Agents Implementing Subject Behaviour: A Manufacturing Scenario 12

Udo Kannengiesser

Abstract

This chapter presents a scenario for the use of agents in the implementation of subject-oriented process models. The scenario is set in a manufacturing company that has already used S-BPM in office-based business processes and now wants to apply this approach on the shop floor using agent technology. The chapter describes a project team meeting in which S-BPM specifications for a pressing process are developed and concepts for the agent-based implementation of that process are discussed. During the meeting, a number of issues are raised that are a consequence of using computational rather than human agents for implementing subject behaviour. The key issues include:

1. An open world of agents: Computational agents can be created on the fly, in various compositional structures and embodiments (virtual, physical, or both) that can change dynamically. As a result, mapping subjects to computational agents is typically more challenging than mapping subjects to human agents as human organisations are often assumed to pre-exist and remain relatively stable during process execution.
2. Access control mechanisms and cognitive capabilities: Explicit mechanisms for controlling which agent can execute a subject need to be engineered, taking into account process requirements, agent performance characteristics, and the current execution status. These mechanisms often require the implementation of some form of cognitive capabilities in the agents, including heuristic reasoning and planning—which are usually taken for granted in human-based processes.

U. Kannengiesser (✉)
Metasonic GmbH, Münchner Straße 29, 85276 Pfaffenhofen, Germany
e-mail: udo.kannengiesser@metasonic.de

A. Fleischmann et al. (eds.), *S-BPM in the Wild*,
DOI 10.1007/978-3-319-17542-3_12

While these issues are discussed in a manufacturing scenario, they are generic in that they can be transferred to agent-based implementations in other process domains.

12.1 Introduction

It is 7:55 a.m. on a Monday morning. Peter Smith, 45, walks into his office, starts up his laptop and quickly grabs a cup of coffee from the kitchen next door. Peter is the CEO of a medium-sized manufacturing company producing parts for the automotive and aerospace industries. Today he is in an excellent mood, because a new project is about to commence that he recently proposed and he is very excited about: "Agent-based manufacturing processes" is the working title that Peter chose for his project. The kick-off meeting is in just a few minutes.

"Good morning," Peter says when entering the meeting room. The members of his project team are already sitting around the table: John, 53, production manager, Jerry, 34, automation engineer, and Diana, 41, head of purchasing. Peter sits down and says: "I already sent you an outline of the project, but let me recap what the project is about and why I chose you to be involved in it." "Yes, that would definitely be useful," John remarks, "as I haven't quite understood some of the things you mentioned in your email."

"I guess you mean the terms 'industry 4.0', 'cyber-physical systems' and 'agents'." Peter takes a sip from his coffee and then starts explaining, "OK, so let's start with giving you some background on these terms. Industry 4.0 is the name of what is often proclaimed as the next industrial revolution. I learned about it just a few weeks ago at a trade fair on production automation. Many of the speeches and even a whole section of the exhibition hall were devoted to that topic. It is essentially about enabling new models of production management by using embedded devices that monitor and control physical processes and communicate over wireless networks. These devices are called cyber-physical systems, because on the one hand they are physical objects, and on the other hand they have a representation in the virtual world—a second identity if you like. Almost any object you can find in a factory can be conceived of as a cyber-physical system: a sensor, an actuator, a conveyor belt, a product, a whole machine—."

"Uhm… hang on a second," John interrupts. "This sounds all great and futuristic, and I sort of heard about these things too. But what problem does it solve? What's the benefit of having all those cyber-physical systems everywhere?" "Well, I've been wondering about that too," says Peter. "At first glance, it seems to add a lot of complexity and communication overhead. On the other hand, cyber-physical systems allow you to decentralise your production planning and control, which gives you much more flexibility and agility. For example, these systems are able to sense unforeseen events such as new production requirements or breakdowns of individual resources, and then reconfigure the production process autonomously and in real time. Remember last month when we had to stop our whole production because of a defective sensor? We had to spend five hours searching for the problem and replanning the process. I believe this could have been handled much faster with the kind of self-organisation capabilities provided by cyber-physical systems." Peter pauses for a few breaths before continuing, "You could even endow cyber-physical systems with knowledge and goals—such as, 'I want to find the most energy-efficient path for the product through the production line'—so you could have some sort of intelligent behaviour. And this brings me to the concept of an agent. An agent is a piece of software that interacts with its environment in a goal-directed, autonomous way, often involving communication with other agents. It is often attributed to notions of human cognition and intelligent behaviour."

"Yes, agent-based systems have been around for quite a while," Jerry adds and adjusts his glasses, "I have already programmed such a system for my university thesis." "I know," Peter looks at him, "and that's why I involved you in this project. Your experience with agent technology will be extremely useful, because the goal of this project is to develop an agent-based cyber-physical production system and trial it on the shop floor." Peter turns back towards John and says "Don't worry, John, this will only be a pilot implementation involving a very small, isolated part of our production system. There won't be any risk of interrupting our core operations if things go wrong with our prototype. Only after extensive testing will we think about rolling it out on other parts of the shop floor."

Diana, who so far has only been listening, now joins the conversation. "Sorry, Peter, what I still don't understand is what my role is in all this. I have no experience whatsoever in agents or production engineering!" Peter looks at Diana and his eyes seem to twinkle. "But Diana… you are the most important person here! You will guide us in the development of a process architecture for our agent-based production system. This architecture is necessary to capture the bigger picture and specify how the system should work in terms of functionalities—and help us understand the impact of any system changes that may be required after implementation. We will use S-BPM for specifying the process architecture. This is because S-BPM has well-defined execution semantics yet can be easily used by domain experts without IT background. So people like John and myself can have a say in the system design, because we can understand and directly change the process model if we see a need for that."

As Diana still does not look convinced, Peter adds, "You will be the perfect facilitator for developing an S-BPM process model. Haven't you been in exactly that role in your department for many years now?" Diana thinks for a few moments. "Yes sure, I've been in this role, but the processes we are dealing with in the purchasing department are completely different from the ones in production. And our processes are all executed by humans, not machines or agents." Jerry quickly jumps into the conversation. "Well, on a conceptual level humans can be viewed as agents too. Agents can be any autonomous entity—human or computational. And in fact, most models and architectures of computational agents are inspired by psychological and social accounts of human agents."

"That's right." Peter looks at each team member to make sure he has their full attention before continuing, "What we want to achieve in this project is an agent-based system whose design and execution is driven by S-BPM models. Why S-BPM? Firstly, it puts a process layer on top of the specific technologies used, such as production technology and agent technology. This additional layer ensures everyone can easily understand the overall process and improve it when necessary. There's a close connection between S-BPM and multi-agent systems, as they share many fundamental ideas: decentralised control, autonomy, concurrent behaviour, and communication. And secondly, S-BPM models are unambiguous and directly executable; so the executed process is not just some IT guy's interpretation of a flow chart, but the result of an automatic, model-driven transformation. The constructs of the S-BPM notation are very generic and can be applied to any kind of process, no matter if it's a business process or a production process. We had excellent results introducing S-BPM in our purchasing department a few years ago—let's try and reap similar benefits with S-BPM in our production department!"

12.2 Specifying a Production Process in S-BPM

Peter takes a whiteboard marker and a set of blue magnetic cards in his hand. "So I thought of using parts of our pressing department as a test bed for an agent-based production system. What would an S-BPM model of the pressing process look like? What are the subjects?" John frowns and asks, "Sorry, can you remind me what subjects are?" Diana quickly replies, "Subjects are the active entities, or actors, in a process. But by subjects we don't mean any specific actor—we rather mean the abstract functionalities required in a process. So at the level of subjects, we don't really care about which actor or agent will implement and execute the functionalities. To give you an example, in a purchasing process we usually have a subject or functionality called 'Purchaser' and one called 'Approver'."

The frown on John's face slowly disappears. "OK, so in our pressing process, one obvious subject would be, I suppose, 'Pressing'." Peter writes the word "Pressing" on a blue magnetic card and sticks it on the whiteboard. "Well, and there are plenty of others," John continues to come up with possible subjects in the pressing process, now with increasing motivation as he begins to understand the notion of a subject. "Before we can use the press, we need to cut blank plates from a coil. These plates are then transported and placed into the press. The pressed part is transported to quality testing and then stored in a warehouse. So the subjects would be 'Blanking', 'Transportation', 'Quality Testing' and 'Storage', I reckon."

After writing these subject names on separate cards and adding them to the whiteboard, Peter says, "OK, so all these are subjects that provide services related to transforming physical materials. Do we need any subjects for providing information, like telling the other subjects when or how to provide these services?" John thinks for a while and then replies, "Well, we need functionality for scheduling the production, and perhaps one responsible for launching the individual production services." Peter writes "Production Scheduler" and "Production Service Launcher" on two magnetic cards and adds them to the whiteboard.

Peter and his team now go on and think about the messages to be exchanged between the subjects. After some discussion, they agree on a set of messages that Peter represents by drawing arrows that connect the different subjects on the whiteboard. The final subject interaction diagram is shown in Fig. 12.1.

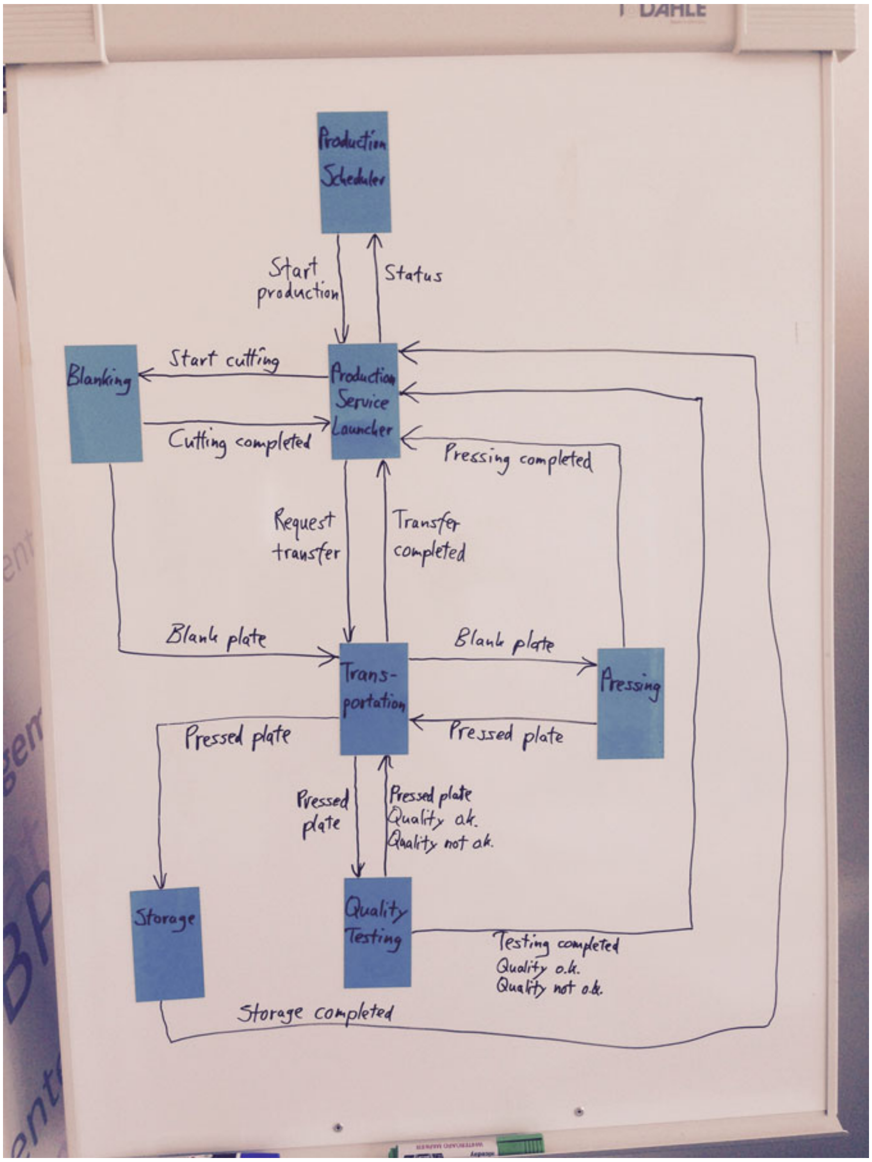

Fig. 12.1 Subject interaction diagram of the pressing process

Peter looks at Diana and asks, “What would you suggest as a next step after agreeing on the subjects and their interactions?” “Well, in our department we would now assign a subject owner to every subject and let them model their internal behaviour. So, uhm…” Diana stops talking and looks slightly confused by the whiteboard. “The problem here is that some of the subjects are machines, or software agents that control a machine, and we can’t really ask, say, the blanking machine to model its own behaviour.” “That’s right,” Peter nods, “in this case we would have to ask an automation engineer to be the subject owner and model the behaviour of the Blanking subject. Someone like Jerry…” “Sure, let me have a go,” says Jerry and comes over to the whiteboard. “Take these cards,” Peter hands over a set of green, yellow and red magnetic cards while explaining, “green is for receiving a message, yellow is for performing a task on your own, and red is for sending a message.” Jerry takes the cards, picks a green one and sticks it on the whiteboard. “So, the Blanking subject becomes active when it receives a message from the Production Service Launcher to start cutting.” Jerry writes “Receive cutting order” on the green card. “Then the Blanking subject draws some material from the coil, cuts off a piece and places it in the output tray.” He takes three yellow cards, places them underneath the green card and writes on them “Draw material from coil”, “Cut material” and “Place blank in output tray”. “At the end, the Blanking subject notifies the Production Service Launcher that the cutting task is completed and sends the blank plate off to the Transportation subject.” Jerry places two red cards underneath the others, labelling them “Notify cutting completed” and “Send blank plate”. Finally, Jerry adds a yellow card labelled “End” to the very bottom and draws arrows between the cards. The complete subject behaviour diagram he produced is shown in Fig. 12.2.

Jerry explains, “I’ve created a very simple behaviour for now. I will refine it later, depending on how I design the specific control architecture of the agent.” “Yes,” Peter adds, “you might even decide to decompose the Blanking subject into a set of more fine-grained subjects. Each of them might again be executed by an agent, for example, an agent for the coil, an agent for the saw, and so on.” Diana nods and says, “Yes, the Blanking subject would then be what is called an interface subject: A subject that establishes the interface to another process whose details we don’t need to care about. For example, some of our processes in the purchasing department include external ‘Supplier’ subjects whose internal operations are unknown to us because they are executed in another company. All we care about is that they accept and provide the messages we expect from them and at the right time. This is one of the key concepts of subject-orientation.” “That’s right,” Peter adds, “S-BPM provides a means to connect any kind of process, where the different processes are defined according to the different views and interests of stakeholders. In the business domain, we typically deal with processes across different companies. In production, we are more concerned with processes across different technical domains and different levels of detail. The mechanism with which we can connect the different views and processes is the use of interface subjects. They provide exactly what the other process needs to get or needs to know, and nothing more.”

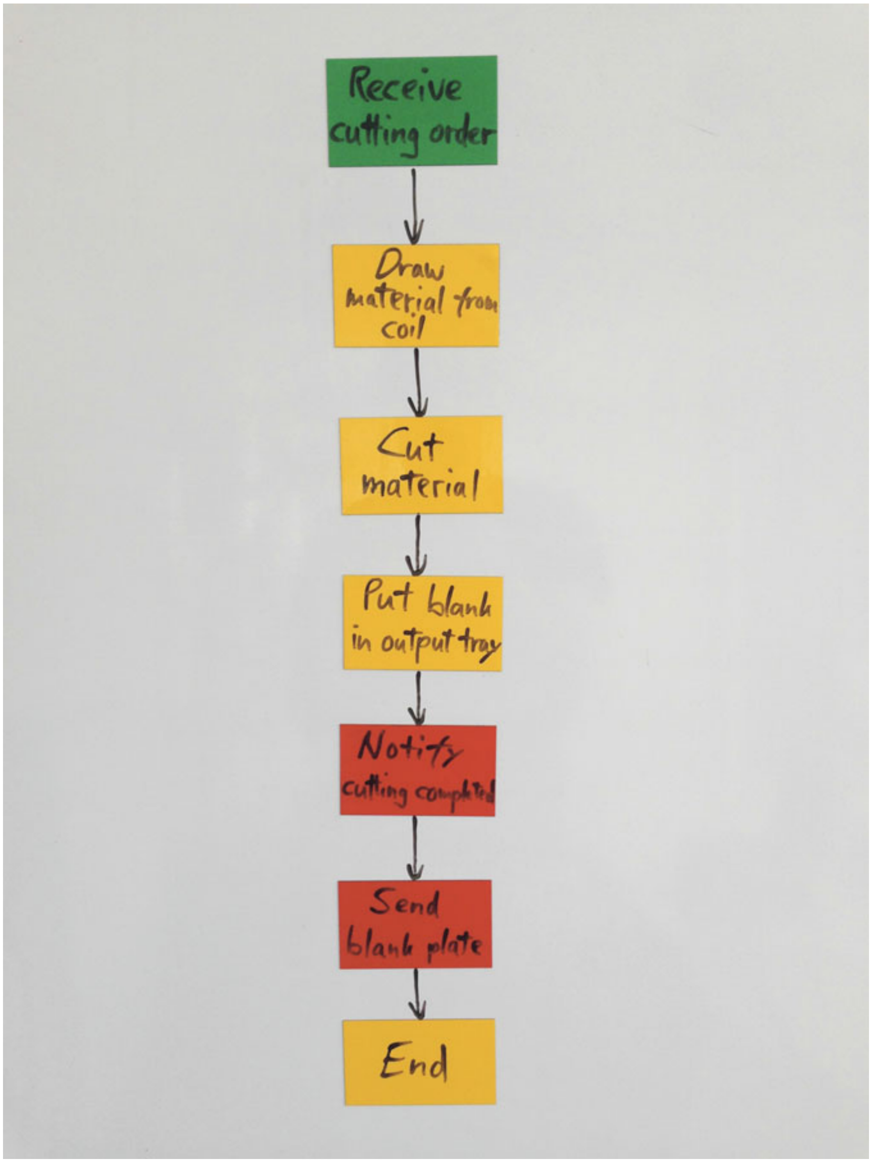

Fig. 12.2 Subject behaviour diagram of the blanking subject

"OK, but back to the question about how to specify the behaviour of a subject when it's not humans but agents who implement that subject," Diana says with a contemplative look, "We can't really decide on the final specification of behaviour before we know which agents will execute it." "Yes," Peter agrees, "behaviour models need to evolve as we go about engineering the system. And S-BPM lets you do this more easily than other approaches, because the S-BPM notation is simple and tailored to the first-person perspective of subject owners."

12.3 Mapping Subjects to Agents

After a 15 min break, the project team comes back together to model the behaviour diagrams of all the other subjects. After finishing these diagrams, Diana looks at them and says "OK, now we could already validate the models, by assigning different people to the subjects and let them play through their behaviour and check if all subject behaviours are correct and complete." Peter says, "I agree in principle, but given we have seven subjects in the process model but only four people in the room I suggest doing the validation later. For now, what I'd like to do instead is to think more carefully about the issues we need to solve when implementing subjects using agents."

Peter walks to a flip chart and creates a list of all subjects in the pressing process. For every subject he quickly writes down a set of agents he thinks would be needed for executing each subject's behaviour. His mappings between subjects and agents are shown in Fig. 12.3.

The agents in Peter's list include production machines and human workers. There are also two question marks, associated with the subjects "Production Service Launcher" and "Production Scheduler". Peter points to these subjects. "These subjects represent quite abstract functionalities—can we map them onto something more concrete?" "That would be good," says John, "because it's not easy to talk about a subject without having a concrete picture of it. When I think of a production process I usually think in terms of physical things, like machines, materials, people and so on." "I agree," Peter looks at John, "we should definitely link the abstract functionalities with something more tangible—because in the end we need to do that anyway. Functions need concrete actors that implement them, otherwise the process never comes to life." "OK," John walks up to the flipchart and places his hand next to the 'Production Scheduler' subject. "For this subject I can visualise our manufacturing execution system as being able to implement it." Peter crosses the question mark next to Production Scheduler and writes down "Manufacturing Execution System". Jerry continues, "And I reckon, the 'Production Service Launcher' is just some kind of software programme that coordinates all the services needed for manufacturing an individual product by communicating with all other subjects, including those that are implemented as physical resources like machines. But it would be nice to also have a physical identity or embodiment for this software programme—a physical object that is in contact with all other physical resources…" "Maybe it's the product being manufactured?!" John interrupts Jerry. Everyone looks astonished. After a short moment of silence Peter says, "Yes, the product being worked on could be the agent implementing the Production Service Launcher. Interesting idea, John! This fits nicely with a popular vision for future

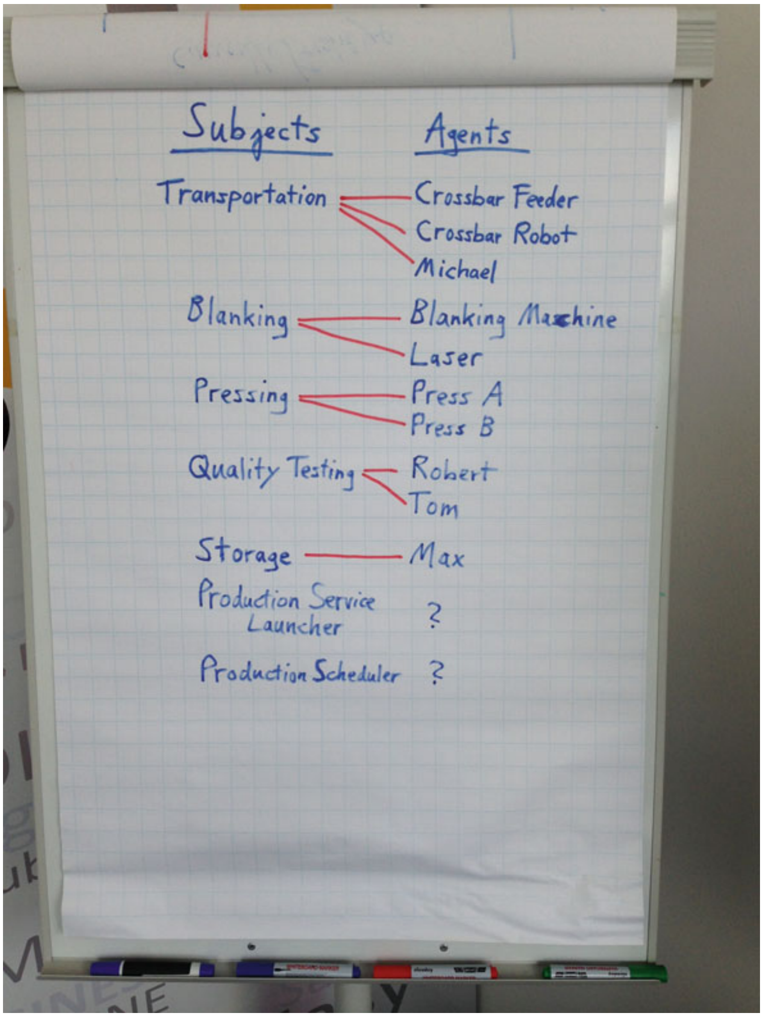

Fig. 12.3 Associating subjects with specific agents

manufacturing systems that I often heard in industry 4.0 talks: intelligent products determining their own path through the factory. By knowing their individual history, their current status and their target state, these products can dynamically react to changes or breakdowns by selecting alternative production paths."

Peter, John and Jerry look enthusiastic. Only Diana is still not convinced. "But the product doesn't really exist before executing the process. At the beginning, there is just this coil. Only when a piece of it is cut off could I imagine it to be a product-to-be." Jerry replies "Yes, the physical embodiment of the agent takes shape only during the process. But the agent can be instantiated in the virtual environment already before cutting the coil. It's like a ghost that is invisible at first but then slowly materialises as the production process unfolds." "You're a spook, Jerry!" Diana laughs, "But OK, I do think the whole process model now becomes much more tangible and intuitive. The only thing I still don't get my head around is how this ghost, this product agent, comes into existence in the first place. If you have hundreds of products to be produced, you also need hundreds of product agents. Who programs all these agents? And who can tell in advance how many of them will be needed?" Jerry points to the subject interaction diagram and says "Well, product agents wouldn't be pre-programmed manually on stock; every agent would be spawned when the production of a new product instance is launched by the Production Scheduler. This can be done automatically, as all instances of the product agent are identical in terms of their software architecture and their initial knowledge. We just need to define a template that is then used for instantiating product agents by some kind of agent factory. So the internal behaviour of the Production Scheduler subject might need to include an activity that creates a new product agent before sending the "start" message to that agent."

Diana scratches her head. "Wow, that's so different from the processes we're dealing with in the purchasing department, where all agents are human and cannot be just "spawned" or cloned on demand. We can only hire new people and then train them, and this often takes a very long time. We basically need to map our processes to an existing organisational structure that doesn't change much over the years. When you have software agents instead of people, the possibilities are almost unlimited—you can just create new agents in next to no time!"

Peter is impressed by the many ideas and insights generated by his team. "I like how we manage to bring the two worlds of subject- and agent-orientation together. I think we're onto something big here. I still have a question though. For implementing the 'Transportation' subject, we have three agents: A crossbar feeder, a crossbar robot, and Michael. But both the crossbar feeder and the crossbar robot can execute only some parts of the subject behaviour: transporting the blank plate from the blanking machine to the press, and transporting the pressed plate from the press to the quality testing station. This is because there are clear physical constraints that allow only certain movements on the shop floor. Moving the pressed plate to the storage area can only be done by manual work, which allows more flexible movements than crossbar feeders or robots. Of course it's not completely manual—when Michael moves the plates around, he uses a trolley for smaller plates and a forklift for larger ones. Now the question is, do we need some form of substitution mechanism to enable agents' replacing each other at execution time?"

Jerry thinks about this for a while and replies, "I think a much easier way is to create a new 'transportation' agent that is composed of three other agents—the feeder, the robot, and Michael. So this would be some sort of 'super-agent', a central control structure for a set of other agents or components specialised for different tasks. It's a bit like Michael who uses different tools for moving the pressed plates." Diana adds, "It's actually very similar to integrating S-BPM models in an IT environment where you also have many different IT tools for executing the same process. Many of our business processes, such as applying for vacation, need to be integrated with various email and calendar programmes, the time recording system and sometimes the project management system. No single IT system can support all tasks—they are all needed together for executing the complete subject behaviour. Our IT guys perform the IT integration by programming refinements for specific states in the behaviour specification." Jerry says, "You could either do that, or give the agent direct control over how it integrates the execution in its components. So rather than executing pre-programmed refinements, the agent could flexibly decide which component is best suited for the situation at hand. This would be useful when the agent needs to consider different alternatives for executing the same task, such as using either the feeder or the robot for transporting a blank plate to the press."

Peter summarises, "So, once again, we can see there's a great deal of design freedom in conceptualising and implementing agents for executing subject-oriented process models. We can view every single machine or person as an agent, or we can aggregate some of them to form a composite agent that has centralised control over its individual components. We can hard-code the physical or IT integration of an agent's behaviour at design time, or we can allow the agent to reason about and modify this integration at runtime."

12.4 Developing Control Mechanisms for Subject Execution

After another short break, it is John who resumes the conversation. "OK, so what exactly does the interplay between all these agents look like? Who exactly talks to whom? How can I ensure that I want to use Press A and not Press B for a particular product?" Diana answers, "In S-BPM when you send a message, you have the choice of who to send it to. You can either leave the recipient unspecified—in this case any agent associated with the receiving subject can choose to accept the message and thus commit to executing the subject—or you can send it to a specific agent or group of agents associated with the receiving subject." "Well, then again, what happens if you no longer have people but software agents to take this decision?" Jerry argues, "In the absence of human intelligence you need a well-defined mechanism for this kind of decision making."

Peter walks to the whiteboard and writes "Mechanisms" on top. "What kinds of control mechanisms for the runtime selection of agents can we think of?" Jerry quickly comes up with a few ideas. "So, in principle we could have a first-come-first-serve mechanism, just as it is now in Diana's department: When a message is sent to a subject, any agent associated to that subject can receive the message, and the first agent to take the message will be the agent executing the subject's behaviour. Of course, we might also think about randomising the message allocation, to prevent messages from being routed to agents simply on the basis of their different reaction times. Peter writes "1st-come-1st-serve" on the whiteboard, and then "Random" underneath. Jerry continues, "We may also think about introducing a system where agents can bid for receiving particular messages, using credits they earned by having done a good job in the past. The agent with the highest bid gets the new job—that means the incoming message in our case. Bidding mechanisms have already been applied for allocating manufacturing tasks among agents in the automotive industry." Peter writes down "Bidding" on the whiteboard. Diana asks, "Could we also have a mechanism similar to human decision making? One where different alternatives are compared and evaluated against some form of criteria?" "Well, I'm not sure if human cognition really works that way," Jerry adjusts his glasses, "there's actually strong evidence that suggests that humans typically use simple heuristics instead of complex analyses. But I do agree that some form of rational, analytical decision making would be useful for our system." Peter writes down "Heuristics" and says, "Let me call both just 'heuristics', without caring about whether they're simple or complex."

"By the way, 'first-come-first-serve' can also be seen as a heuristic," Jerry adds, "namely when the sender applies this mechanism to its memory of possible recipients. So the sender would choose the first agent that comes to its mind, perhaps based on some key feature that was used for cueing its memory. For example, when I'm asked to think about a city in Europe, the first cities popping up in my head would probably be Paris, Berlin or London, because they are among the biggest or the most popular ones."

Peter steps back from the whiteboard and looks at it from a little distance. "So, there's definitely some overlap between the mechanisms, but that's OK. Maybe they can also be combined. We can decide that later. But what I'm thinking now is that we might include a second dimension that is about where the execution of these mechanisms may be located: on the side of the sender or the receiver of a message." He writes "Sender" and "Receiver" in two new columns on the top of the whiteboard. Jerry adds, "And there might also be a third party, something like a facilitator, who can execute the mechanisms." Peter writes "Facilitator" in a new column, resulting in the table shown in Fig. 12.4.

"There may also be combinations within this dimension, where the sender and the receiver, and perhaps also a facilitator, cooperate on executing a mechanism," says Jerry. "A well-known example in agent-based systems is the 'Director Facilitator' approach. The facilitator can be thought of as the 'yellow pages' of the agent system. All agents must register their services with the Directory Facilitator. When you need a specific service, you can just ask that facilitator to give you a list of agents that provide that service, and you can then choose a specific agent or perhaps let the facilitator choose for you."

Peter is content with the table the team just produced. "We can use this table as a basis for developing control mechanisms for every message exchange defined in the subject interaction diagram. John, which of these mechanisms would be suitable for your problem of selecting the right press to send your blank plate to?" John thinks for a while and says, "I reckon it should be the heuristic approach, because there's a great deal of specific knowledge involved in selecting the press. On the one side we have product specifications and production requirements that need to be satisfied, and on the other side we have production services with their specific capabilities

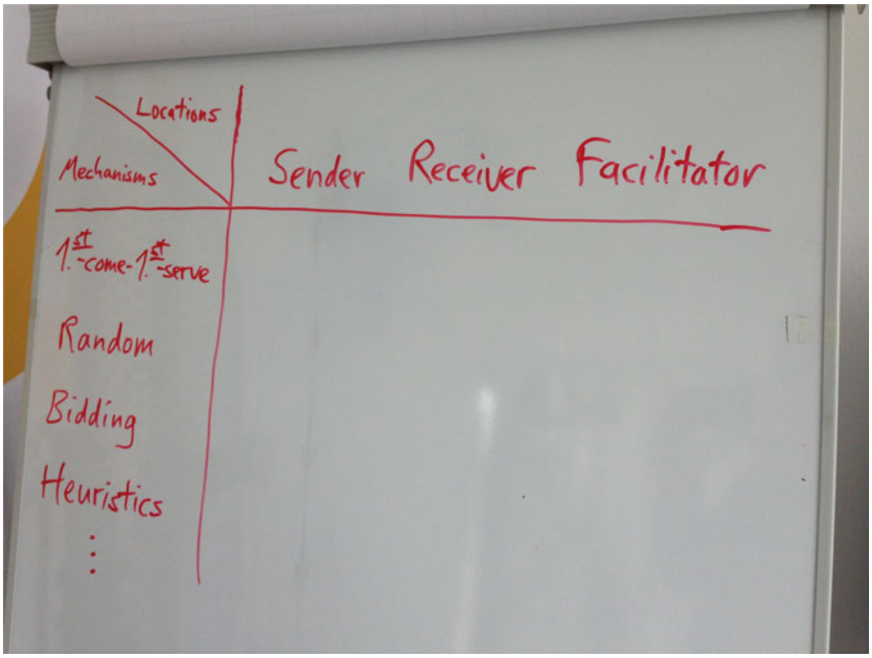

Fig. 12.4 A few possible mechanisms for selecting agents to receive messages, and possible locations of these mechanisms in the sender, the receiver, or a facilitator

and performance values. Both sides need to be matched. For example, if I need a specific product to be produced within a certain timeframe or satisfying certain quality constraints, some of the services might not fulfil these requirements." "Yes," Jerry remarks, "and the Directory Facilitator approach I was just talking about seems to be appropriate here, because the basic capabilities and many performance values of the production resources do not change over time, so they can be stored in a central directory."

Peter hands the whiteboard marker over to Jerry. "Could you try to model the process of querying and selecting a service via a facilitator—that is, selecting the agent providing that service? This would help us understand the selection process—besides making it ready for execution." "Sure," Jerry walks to the whiteboard and places three magnetic cards onto it. While writing on them he explains, "We need three subjects: a 'Service Requestor', a 'Directory Facilitator' and a 'Service Provider'." He quickly adds a few message arrows to produce a subject interaction diagram as shown in Fig. 12.5.

"The diagram is pretty self-explanatory," Jerry says. "The Directory Facilitator identifies the agents that can provide a specific service, and the Service Requestor selects from among these agents based on some criteria or priorities and finally asks for a commitment from the Service Provider to deliver the service." John remarks, "Well, the most frequently used criterion for deciding on a production service

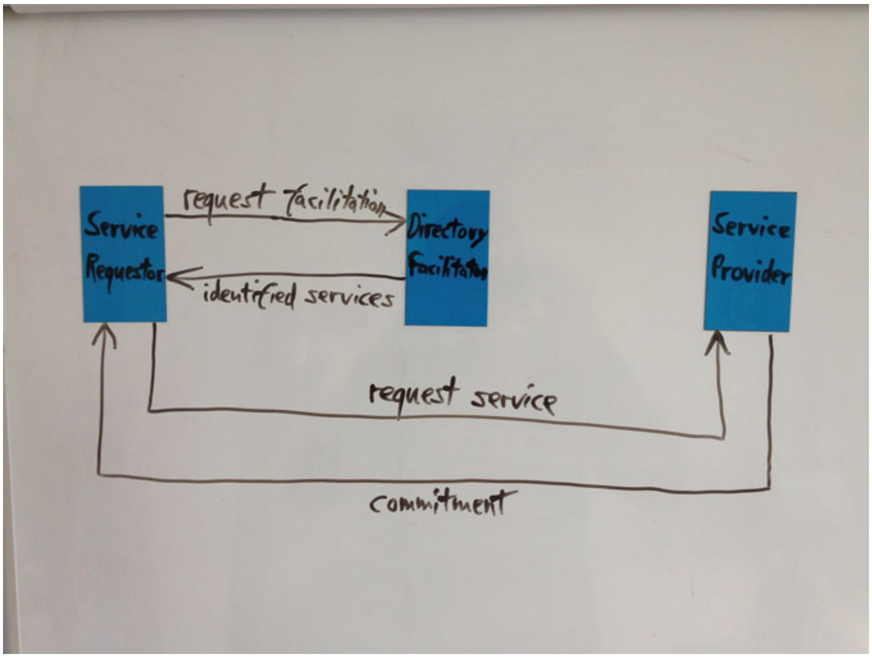

Fig. 12.5 Subject interaction diagram of an agent (service) selection process

would probably be the current availability of the service. If a press is busy with another production job, I usually need to select a different press—preferably one that's idle. Would the Directory Facilitator know about this status information?" After thinking for a short while, Jerry answers, "If the Service Requestor is executed by the same agent that also executes the Production Service Launcher in our main process—that is, the pressing process—then the agent can infer the current status of a machine from the status of the process instance." Jerry points to the subject interaction diagram shown in Fig. 12.1. "So, for example, if the Production Service Launcher was just told by the Transportation subject that the blank plate has been delivered to the press—using the message 'Transfer completed'—but is still waiting for the notification message 'Pressing completed' from the Pressing subject, then the Production Service Launcher can infer that the press is still busy."

Peter summarises, "So the logical connection between these two processes is established in the knowledge of the agents involved in them. The agent executing the Production Service Launcher would also need to execute the Service Requestor, because it knows that the results of the agent selection process are needed to perform its tasks within the pressing process." "Yes," Diana adds, "when people execute processes in our department, they also have a range of processes they can select. For example, before booking a business trip they might realise that they are running low on business cards, so they would run the 'ordering business cards' process before running the 'business travel booking' process, to make sure they get the new business cards before travelling. Of course, we might think of connecting the two processes on the modelling level, if there is a close connection between the two. Although in this example, I would say that these processes are quite independent of each other. So we wouldn't need to include a task in someone's travel booking behaviour to check if sufficient business cards are available and if not then to launch the card ordering process. We trust that our staff can organise themselves enough to make the connection between the processes in their minds and choose the right processes to be executed at the right time."

"Good observation," Peter compliments Diana. "So for our two separate models, we would need to implement some form of common sense reasoning in the agent to decide which process to execute at what point in time. Jerry, what are your thoughts on this?" Jerry answers, "That's certainly possible. Many agent architectures incorporate cognitive notions such as goals, beliefs and plans. We could use these architectures to represent the goals associated with different processes. Our two processes would then be connected with each other because the goal of one process is to support the other process."

Peter asks Jerry, "What other issues do we need to consider for implementing the agents and the control mechanisms we just discussed?" "Well, there are always a number of points to be thought about when implementing agent systems. For example, what specific decision-making mechanisms and learning capabilities should we implement? What are the real-time requirements? Do we want a closed system that has always the same set of agents, or an open system where new agents may come in dynamically? What are then the implications for interoperability; what types of knowledge can be predefined among the agents and what types of knowledge are unknown prior to running the process?" Diana agrees, "This is also important for defining data objects—or business objects, as we call them—that we need to add to the messages and behaviours in the S-BPM models."

Peter looks at his watch. "Well, it's almost lunch time, so let's wrap up our meeting. Today we outlined the major challenges to be addressed when designing and executing our agent-based production system based on the S-BPM approach. I think we are now in a position where we are able to define concrete work packages and schedule our tasks. But we will do this in our next meeting. For now, let me just summarise the main issues and what we learned from this morning's discussions."

12.5 Conclusion

Peter glances over the notes he took during the meeting. "There are quite a few interesting ideas that came out of our discussion. What I found most valuable are the issues arising from applying the S-BPM approach to a new domain, one in which processes are executed by computational agents rather than people."

Peter looks at Diana. "From the point of view of an S-BPM practitioner, what is probably most intriguing about agents is the almost unrestricted design freedom in mapping subjects to agents, because agents can be specified in so many ways.

In traditional S-BPM applications all you have is a set of people, grouped together in hierarchies, networks or geographies. So the organisational units and structures here are relatively fixed. Now what happens when we have agents instead of people? We can create our agent world in almost any way we can think of. Some agents can be embodied in the physical environment, while others have only a virtual existence. The embodiment of agents can even change during the execution of a process, for example, from being completely virtual to becoming both virtual and physical—cyber-physical—as we've seen in the case of our product agent. Multiple agents may be combined to form a composite agent, so that the individual agents are just like the tools of a Swiss army knife used by that agent. And one agent may even spawn other agents as needed during the process. All this is not possible in human organisations—or at least it requires a level of abstraction that is usually not needed in traditional business applications of S-BPM. So with such an unlimited set of possibilities in conceptualising agents, establishing mappings between subjects and agents may be difficult, especially if you haven't worked with agent-based applications before."

Peter pauses for a while, then continues, "The other thing that struck me is that many things we take for granted in people-based process execution need to be carefully engineered in agent-based applications. This includes devising mechanisms for controlling which agents are to be allowed to execute a subject's behaviour. The mechanisms vary based on the specific requirements associated with different parts of the process. For many agent-based applications, a simple first-come-first-serve mechanism is not suited as there is typically the need for more constrained access control based on specific performance characteristics and the current state of the production process. This often requires endowing agents with heuristic knowledge and decision-making capacities, and the ability to reason about the goals of different processes and how they relate to one another."

"I really want to thank you, Peter, for initiating this interesting project and inviting me to be part of it," Diana says. "I learned a lot today, because our discussions about agent implementations made me think about S-BPM in a more general way. I believe that some of the concepts we developed, such as the control mechanisms for subject execution, may even be useful for some of the business process applications in my department."

"Great," Peter smiles, "I'm looking forward to working with all of you to make this project succeed. I think the S-BPM approach helps us develop our agent-based prototype fairly quickly. But now let's have a well-deserved lunch!"

Acknowledgments The research leading to these results has received funding from the European Union Seventh Framework Programme FP7-2013-NMP-ICT-FOF(RTD) under grant agreement no. 609190.