This chapter introduces the three basic approaches, namely deliberative, reactive and hybrid, to building intelligent agents. The limitations of individual intentions for joint problem solving are listed and the joint persistent goals formalism of Cohen and Levesque [48,101,102], for collaborative problem solving is explained. This formalism forms the foundation for the joint responsibility model [96], which we adopt for our agents.

Section 2.1 introduces the three types of agent architectures. Section 2.2 points out the role of intentions in an agent's reasoning process and explains the reasons for the unsuitability of individual intentions to joint problem solving. Joint persistent goals, which is a formulation developed for joint problem solving is described in Section 2.3. Section 2.4 explains the joint responsibility model. Finally, Section 2.5 gives the conclusions.

2.1 Agent architectures

Based on the kind of representation and reasoning used, agent architectures are classified as deliberative, reactive and hybrid [97].

2.1.1 Deliberative architectures

Deliberative architecture is defined to be one that contains an explicitly represented symbolic model of the world (comprising mental states like beliefs, desires and intentions - BDI) and in which decisions (for example about what actions to perform) are made via logical reasoning based on pattern matching and symbolic manipulation [3,6,94,126].
The reasoning process in a BDI agent is shown in Figure 2.1. As this Figure illustrates, there are seven main components to a BDI agent [47]

- A set of beliefs representing information the agent has about its current environment;

- A belief revision function, (brf), which takes a perceptual input and the agent's current beliefs, and on the basis of these, determines a new set of beliefs;

- An option generation function, (options), which determines the options available to the agent (its desires), on the basis of its current beliefs about its environment and its current intentions;

- A set of current options, representing possible courses of actions available to the agent;

- A filter function (filter), which represents the agent's deliberation process, and determines the agent's intentions on the basis of its current beliefs, desires, and intentions;

- A set of current intentions, representing the agent's current focus - goals it has committed to trying to bring about;

- An action selection function (execute), which determines an action to perform on the basis of current intentions

A state of a BDI agent at any given moment is given by a triple (B D I) where Bel, Des, and Int and which should always remain consistent.

An agent's belief revision function is a mapping from the current percept and current beliefs to a new set of beliefs. The option generation function, options maps a set of beliefs and a set of intentions to a set of desires. This function is responsible for the
Figure 2.1 Belief - Desire - Intention Architecture
agent's means ends reasoning - the process of deciding how to achieve intentions. Thus, once an agent has formed an intention to do $x$, it must subsequently consider options to achieve $x$. These options will be more concrete and less abstract than $x$. As some of these options become intentions themselves, they will also feed back into option generation, resulting in more concrete options being generated. The BDI agent's option generation process can therefore be thought of as one of recursively elaborating a hierarchical plan structure, considering progressively more specific intentions, until finally it reaches the intentions that correspond to executable actions. The options function should ensure that the options generated are consistent with both the agent's current beliefs and current intentions.

A BDI agent's deliberation process is represented in the filter function, which updates the agent's intentions on the basis of its previously held intentions and current beliefs and desires. This function does the following things: drops any intentions that are no longer achievable, retains intentions that are not achieved, and adopts new intentions.

The execute function returns executable intentions – ones that correspond to directly executable actions.

The BDI agent's action function is defined by the following pseudo code

```plaintext
function action (p: P)  //P: percept
begin
    B := brf(B, p)
    D := options(D, I)
    I := filter (B, D, I)
    Return execute (I)
end function action
```

The main advantages of BDI model are that firstly it is intuitive; it is similar to the way humans go through the process of deciding what to do and then how to do it, and it is easier to understand. Secondly, it gives a clear functional decomposition, which indicates what sorts of subsystems might be required to build an agent
However, providing an accurate and complete description and using it to reason in time are two main challenges of deliberative systems.

Planning agents for instance, fall under this category of deliberative systems. The best known early planning system was STRIPS [109]. The STRIPS planning algorithm was very simple, and proved to be ineffective on problems of even moderate complexity. Much effort was subsequently devoted to developing more effective techniques. Two major innovations were hierarchical and non-linear planning [34,35]. Another example is the Intelligent Resource Bounded Machine Architecture (IRMA), which is based on BDI model [82,83,84]

However, in real world things do not always go as planned. The assumptions of planning systems are that the environment is totally predictable and the internal model is totally complete and correct. These assumptions are often inappropriate - many environments are dynamic, ongoing, real-time and unpredictable To cope with this mismatch some researchers started investigating the idea of reactive systems [19,106].

2.1.2 Reactive Architectures

A reactive architecture is defined to be one that does not include any kind of central symbolic world model, and does not use complex symbolic reasoning [94] In such systems agents merely react to situations and do not reason about the world Usually, both, the agents and the actions are relatively simple and global properties are seen as emerging from the interaction of behaviors [113,114] The advantage of this approach is that because of their lack of explicit reasoning, agents are fast and can respond to changing environmental conditions so long as they have a predefined stimulusresponse pairing.

Brooks [104,105,106] proposed the sitbsumption architecture, which is the best-known reactive architecture and built some robots using it to demonstrate his claim that intelligence does not require explicit symbolic representation and reasoning
There are two important characteristics of the subsumption architecture. The first is that agent's decision making is realized through a set of task accomplishing behaviors. Each behavior is an individual action function, which continuously takes perceptual input and maps it into an action. Each behavior is intended to achieve some particular task. These behaviors are implemented as rules of the form

\[
\text{Situation} \rightarrow \text{action}
\]

which map perceptual input directly to actions without the use of any symbolic representations or reasoning.

The second characteristic of subsumption architecture is that many behaviors can fire simultaneously. In order to choose between different actions, Brooks proposed arranging the modules into a subsumption hierarchy, with the behaviors arranged into layers. Lower layers in the hierarchy are able to inhibit higher layers the lower the layer is, the higher its priority The higher layers represent more abstract behaviors. Similar kind of work has been reported by Steels, who described simulations of 'Mars Explorer' systems, containing a large number of subsumption architecture agents [81].

The advantages of reactive architectures are their simplicity, economy, and robustness against failure. But purely reactive architectures have some unsolved problems. For instance, no principled method exists for building such agents While effective agents can be generated with small numbers of behaviors, it is more difficult to build agents that contain many layers The interaction between the different behaviors becomes too complex to understand

Though deliberative and reactive approaches appear to be opposite to each other, intentions actually provide a link between them [96] Reactive agents merely exhibit a special type of intention Each individual has a number of fixed, simple intentions, which are specified by the system designer and are implicitly available to the agent
When the designer defines an agent's behavior he is in fact installing its intentions. These fixed intentions (or precompiled behaviors) are then invoked automatically whenever certain conditions prevail; there is no run-time means-end reasoning. This contrasts with deliberative (non-precompiled) intentions, which are subject to means-end analysis. Jennings thus takes the unifying perspective that deliberative and reactive systems are just opposite ends of a spectrum rather than fundamentally different technologies. Similar reasoning is carried out by both types of system, although at different stages of the development process - run time for reflective systems and design time for reactive ones.

2.1.3 Hybrid Architectures

In most applications neither a completely deliberative nor a completely reactive approach is suitable for building agents. In such cases hybrid systems, which attempt to combine deliberative and reactive approaches are used, as is done in the Procedural Reasoning System [85].

One approach to build a hybrid agent is to use a deliberative component, which develops the plans, and a reactive component capable of reacting to events in the external world. Often, the reactive component is given some kind of precedence over the deliberative one, so that it can provide a rapid response to important environmental events. This kind of structuring leads to the idea of a layered architecture, of which Touring Machines [52], INTERRAP [55,56], and CIRCA [54] are good examples. In such architectures, an agent's control subsystems are arranged into a hierarchy, with higher layers dealing with information at increasing levels of abstraction.

Generally, there is a minimum of two layers, to deal with the reactive and proactive behaviors respectively. Broadly speaking, two types of information and control flow within the layers can be identified [47]:

Chapter 2 Intelligent Agents Approaches

Figure 2.2 Horizontal layering

Figure 2.3 Vertical layering
• Horizontal layering

In horizontally layered architectures, the software layers are each directly connected to the sensory input and action output. This is shown in Figure 2.2 [47]. Each layer itself acts like an agent, producing suggestions as to what actions to perform.

• Vertical layering

In vertically layered architectures, see Figure 2.3 [47], sensory input and action output are each dealt with by at most one layer each.

The advantage of horizontal layering is its conceptual simplicity; if an agent is needed to exhibit \( n \) different types of behavior, then \( n \) different layers can be implemented. However, as the layers compete with one another to generate action suggestions, there is a danger that the overall behavior of the agent will not be coherent. To ensure consistency among layers some central control is required which makes decisions about which layer has control of the agent at any given time. But the use of central control creates a bottleneck to the agent's decision making.

This problem is partly overcome in the vertically layered architecture. These are subdivided into one pass and two pass architectures. In one-pass architectures, control flows sequentially through each layer, until the final layer generates action output. In two pass architectures, information flows up and control then flows back down. Both these architectures reduce the complexity of interaction between layers.

2.2 Intentions

From the above discussion it can be seen that intentions from an important component of intelligent agents. We therefore look at intentions in more detail. Intentions mentioned in Section 2.1 are individual intentions. These define individual
behavior and, as such, cannot be used for describing collaborative actions. There are two main reasons for this, [102]. Firstly, joint action is more than just the sum of individual actions, even if the actions happen to be coordinated (the coordination can be accidental). In order to represent this 'extra' part, formalisms and structures specifically related to collaborative activity are needed.

Secondly, there is a fundamental difference between individuals and groups. This can be shown by considering the notion of commitment. A group's commitment to an objective cannot be a version of individual commitment, where the team is taken to be an agent, because teams may diverge in their beliefs (this can only happen to schizophrenic individuals) [102]. If an individual comes to believe that a particular goal is unachievable, then it is rational for it to give it up. The agent can drop the goal because it knows enough to do so. Similarly, when an individual finds that the team's overall objective is impossible, the entire team must stop trying to achieve it. However, it is not necessary for the whole team to know enough to do so. Consider the example of a group of agents collaboratively trying to lift a table. One of the agents may observe that the table to be moved is nailed to the floor and therefore the group's objective cannot be attained. However, it cannot be assumed that all the other team members have also been able to make this observation and the corresponding deduction. Hence although there is no longer mutual belief that the goal is achievable, since one agent has seen the nails, there is not yet mutual belief that the goal is unachievable and therefore some parts of the team remain committed.

Joint action requires an objective the group wishes to achieve and a recognition that they wish to achieve it in a collaborative manner [96]. So in a cooperative lift, all team members must want to lift the table and they must want to do it as a team. The second component of this definition is important because it distinguishes between identical and parallel goals [107]. For instance if \( x \) and \( y \) have the goal to have cake baked, their goals are identical, but if they both have the goal to eat cake (i.e. \( x \) has the goal that \( x \) eats cake and \( y \) has the goal that \( y \) eats cake), they merely have
parallel goals. This distinction is important because the two relationships imply different consequences in social interaction. Parallel goals give rise to competition if resources are scarce, whereas identical goals result in cooperation and coordination. We focus on identical goals, which are also termed common, joint or collaborative goals.

There are two prerequisites for collaborative problem solving to take place. A group of agents can engage in collaborative problem solving firstly by recognizing a shared objective and making a commitment to achieve it. This shared objective binds the individual actions into a cohesive whole. However, having a common aim is not enough for attaining the objective in a collaborative manner. The next important prerequisite is that agents also need to agree upon a common solution and base their subsequent actions on it.

In order to develop a common solution, participating agents need to augment their individual intentions to comply with those of others. So if one agent wants to lift a table at time 10 and another at time 15, then one or both of them will need to modify their intentions in order to make the joint action possible. From the time a common solution is agreed upon till the completion of joint activity, all the agents in a team believe that the others are also doing their bit and adhering to the agreed solution. Thus team problem solving is characterized by the mental state of the participants. A joint activity is performed by individuals sharing certain specific mental properties. Several social action formulations like the theories of collective intentionality, shared plans, we intentions, social plans, group intentions, joint persistent goals, and joint responsibility model, have addressed various aspects of mental state. However, we adopt the joint responsibility model for agents in our framework because we are interested in dynamic and unpredictable environments and joint responsibility model ensures coordination among agents even when there are unanticipated changes in the environment. The joint responsibility model is based on the work of Cohen and
Levesque on joint persistent goals. We therefore look at joint persistent goals in the following section and at the joint responsibility model in Section 2.4.

2.3 Joint Persistent Goals

Cohen and Levesque [101,102] propose a definition of joint intentions which is

*Joint intention is a joint commitment to perform a collective act while in a certain shared mental state.*

The central notion of joint commitment is formalized through the definition of joint persistent goals, which in turn is based on the concept of *achievement goals*. Achievement goals define the state of individuals participating in a team which is working on a collective goal (like move table) with a specified motivation (because it is necessary to gain access to a cupboard). Agent $A$ has a *weak achievement goal*, relative to motivation $q$, to bring about $p$ if either of the following is true:

- $A$ does not yet believe that $p$ is true and has $p$ being eventually true as a goal (i.e. $A$ has a *normal achievement goal* to bring about $p$).

- $A$ believes that $p$ is true, will never be true or is irrelevant ($q$ is false), but has as a goal that the status of $p$ be mutually believed by all team members.

Thus a weak achievement goal involves four separate cases either $A$ has $p$ as a normal achievement goal (it wants the table to be moved); thinks that $p$ is true and wants to make this fact mutually believed (it believes the table has already been lifted); believes that $p$ will never be true (it believes that the table is nailed to the floor) and wants to make this fact mutually believed or. finally, believes there is no longer a need to gain access to the cupboard ($q$ is no longer true)

Weak achievement goals form the basis of the definition of *joint persistent goals* (JPGs). A team of agents has a JPG to achieve $p$, relative to $q$, if and only if
1. they mutually believe that \( p \) is currently false (the table has not been lifted);

2. they believe that they all want \( p \) to be eventually true (they all want the table to be lifted); and finally

3. until they come to mutually believe either that \( p \) is true, that \( p \) will never be true or that \( q \) is false, they will continue to mutually believe that they each have \( p \) as a weak achievement goal relative to \( q \).

Thus if a team is jointly committed to achieving \( p \), the agents mutually believe that they each have \( p \) as a normal achievement goal initially. However, as time passes, team members cannot rely on the fact that they still have \( p \) as a normal achievement goal; they can only assume that they have it as a weak achievement goal. The reason for this is that one team member may have discovered privately that the goal is finished (true, impossible, or irrelevant) and be in the process of making this fact known to its associates.

If at some point, it is no longer mutually believed that everybody still has the normal achievement goal, then there is no longer a JPG as not all the agents wish \( p \) to be true In this case the team is no longer committed to \( p \). However there is still mutual belief that a weak achievement goal will continue to persist which ensures that all team members are informed of the lack of commitment by an individual within the group.

A joint intention between agents \( x \) and \( y \) to achieve action \( a \) relative to motivation \( q \), is then defined as a joint commitment to the agents* having done a collective action, with the agents of the primitive events as team members and with the team acting in a joint mental state.

This formulation provides criteria with which team members can evaluate their ongoing problem solving activity. JPGs specify that agents must track a goal's validity and also provide the conditions, which must be monitored in order to
perform this task. In addition to the evaluation criteria the model also provides the causal link to behavior; when an agent comes to privately believe that the joint act is untenable, or successfully completed, it must ensure that all its fellow team members are made aware of this fact.

JPGs however are not sufficient for attaining joint action firstly because joint action requires that agents work within the context of a commonly agreed solution, as there can be no joint action without this. A comprehensive formulation must make reference to plan states as well as goal states and explicitly specify the need for a common solution [96]. Joint action requires both a joint goal and adherence to a common solution while attaining it. Plan states should therefore be an integral component of the formulation of joint intentions. A code of conduct, which specifies how team members should behave with respect to the plan states, is therefore necessary. Secondly, an important requirement for being a team member is the ability to contribute something towards the group's objective. The joint responsibility model subsumes and extends the work on joint persistent goals on these two fronts. Responsibility uses joint intentions as conduct controllers, specifying how both individuals and collectives should behave while engaged in collaborative problem solving. The model addresses the problem of ensuring that groups remain coordinated in the face of unanticipated changes in the environment.

2.4 Joint Responsibility

In single agent systems, the behavior law is that of rationality: an agent only selects those actions that will lead to the satisfaction of one of its goals. This kind of individual rationality is not sufficient for defining the behavior of participants engaged in cooperative problem solving. What is required is team rationality. The joint responsibility model proposed by Nick Jennings [96], for team rationality, has its roots in the theory of joint intentions of Cohen and Levesque [101, 102], which was described in the previous section.
Responsibility extends the work on joint intentions by defining a structure for joint commitment, which involves both plan and goal states. It provides prescriptions of behavior for agents engaged in collaborative problem solving. The responsibility model addresses two facets of collaborative action. It defines preconditions that must be satisfied before joint problem solving can commence and prescribes how team members should behave once it has started.

Before joint problem solving can commence, four conditions need to be satisfied [96]:

An agent must recognize the need for collaborative action to solve a particular problem. This agent must contact other agents, who it believes will be able to play an active problem-solving role, to see whether they wish to be involved in bringing about the collective aim. An important constraint when carrying out this task is to ensure that all members are able to contribute something to the group's efforts.

Once a group, which shares a common purpose, has been formed, members must agree that they wish to achieve the common objective in a collaborative manner.

In particular, this involves agreeing they need a common solution, which they will adhere to. Agents must agree for the duration of the joint action that they will follow a code of conduct with respect to their activity. Adherence to this code (the responsibility model) ensures that the group operates in a coordinated and efficient manner even if things do not go according to plan.

The second aspect of the responsibility model is applicable once the above conditions have been met and the joint action is initiated. An agent follows the agreed plan of action (honoring its commitments) while continuously monitoring its local activity and information received from other agents to ensure everything is progressing smoothly (tracking the rationality of its commitments).
As agents are situated in dynamic and unpredictable environments, events may occur which affect their commitment. For example, newly acquired information may invalidate previous assumptions or important events may require urgent attention and distract the agent from its agreed course of action. Therefore conditions for dropping commitments (both to the overall objective and the common solution for obtaining it) need to be enumerated and the agent needs to continuously monitor its activity to detect when these conditions occur. If such circumstances do arise, agents need to reassess their position. This may require rescheduling actions, or re-planning, or dropping the joint objective altogether if the motivation is no longer present.

When an individual becomes uncommitted, it cannot simply stop its own activity and ignore others. This is because the other group members also need not necessarily be able to detect the condition for dropping the commitment. Therefore the agent that realizes that a joint goal can no longer be achieved, must inform all its fellow team members of this fact and also the reason for its change of commitment. This ensures team members are kept informed of events, which affect their joint work, and so when things do go wrong, the amount of wasted resources can be minimized.

As we are interested in time constrained domains, reducing the amount of wasted effort is one of our major concerns. Hence the architecture we chose for our agents also uses intentions for representing collaborative problem solving. The agent architecture in TRACE is described in detail in Chapter 4.

2.5 Conclusions

This chapter introduced the three different types of agent architectures. Intentions are an integral component of intelligent agents. As individual intentions are inadequate for joint problem solving, some formulation that is specifically related to joint problem solving is necessary. The work on JPGs was described as it forms the foundation for the joint responsibility model that we adopt for our framework. We chose the responsibility model because our focus is on dynamic and unpredictable
environments, and the responsibility model ensures coordination among agents in such an environment.