Kötting, F.Maurer, A Concept for Supporting the Formation of Virtual Corporations through Negotiation. *IEEE Post-Proceedings of the 8th Intl. Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*, Stanford (USA), 1999.

[BB 93] J.A. Byrne, R. Brandt, O. Port. The virtual corporation", Business Week, February 8, 1993.

[DM 92] W. H. Davidow, M. S. Malone. The virtual corporation: Structuring and Revitalizing the Corporation for the 21st Century, New York 1992.