CAPE Web Services: The COGents way†

Bertrand Braunschweig², Eric Fraga⁶, Zahia Guessoum³, Wolfgang Marquardt⁵, Othmane Nadjemi²,³, Didier Paen⁴, Daniel Piñol¹, Pascal Roux², Sergi Sama¹, Manel Serra¹, Iain Stalker⁶, Aidong Yang⁵

¹ Aspen Technology, Barcelona, Spain
² IFP, Technology, Computer Science and Applied Mathematics Department
³ OASIS team, Laboratoire d'Informatique de Paris 6, Paris, France
⁴ RSI - Réalisation en Systémique Industrielle, Meylan, France
⁵ RWTH Lehrstuhl für ProzessTechnik, Aachen, Germany
⁶ Dept. of Chemical Engineering, University College London, London UK

Abstract
The COGents approach to dynamic CAPE service composition uses the paradigm of multi-agent systems, where a number of software agents collaborate to configure a process model, according to a user's requirements defines using the OntoCAPE ontology. Our agents are "CAPE web service choreographers", building and running suites of CAPE-OPEN compliant process modelling components.

Keywords: Agent-Based Simulation, CAPE-OPEN, ontologies

1. Introduction
CAPE-OPEN (Belaud et al. 2002) defines syntactic interoperability of process modelling software components through commonly agreed standard interfaces for inter-component communication. Thanks to CAPE-OPEN (CO) interfaces, process modelling components can be assembled in process modelling environments at runtime, thus providing opportunities for using best-of-breed software in a variety of CAPE applications, from design to operation. CO provides the "plumbing" between CAPE components, that is, a communication mechanism by which software components can exchange functionality at runtime. CO semantics are implicit, i.e. only human providers and users of CO systems are able to decide which software component will best suit a specific application's need. In the COGents project (Braunschweig et al. 2002; Guessoum et al. 2003), we propose a framework for combining and assembling CAPE-OPEN compliant components using software agents (Batres et al. 2002); we define the categories of agents needed for the task; we develop an ontology, OntoCAPE, allowing our agents to reason with process modelling knowledge; and with this we can run three representative case studies in process design, process synthesis and process simulation. This paper presents the overall architecture and results of the project.

† Work funded by COGENTS, Agent-Based Architecture For Numerical Simulation, funded by the EC under the Information Society Technologies Programme, contract IST-2001-34431. Author to whom correspondence should be addressed: bertrand.braunschweig@ifp.fr
2. Functional blocks in case studies

We motivate our work through three case studies which provide a compelling case for our approach. Each illustrates a different aspect of the project and we discern a number of functions indispensable to each. Two case studies address different stages of the Hydrodealkylation of Toluene to produce Benzene (HDA Process) (Douglas 1988). A third considers the selection of models and solvers for a leacher unit in a simulation of the Polyamide-6 Process (Eggersmann et al. 2002).

Analysis of the case studies reveals a common core of indispensable functional blocks. These include: user interaction; specification of required software components; libraries and catalogues; matching with available components; selection; and integration into the calling environment. The first HDA Case Study treats preliminary conceptual design: using minimal input information, we wish to generate a ranked list of designs by making use of existing CO compliant software components to inform an existing synthesis tool, Jacaranda (Fraga et al. 2000); this requires additional functional blocks for problem clarification and identification of relevant processing technologies. The second HDA Case Study concerns simulation and makes use of the commercial tool HYSYS: the limits of the region of validity of a unit operation are exceeded and this requires that it be replaced by a different unit; this requires an additional functional block for monitoring. In the Polyamide Case Study a particular modelling strategy is adopted and software components relevant to it are sought from those available: this requires additional functional blocks for determining engineering purpose and choosing modelling strategy. The functional blocks identified from analysis of the case studies form the basis of the agents in the COGents framework.

![Figure 1: Some Functional Blocks: a box with sharp corners denotes a common functional block; rounded corners denotes an additional functional block; an ellipse signifies an activity.](image)

3. Technology: Ontology, Agents, Implementation Framework

OntoCAPE (Yang 2002) is the ontology used in COGents to support selection and use of PMCs. It essentially provides concepts for specifying modelling tasks and for characterising existing PMCs including software components implementing process
models and numerical solvers. A modelling task specification describes among other requirements the process system object to be modelled, the properties to be specified and those to be calculated, and the preferred laws and property models that should be applied in the model. An existing process model component is characterised by the object modelled, laws and property models adopted, model variables, some mathematical characteristics, as well as software properties such as supported interfaces. An existing numerical solver component is characterised by properties of numerical algorithms as well as the software aspect.

To meet the requirements from COGents but also to be extensible for more general purposes, OntoCAPE has been designed to have two different parts, namely the part of common concepts sharable by different CAPE applications and the part of application-specific concepts. Each of the two parts contains a number of partial models. OntoCAPE has been developed on the basis of CLLP, a comprehensive data model for process engineering (Bayer and Marquardt 2003). OntoCAPE is formally specified using DAML+OIL, an ontology modelling language widely used in the semantic web society (cf. www.daml.org). In reacting to the emergence of the new ontology modelling language standard OWL (cf. /www.w3.org/TR/owl-features/), an assessment has been made which shows the feasibility of migrating OntoCAPE to an OWL ontology in the future.

Agents
Multi-agent systems provide a powerful paradigm for the modelling and the development of complex systems. They are based on the decomposition of systems into several interacting and autonomous entities. According to the definition of (Shoham 1993), "an agent refers to an entity that functions continuously and autonomously in an environment in which other processes take place and other agents exist".

To facilitate the design and dynamic modification of component-based models, we have adopted a distributed architecture based on the multi-agents paradigm. Each agent has specific capabilities and exchanges information by message-passing. The agents society is illustrated in Figure 2. For example:

- **Wrapper Agents** manage the interaction between the existing software (Process Modelling Environments (PMEs), Process Modelling Components (PMCs), …) and the other agents of COGents. They may be distributed on the Internet or on intranets. Their
Functionalities are represented by services. Each Wrapper Agent holds therefore a set of services that can be provided to the other agents.

- **MatchMaker** selects those components from an available library of components (a catalogue), from the Internet and/or from intranets, that match a given description of the problem or a task specification.
- **ModellingTaskManager** interacts with **Personal Assistant Agent** and uses OntoCAPE to define a consistent specification of the modelling task.

The arrows of Figure 2 indicate the default flow of information between the agents. For example, **Modelling Task Manager** sends a task specification to **MatchMaker** to select the components that match this modelling task specification. To find these components, **MatchMaker** needs to interact with **Library Wrappers**.

**Framework**

The ambitious objectives of COGents require the assembling of many different technologies for its success. The demonstration that all the best-in-class technologies used integrate smoothly in a single framework has been one of the achievements of the COGents project. The figure below offers a high level view of the framework and its positioning above the existing CAPE-OPEN paradigm.

The COGents framework is built on top of the original CAPE-OPEN (CO) framework. The CO standard interfaces, allow seamless interoperability of process modelling software from various suppliers through a set of published standard interfaces for the main categories of software modelling components and environments. The interoperability works on top of standard middleware services, such as COM and CORBA.

![Figure 3: COGents framework](image)

The extended functionality of COGents is provided by a multi-agent system, the boxes just over the CAPE-OPEN components. This functionality includes negotiation mechanisms for composing the simulation during the design phase, as well as runtime facilities such as diagnostics and guidance to the users. These negotiations are managed by the multi-agent system, represented in the diagram by the DIMA block. The DIMA
block provides the infrastructure for the multi-agent system: it launches the supported agents, it is aware of their termination (control the agent execution cycle), and it handles all the semantic communication functionality.

The communication between individual agents is done with messages exchanged using ACL, whose content is expressed using the ontology for process modelling developed for the project, OntoCAPE. The multi-agent infrastructure provided by DIMA is complemented with the DARX block, which extends DIMA with a middleware to deploy agents on a network of machines. DARX provides a global naming and location service for agents. For those remote communications, COGents integrates a security layer based on SSH, which provides strong authentication and secure communications over unsecured channels such as the Internet.

4. Some results of COGents

Using the framework we can run the three case studies. For example, the second case study has been implemented using HYSYS\(^1\). Recall, the objective is to replace a unit which does not function as desired. The user defines a Modelling Task Specification (MTS) in OntoCAPE format to describe the unit he requires in terms of functionality and parameters using HYSYS, which translates its own XML unit definition into OntoCAPE format. Library and match maker agents find the appropriate Unit Operation using the generated MTS file. A variation of this has also been implemented in INDISS\(^2\). It assumes that the modelling task specification exists and deals with a request from the user of the simulator, to find a component that matches new requirements defined in the MTS. The PME sends a COM event to the wrapper agent with the modelling task specification as a parameter. Then, we use library agents to search for available OntoCAPE model component descriptions. The matchmaker agent finds inside the OntoCAPE file of the selected model component the ID of the CO component. The PME wrapper uses specific services to create the new unit operation on the flowsheet, connects it and deletes the previous unit operation. Parameters and material connections are set using CO interface calls from PME wrapper.

5. Conclusion

The advent of internet-based service frameworks (Sama et al. 2003) allows us to envision new facilities for the dynamic configuration of CAPE applications from software components available on the network. Frameworks such as W3C's web service infrastructure provide means to locate a software service using web catalogues, to define what the software does, and how to operate it. Such representations can be interpreted by machines thanks to extensive use of ontology languages on top of XML and RDF. Recently proposed extensions of web services such as the DARPA Agent Markup Language for Services "DAML-S" (Ankolenkar et al. 2002) even go beyond, by proposing ways to describe the services' internal processes so that software agents

---

\(^1\) HYSYS is a CAPE-OPEN compliant steady-state PME from AspenTech
\(^2\) INDISS is a CAPE-OPEN compliant dynamic PME from RSI
would not only be able to locate and activate services, but also understand how to best use and combine them.

Our approach to dynamic CAPE service composition uses the paradigm of multi-agent systems, where a number of software agents collaborate to configure a process model, according to a user's requirements. For process modelling, our agents act as "service choreographers", building and running suites of CAPE-OPEN compliant process modelling components. In the case of process design, our agents emulate a design team: clarifying problem structure, identifying generic processing technologies and seeking appropriate CAPE-OPEN compliant resources to inform an automated process design tool. This approach brings added value to the CAPE-OPEN standard, introducing semantic information and supporting industrial users of CAPE software in developing complex applications using the best available knowledge.

References


