Chapter 5

Actions of EM, PD and SD

The activities of EM, PD and SD are essentially sequences of situated actions performed on artefacts by modellers, designers and software developers. Chapter 3 argued that the character of actions and other aspects of activities is determined by artefacts. It follows that the nature of the artefacts identified in Chapter 4 can be expected to determine the nature of actions.

This chapter compares the actions of EM, PD and SD to identify how they are essentially different. A suitable framework for comparison is provided by the theory of creative cognition [FWS92] that characterizes processes as generative and exploratory. The results of examining the processes of the lift project with respect to each kind of action are given. The characterization of processes is extended to observation and experimentation in scientific inquiry [Kap64].

5.1 Definition

Actions, in this thesis, correspond to the mental processes associated with the Geneplore model of Finke *et al* described in their book entitled "Creative Cognition: Theory, Research, and Application" [FWS92]:

> The Geneplore model consists of two distinct processing components: a generative phase, followed by an exploratory phase (Figure 5.1). In the initial, generative phase, one constructs mental representations called preinventive structures, having various properties that promote creative

> discovery. These properties are then exploited during an exploratory phase in which one seeks to interpret the preinventive structures in meaningful ways. These preinventive structures can be thought of as internal precursors to the final, externalized creative products and would be generated, regenerated, and modified throughout the course of creative exploration.

Appendix D gives a review of the book by Finke *et al* entitled "Creative Cognition: Theory, Research and Applications" [FWS92] and a critical analysis of the Geneplore model it describes.

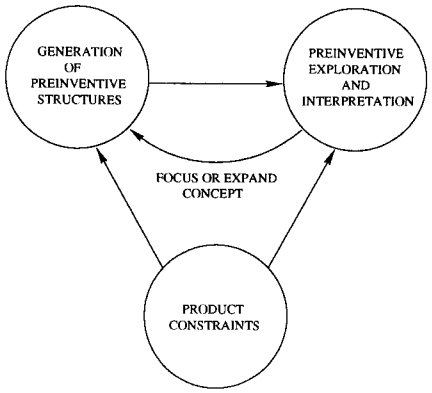


Figure 5.1: Geneplore Model.

This chapter investigates the application of generative and exploratory actions (Table 5.1) to artefacts with the properties that encourage analysis and creativity discussed in Chapter 4. Finke *et al* recognize that man's "cognitive capacity" is limited and that external support is needed: "Although we are treating these structures as internal representations, there is no reason that the structures could not be externalized at any point in the creative act ... this has the advantage that one could then deal with more complex structures but the disadvantage that it might limit the flexibility in modifying and transforming the structures" [FWS92]. This chapter investigates the advantages and disadvantages of externalization. This chapter also addresses Norman's claim that cognition can occur both in the head and in the world: "we should not see cognition as a purely unsupported activity" [Nor91].

Generative actions	Exploratory actions
Retrieval	Attribute finding
Association	Conceptual interpretation
Synthesis	Functional inference
Transformation	Contextual shifting
Transfer	Hypothesis testing
Reduction	Searching for limitations

Table 5.1: Generative and exploratory actions

5.2 Generative actions

This section examines the actions for generating artefacts that correspond to the six mental generative processes identified in [FWS92] (this book is reviewed in Appendix D) as being some of the most important for generating preinventive structures:

- the most basic processes consist of the **retrieval** of existing structures and the formation of **associations** among these structures. Typically these retrieval and associative processes happen quickly and automatically, but sometimes they are inhibited, resulting in mental blocks and fixation effects.

- a richer variety of structures results from the **synthesis** of component parts and by the **transformation** of the resulting forms. These processes usually yield more intricate creative possibilities than simple retrieval and association.

- **analogical transfer** is when a relationship or set of relationships in one context is transferred to another resulting in structures that are analogous to those that are already familiar. For example, early models of the structure of atoms resulted from analogical transfer of the relationships among the sun and planets in the solar system [FWS92].

- **categorical reduction** means mentally reducing objects or elements to more primitive categorical descriptions. For example, one might try to develop a better coffee cup not by considering it as a "cup" but as a container for keeping liquid hot and allowing it to be consumed [FWS92].

Finke *et al* [FWS92] recognize that these generative processes are not restricted to the generation of creative structures but are also used in the generation of structures for analysis: "Each of these generative processes has already been explored to some extent in traditional areas within cognitive psychology" [FWS92] with traditional cognitive psychology exploring the use of artefacts in essentially non- creative contexts. The use of generative actions by software developers in the lift project supports this view.

For convenience, the illustrative examples that go with this section have been organized into an appendix at the end of this chapter.

5.2.1 Retrieval

Constructing artefacts by retrieval was found to be a common technique in the lift project. By reusing parts of existing artefacts the modellers, designers and software developers were able to represent a subject more quickly than by synthesis. After establishing a primitive representation of the subject, by way of retrieval, the modellers, designers and software developers continued to refine the artefacts by using other generative actions.

Modellers began representing a subject by retrieving the artefacts describing the previous subject. This resulted in a continuity within EM artefacts, as shown in Example 5.1, with the artefacts of each subject being reused as the basis for constructing the artefacts of the next subject. For example, the LSD specification, visualization and animation of the MUL were reused by modellers as the starting point for constructing the Hydrolift artefacts.

The design of a subject began by retrieving sketches of the previous subject. As in EM, this resulted in a continuity within the sketches in the lift project. Sketches of subjects were reused as the basis for the sketching the next subject, as shown in Example 5.2. For example, the sketch of the MUL was used by designers as the starting point for sketching the Hydrolift.

Retrieval was used cautiously by software developers in the lift project. Inconsistencies were introduced into the structure, behaviour and process models by software developers reusing parts inappropriately. Retrieval was generally restricted

to a small number of parts corresponding to class definitions. These parts of the structure, behaviour and process model corresponded to parts of the statement of requirements that were similar for both subjects. For example, the software developer was able to reuse the MUL model of the brake in the Hydrolift models because the MUL and Hydrolift requirements for the brake were expressed by the same statement.

5.2.2 Association

Parts retrieved by modellers, designers and software developers were grouped to form new artefacts. Each part represented some part of the lift systems, such as a door, shaft or button. By associating these parts with one another they formed a representation of the subject as a whole.

EM artefact parts were associated without any explicit representation of how they were related. Relationships between parts were implied by the correspondence between the arrangement of parts in the subject and the arrangement of parts in the artefacts. For example, the LSD specification of the Hydrolift began as the combined MUL, pump, sonar and sensor LSD specifications, as shown in Example 5.4. Creative exploration of this incongruous association resulted in the eventual emergence of an LSD specification containing details about the pump, sonar and sensor agents. Sections of DoNaLD and ADM scripts were combined in a similar way to LSD specifications, without any explicit representation of how they were related, in order to construct visualizations and animations in the lift project, as shown in Example 5.4 .

Designers also used juxtaposing to associate artefact parts in the lift project. The designers arranged component caricatures into groups to represent the subject. For example, the sketch of the Hydrolift began as a representation of the MUL with a flooded shaft. Creative exploration of this incongruous association resulted in the emergence of the detailed Hydrolift sketch, as shown in Example 5.5.

In contrast to modellers and designers, software developers represented associations between artefact parts explicitly. For example, associations between classes in the structure model were represented as labeled arrows, as shown in Example

5.6. This avoided the problem of introducing creative properties into the models that might have resulted from simply arranging class definitions in the structure model without showing how they were related structurally and functionally. The software developers were able to transform the associations represented in the structure model into transitions and data-flows represented in the behaviour and process models by following the SD method of analysis, as shown in Example 5.6.

5.2.3 Synthesis

Artefact parts were created by the modellers, designers and software developers in the lift project whenever parts did not already exist. The SUL artefacts had to be synthesized because the SUL was the first subject in the lift project. The MUL and Hydrolift artefacts were constructed based on the SUL artefacts.

Synthesis of EM artefacts was found to follow a sequence in the lift project when the subject was novel. First the LSD specification was synthesized, shown in Example 5.7, followed by the synthesis of the visualization followed by the synthesis of the animation. The modeller represented his perception of the subject in an LSD specification then constructed the visualization and animation based on the description of observables and agents in the LSD specification. The modellers found it easier to start by creating an LSD description of the subject than to represent the structure and function of the subject in a visualization and animation directly. The LSD specification acted to guide the synthesis of the visualization and animation

The software developers had to verify synthesized models against the statement of requirements. The requirements were used to check synthesized parts of the structure, behaviour and process models, as shown in Example 5.8. The software developers preferred to transform the statement of requirements into models rather than synthesize and check models because it was more direct. Synthesized models typically needed a number of refinements before they satisfied the requirements, whereas models generated by transformation satisfied the requirements without any need for refinement or verification.

5.2.4 Transformation

There were opportunities in EM and SD for constructing artefacts by transforming existing artefacts. Since transformations were perhaps the quickest way of generating artefacts they were favoured by modellers and software developers where appropriate.

Transformation was found to be the most common means of generating SD artefacts in the lift project. The software developer transformed the structure of the statement of requirements into the structure, behaviour and process models by following the method of analysis, as shown in Example 5.9. Details from previously constructed models were used in the transformations in addition to the statement of requirements.

The modeller performed a straightforward transformation from the structure of the LSD specification into an ADM script to construct animations in the lift project, as shown in Example 5.10. Essentially, the transformation was done by renaming agents as entities, privileges as definitions and protocols as actions.

5.2.5 Analogical transfer

Although analogical transfer is normally associated with the relation between mental models [HT95], an externalized analogical transfer was observed in the lift project. Analogical transfer is when relations in one context are transferred to another in order to generate structures that are analogous to those that are already familiar [FWS92]. In order to avoid confusion I shall simply refer to the externalized form of analogical transfer as "transfer." Transfer in the lift project involved modellers, designers and software developers generating new artefacts by keeping the structure of existing artefacts and changing their content.

Modellers often retrieved artefacts to reuse their structure in the lift project. For example, the modeller retrieved the shaft agent definition and reused its structure to construct the pump in the Hydrolift, as shown in Example 5.11. Transfer was also used by modellers in order to construct visualizations and animations: the organization of DoNaLD and ADM scripts made it possible to redefine shapes, whilst keeping their relative positions the same, and make changes to the behaviour

of entities, whilst keeping their basic agency and protocols the same. In this way, the essential structure and function of lift systems, established in the early stages of the lift project with the SUL, was preserved throughout.

Designers in the lift project performed transfer between sketches. Transfer was achieved by reusing the layout of existing sketches. New component caricatures were added to complete the sketch after transfer, as shown in Example 5.12. The detail of the sketch was changed whilst preserving the layout that was established early in the lift project with the sketch of the SUL.

Transfer was seldom used for generating SD artefacts. The reason for this was that the meaning of SD models was determined more by the arrangement of symbols than by the symbols themselves. By transferring the structure of a model the greater part of its meaning was transferred with it, as shown in Example 5.13. Reusing the structure of artefacts had to be done with caution by software developers so as to avoid introducing inconsistencies and other creative properties into the models.

5.2.6 Categorical reduction

As with analogical transfer, categorical reduction is normally associated with mental structures. Categorical reduction means mentally reducing objects to more primitive descriptions of constituent parts by disregarding their more abstract higher-level conceptual structure [FWS92]. However, an externalized categorical reduction was observed during EM, PD and SD in the lift project. So as to avoid confusion I shall refer to the externalized categorical reduction simply as "reduction". Reduction in the lift project involved modellers and software developers generating new artefacts by stripping away the higher-level structure of existing artefacts.

Reduction was commonly used by modellers for generating artefacts in the lift project, as shown in Example 5.14. New artefacts were generated quickly by retrieval and association. Modelling continued by repeatedly reducing the artefacts into basic parts and rearranging the parts to form new artefacts. In this way, the modeller gradually converged upon a detailed representation of the subject. The structure of the EM artefacts was found to support the decomposition of artefacts into meaningful parts right down to the most basic element - the observable.

Reduction was found by software developers to be limited as a technique for SD. The importance of structure to the meaning of the models made it difficult to reduce SD models without rendering them meaningless, as shown in Example 5.15. Reduction was not performed on the structure, behaviour and process models because the parts, including states, transitions and actions, tended to have little meaning without the context of the symbols that surrounded them. Reduction was done by identifying a class definition then separating it from its context by representing its relationship to other classes as an interface.

5.3 Exploratory actions

This section examines the actions for exploring artefacts that correspond to the six mental generative processes identified in [FWS92] (this book is reviewed in Appendix D) as being some of the most important for creative exploration:

- **attribute finding** is the systematic search for emergent features in the structures.
- **conceptual interpretation** refers quite broadly to the process of taking a structure and finding an abstract, metaphorical, or theoretical interpretation of it. Conceptual interpretation can be thought of as the application of world knowledge or naive theories to the task of creative exploration.
- **functional inference** refers to the process of exploring the potential uses or functions of a structure. This process is often facilitated by imagining oneself actually trying to use the object in various ways.
- **contextual shifting** is considering a structure in new or different contexts as a way of gaining insights about other possible uses or meanings of the structure. This process often helps to overcome fixation effects and other obstacles to creative discovery.
- **hypothetical testing** is where one seeks to interpret the structures as representing possible solutions to a problem. A creative solution to a problem can often be found when more direct methods fail.

- **searching for limitations** is searching out what structures will not work and are not feasible. This is often just as important as actually discovering what will work.

The following examination focuses on the use of exploratory actions in EM and PD because there was little evidence of software developers using exploratory actions in the lift project.

For convenience, the illustrative examples that go with this section have been organized into an appendix at the end of this chapter.

5.3.1 Creative exploration and SD

There was little evidence of software developers using exploratory actions in the lift project. This was probably due to the analytical properties of the artefacts of SD discussed in Chapter 4:

- Familiarity
- Unambiguity
- Explicit meaning
- Completeness
- Consistency
- Convergence

These properties mean there is little incentive for the software developer to explore the artefacts, as shown in Example 5.16. There is no incentive because the artefacts make finding emergent features difficult and unnecessary to explore:

- the property of completeness means that all the information the software developer needs is within the artefacts so there is little incentive to search for emergent features;
- the SD method maps symbols from one domain to symbols in another so there is little incentive for the software developer to explore alternative interpretations of the subject;

- the functional meaning of the subject is explicit in the artefacts so there is little incentive for the software developer to explore alternative behaviours;
- the property of completeness means that the artefacts have the same formal meaning independent of context so there is little incentive for the software developer to shift the context of the artefacts either physically or conceptually;
- the property of convergence means that the SD process converges upon a solution so there is little incentive for the software developer to test artefacts;
- so long as the software developer follows the SD method the only limitations of the artefacts will be due to limitations in the statement of requirements which is not his responsibility.

SD in the lift project was essentially a methodical transformation of the statement of requirements into the structure, behaviour and process models. This suggests that SD is essentially a generative activity with little incentive for creative exploration.

5.3.2 Attribute finding

Modellers found emergent features in visualizations and animations. These features were not intentionally modelled and were only found by exploring the EM artefacts. The features discovered by modellers were included in subsequent visualizations and animations. Discoveries about observables and agency in the subject informed revisions to LSD specifications [BR94], as shown in Example 5.17.

Finding attributes in sketches was limited by the designers' knowledge of lift systems. The success of thought experiments in discovering emergent features in the sketches depended on the knowledge of the designers [Gre70]. The designers were not experienced in the lift project so they were unable to find many structural and functional attributes within the sketches. Experienced designers would have been expected to benefit far more from exploration based upon their mental models of lift systems.

5.3.3 Conceptual interpretation

Modellers and designers explored the subject by interpreting it in terms of abstract and concrete concepts. The modellers interpreted the SUL, MUL and Hydrolift in terms of the concepts of LSD, DoNaLD and ADM. The designers interpreted the SUL, MUL and Hydrolift in terms of the elements of a "graphical language" in the sense of Ferguson [Fer92].

Modellers interpreted the subject in terms of the LSD concepts of agent, protocol, derivate and observable. By interpreting the subject in terms of these concepts the modellers were able to represent and explore the SUL, MUL and Hydrolift in familiar terms that corresponded to elements that they perceived within the subject.

The modellers interpreted the LSD specification in terms of DoNaLD and ADM concepts in order to construct visualizations and animations in the lift project, as shown in Example 5.18. The concepts of these languages have precise and unambiguous meanings that define the structure and function of the visualizations and animations. Interpreting the LSD specification in terms of these concepts resulted in the emergence of features in the subject to do with the geometry of shapes and the synchronization of actions.

Designers interpreted the subject in terms of the elements of a "graphical language" . Although designers did not use a verbal language, they did use conventions for representing subjects in sketches. These conventions can be thought of as a kind of "graphical language" in the sense of Ferguson [Fer92]. The designers explored the spatial relations between components in the subject by arranging component caricatures in the sketch. The patterns of caricatures in a sketch can be thought of as "statements" that describe the subject in a graphical language of the designer, as shown in Example 5.19.

5.3.4 Functional inference

Modellers and designers used artefacts to explore the emergent behaviour of the subject. Norman has observed that mental models of systems are difficult to "run" [Nor91] so interactive artefacts play a particularly important supportive role in the

process of functional inference. The visualization and animation were found to be more suitable for functional inference than the LSD specification and sketch.

Visualizations and animations were found to be most appropriate for supporting functional inference in the lift project. Visualizations and animations provided the modeller with a context for interaction that mimicked that provided by the subject. For example, the MUL visualization and animation represented lift buttons by graphics that were sensitive to the point-and-click of a mouse and represented the state of the lift system by a picture for the modeller to see that reflected the current state of the lift system, as shown in Example 5.20. The ADM entities simulating LSD agents meant that the animation was more realistic than the visualization. However, by the modeller playing the roles of LSD agents, the visualization was found to give more freedom in inferring alternative functions of the subject.

Functional inference based on the LSD specification and sketch was found to be limited as a technique for exploration. Functional inference using LSD specifications and sketches involved thought experiments [Kap64] that depended on the knowledge and experience of lift systems. The modellers and designers in the lift project were not experienced in the workings of lift systems. Designers who have experience of lift system components are able to infer more complex behaviours from design sketches.

Modellers and designers explored the function of the subject by imagining themselves using the lift system. Modellers commonly represent themselves as agents in LSD specifications to help them imagine their interaction with a system, as shown in Example 5.20. This accords with the findings of Finke *et al* who state that "the process of functional inference is often facilitated by imagining oneself actually trying to use the object in various ways" [FWS92].

5.3.5 Contextual shifting

The artefacts of EM and PD were found to be context sensitive which meant that they were suitable for supporting creative discovery through contextual shifting. New features emerged in the artefact by placing it in different contexts. Modellers and designers incorporated the emergent features in subsequent artefacts.

Contextual shifting was used by modellers to explore EM artefacts. Modellers changed the context of the LSD specification, visualization and animation:

- *conceptual shifting* involved changing the context of an agent definition within an LSD specification, as shown in Example 5.21;
- *physical shifting* involved changing the physical environment of the computer running the visualization or animation.

Physical contextual shifting was limited in the lift project to changing the person interacting with the visualization or animation. In the future it is hoped that interfaces will be developed that allow components and systems to be linked to the computer so that they interact directly to change variables in the visualizations and animations. These physical devices could then replace their virtual representations in the form of ADM entities over a period of systems development.

Contextual shifting by designers had some similarities with conceptual context shifting by modellers in the lift project. Contextual shifting by designers involved changing the context of a component caricature within a sketch rather like changing the context of an agent definition within an LSD specification. This typically had the effect of changing the functional meaning of the component representation.

5.3.6 Hypothesis testing

The principal aim of the modellers and designers was to construct satisfactory representations of the subject. The artefacts were searched to find similarities that confirmed them as being appropriate representations of the subject. The more similarities the modellers and designers discovered the more confident they were that the artefacts were indeed satisfactory representations of the subject.

Modellers applied hypothesis testing to visualizations and animations. Testing of the LSD specification was less common because its direct correspondence to the subject as perceived by the modeller generally meant it was a satisfactory representation, albeit one that lacked structural and functional detail. The visualizations and animations were regularly tested by modellers to check their representation of

the subject's structure and function. For example, the visualization and animation were regularly tested for essential safety and liveness properties [MP92a]: the car must never move when the door is open [She96]; all user requests must be serviced within a respectable time period.

It was recognized during the lift project that there were alternative approaches to testing for safety and liveness properties than having the modeller observe the visualization and animation:

- Define an hypothesis testing ADM entity that would automatically check the constraints and produce a signal if they were violated. This idea raised questions about the reliability of the observations of such an agent.

- Use formal methods for verifying concurrent real-time systems [OG75, Bar85, Pnu86, Hoo91, AO91, MP92a] to analyze the ADM script. This idea raised the issue of how appropriate it was to attempt to formalize the behaviour of a visualization or animation.

These two approaches correspond to "watchdogs" and temporal logic as recommended for system verification by Harel [Har92].

Hypothesis testing in PD was found to be rather limited. Designers did not have anything equivalent to the visualization and animation with which to evaluate sketches. The designers lacked the experience needed to generate mental models of sufficient detail to test for safety and liveness properties.

5.3.7 Searching for limitations

Modellers and designers searched for inadequacies in artefacts. Instead of searching for examples of similarities between subjects and artefacts the modellers and designers purposely searched for counter-examples that showed mismatches between the subject and its representation. This was to counteract the natural tendency of the modellers and designers to find evidence that confirmed the artefacts as satisfactory. This phenomenon is known as "confirmation bias" [FWS92, Wol92].

Modellers searched for limitations in visualizations and animations by setting up scenarios that might potentially fail safety and liveness criteria. For example,

users were placed on the top and bottom floors in the MUL and Hydrolift animations in order to try and fail the liveness property that all user requests must be serviced within a respectable period of time. By setting up scenarios the modellers were able to search for limitations in the behaviour of visualizations and animations.

Searching for limitations was found to be a less effective technique in PD than in EM. Designers did not have anything equivalent to the visualization and animation with which to setup scenarios for determining the limitations of designs. Designers lacked the knowledge and experience needed to generate mental models of sufficient detail to test routine, let alone, exceptional lift system behaviour.

5.4 Further characterizations of actions

In this section, observation, experimentation, method and methodology, in the sense of Kaplan [Kap64], are discussed in terms of generative and exploratory actions.

5.4.1 Observation and experimentation

In [Kap64] Kaplan describes various kinds of experiments for the purposes of scientific enquiry:

- Methodological experiments serve to develop or improve a technique of scientific enquiry.
- Heuristic experiments are designed to generate novel ideas for further scientific enquiry and are of the form "What would happen if ..."
- Fact-finding experiments aim at determining some particular magnitude or property of a relatively familiar object or situation.
- Boundary experiments are fact-finding experiments to determine the extent of a theory or law.
- Simulation experiments are designed to learn what will happen in artificial conditions which directly correspond to real ones.

- Nomological experiments aim to establish a theory by confirming or disproving an hypothesis.
- Illustrative experiments do not add anything to the knowledge about something only to the knowledge of the audience.
- Thought experiments are those involving mental concepts as opposed to physical apparatus and of the form "Imagine what if ..."

There are clearly parallels to be drawn between these kinds of experiments and the exploratory acts discussed previously in this chapter:

- The purpose of an heuristic experiment is to interpret an object or situation in a creative way similar to the act of creative exploration.
- Fact-finding experiments correspond to exploring an artefact to discover emergent features in creative exploration.
- Models are used in simulation experiments in the same way models are used to explore their function and shift contexts in creative exploration.
- Nomological experiments correspond to the creative process of hypothetical testing.
- Thought experiments correspond to the mental processes described by Finke *et al* [FWS92] whereas the experiments using physical apparatus correspond to the actions described previously in this chapter.

This suggests that the scientific knowledge generated through experimentation is the result of creativity as well as analysis.

Although experimentation might be creative it could be argued that observation is a passive non-creative activity resulting in the recording of facts during an experiment. However, Kaplan [Kap64] counters this argument with the following statement:

> Basically, experimentation is a process of observation, to be carried out in a situation especially brought about for that purpose ... No scientific

> observation is wholly passive; how much the scientist intervenes before or during the process of observation is a matter of degree. Correspondingly, there is no sharp distinction between observation and experiment, only a series of gradations and intermediates.

This would suggest that both observation and experiment involve creativity to some degree resulting in certain artificiality in scientific knowledge.

This theme of perceived reality being to some extent the products of the experimenter's creativity is continued by Gooding in [Goo90] in his discussion on construals (see Section 4.4.1):

> "Making sense" involves achieving stable interaction with a bit of the world. If a construal succeeds in this, then it will be accepted provisionally as a model of the phenomenon ... The effectiveness of a construal emerges as it is vindicated in the outcomes of further exploratory and communicative behaviour. After a while it becomes "easy to see" phenomena in terms of it, and it paves the way for the "self-evidence."

It is clear from the above statement that he views observation as an activity which is combined with the creation, by the experimenter, of the construal. The same sentiment is shown by others writing about science [Bro86].

5.4.2 Methods and methodology

In [Kap64] Kaplan defines a method as a general technique in science:

- Forming concepts and hypotheses.
- Making observations and measurements.
- Performing experiments.
- Building models and theories.
- Providing explanations.
- Making predictions.

There are clearly parallels between these examples and the generative and exploratory actions discussed previously in this chapter. This suggests that the actions in this chapter can be thought of as methods in the sense of Kaplan [Kap64].

Kaplan [Kap64] is careful to distinguish between the terms method and methodology - two terms that are often confused especially in the area of SD [You92]. He defines methodology as

> the study - the description, the explanation, and the justification - of methods, and not the methods themselves ... The aim of methodology, then, is to describe and analyze these methods, throwing light on their limitations and resources, relating their potentialities to the twilight zone at the frontiers of knowledge. It is to venture generalizations from the success of particular techniques, suggesting new applications, and to unfold the specific bearings of logical and metaphysical principles on concrete problems, suggesting new formulations. It is to invite speculation from science and practicality from philosophy. In sum, the aim of methodology is to help us to *understand*, in the broadest possible terms, not the products of scientific inquiry but the process itself. [Kap64]

In this sense, the examination of generative and exploratory actions in this chapter may be construed as a methodology. However, it should be noted that the purpose of this examination is to gain a better understanding of the activities of EM, PD and SD not to improve upon them. Kaplan [Kap64] warns against attempting to refine methods through retrospective reconstruction, arguing that "pressing methodological norms too far we may inhibit bold and imaginative adventures of ideas." Perhaps, a contributing factor to the analytical nature of SD methods, such as the Shlaer-Mellor object-oriented method of analysis and design [SM88, SM92], is due to an emphasis on methodology in an effort to solve the software crisis.

5.5 Summary

In this chapter the actions of EM, PD and SD were compared. It was found that actions of both modellers and designers in the lift project consisted of generative

and exploratory actions:

- retrieval of artefacts from archives and association through combination;
- synthesis by essentially reassembling and rearranging artefact parts;
- transfer of high-level structure between artefacts;
- reduction of artefacts into constituent meaningful parts.

It was found in the lift project that SD had similar generative actions but that they were more constrained and transformational than in EM and PD. There was less incentive for the software developer to explore the resulting artefacts. Exploratory actions were found to be important to the construction of artefacts in EM and PD:

- searching for emergent features in the artefact;
- interpreting the artefact in terms of abstract concepts;
- inferring the function of the artefact;
- shifting the context of the artefacts;
- testing the artefact as a solution to a problem;
- searching for limitations in artefacts.

These exploratory actions were found to be done in parallel with exploring the subject.

This chapter also shows similarities between experimental approaches in science [Kap64] and EM, PD and SD in the lift project. The discussion of methodology in science [Kap64] gives insight into the use of methodical approaches in SD.

Appendix: Illustrative examples for Sections 5.2 and 5.3

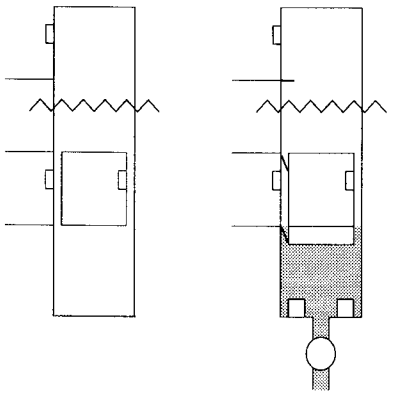
Example 5.1. Retrieval in EM. The outline of the MUL LSD specification (shown on the left) and the outline of the Hydrolift LSD specification (shown on the right) show that retrieval led to a continuity in the basic structure of the LSD specifications.

```
agent door() {
state
  door
oracle
  brake
}

agent landing(_F) {
state
  landButton
oracle
  floor direction brake
handle
  brake destination
}

agent car(_F) {
state
  carButton
oracle
  floor direction brake
handle
  brake destination
}

agent shaft() {
state
  floor destination direction
oracle
  brake
handle
  brake
}
```

```
agent door() {
state
  door
oracle
  brake
}

agent landing(_F) {
state
  landButton
oracle
  sensed brake
handle
  brake
}

agent car(_F) {
state
  carButton
oracle
  chan2
handle
  chan1
}

agent pump() {
state
  change target
oracle
  brake pressure chan1
handle
  brake pressure chan2
}
```

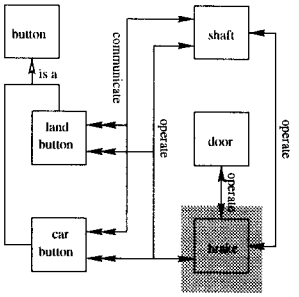
The change in agent name, from shaft to pump, and the changes in observables reflects the refinement of the Hydrolift specification following the initial generation by retrieval.

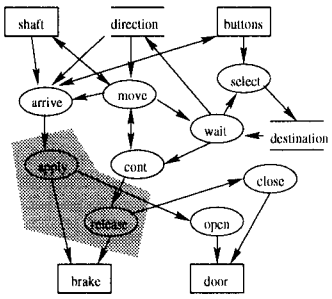
Example 5.2. Retrieval in PD. The sketches of the MUL and Hydrolift show that retrieval led to the continuity of the basic structure in lift systems during PD in the lift project.

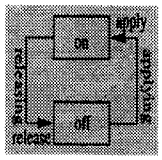


Additional component representations were added during the refinement of the Hydrolift following its initial generation by retrieval.

Example 5.3. Retrieval in SD. The MUL artefact parts corresponding to the brake Object class (shown highlighted)

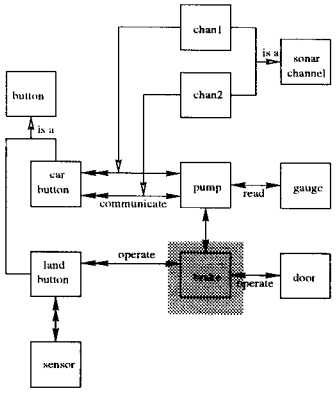


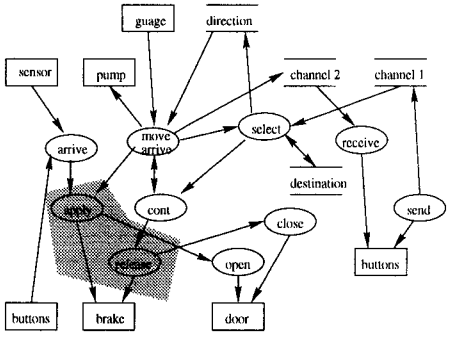


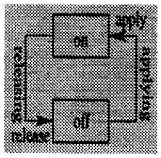



```
apply is
        apply brake;
        generate opening.

release is
        generate closing;
        release brake
```

were retrieved and reused by software developers in the generation of the Hydrolift structure, behaviour and process models








```
apply is
        apply brake;
        generate opening.

release is
        generate closing;
        release brake
```

As can be seen from the models, the Hydrolift brake is related to the car button and door the same as with the MUL brake. Also, the relation between the MUL brake and shaft is the same as the relation between the Hydrolift brake and pump.

Example 5.4. Association in EM. The association between the Hydrolift pump, sonar and sensor was represented by juxtaposing their agent definitions

```
agent pump() {
state
  change target
oracle
  pressure chan1
handle
  pressure chan2
}
```

```
agent sonar() {
oracle
  chan1
handle
  chan2
}
```

```
agent sensor()
state
  floor
oracle
  direction
handle
  sensed
}
```

within the Hydrolift LSD specification. The LSD specification shows that the modeller imagines the associations to be three distinct kinds of agent within the Hydrolift that share certain observables. The LSD specification does not describe the detailed structure or function of components.

Within the framework of the ADM, the entities corresponding to the above agents, such as the pump entity

```
entity pump() {
definition
  k = 100,
  change is (pressure < target) ? k :
            (pressure > target) ? -k : 0
action
  target == pressure + change && brake == OFF -> brake = ON,
  change == 0 -> target = chan1*k,
  pressure == target -> chan2 = target/k,
  brake == OFF -> pressure = pressure + change,
  brake == ON && change != 0 -> brake = OFF
}
```

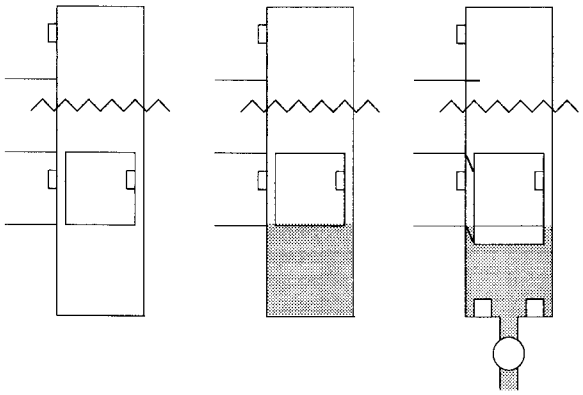
are associated by mechanisms for the instantiation of entities and the evaluation of variables. The associations between entities within the ADM are formalizable. A major part of transforming an LSD specification into scripts is interpreting the associations between LSD agents in terms of structure and function.

Most EM projects have involved the modellers associating agents by juxtaposing definitions in an LSD specification and then transforming the LSD specification into a script defining structural and functional relations. For example, the classroom simulation project can be thought of as a two-tier process:

- associate pupils and teachers by describing them in an LSD specification with shared observables;
- formalize the associations between pupils and teachers by transforming the LSD specification into scripts, such as the decision function into ADM.

A similar two-tier process was observed in other EM projects, including the VCCS, OXO and SBS.

Example 5.5. Association in PD. During the early stages of PD in the lift project the designer associated components by juxtaposing representations of them within a sketch. This was rather similar to the juxtaposing of agents within an LSD specification during EM. Subsequent stages of PD involved the designer exploring these associations. For example, the following three sketches

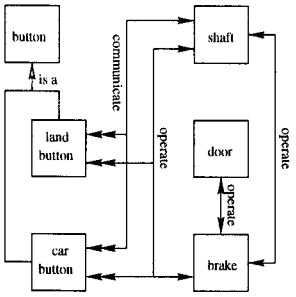


show the three steps in designing the Hydrolift:

1. the designer retrieved the sketch for the MUL;
2. the designer formed an incongruous association between a conventional lift system and water;
3. during the exploration of the association the designer added devices including a pump and sonar device.

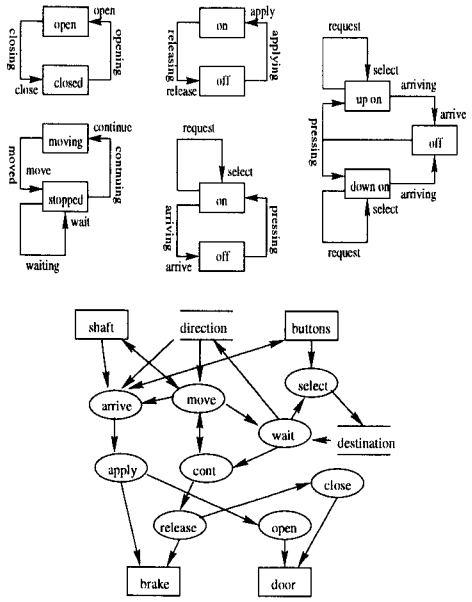
The design process was largely motivated by the designer's desire to progress from an incongruous sketch associating a conventional lift system with water to a congruous sketch of a Hydrolift.

Example 5.6. Association in SD. The association between buttons, shaft, door and break is shown in the MUL structure model



by the explicit representation of relations by directed arrows. The white arrow represents a structural relation and the black arrows represent functional relations between Object class instances. There is not necessarily any correspondence between the juxtaposing of Object class definitions and the juxtaposing of lift system components.

The functional relations represented in the structure model map onto relations between states and functions in the MUL behaviour and process models.



For example, the operate relations between the brake, shaft and buttons in the structure model correspond to the releasing and applying transitions in the behaviour model of the brake and the directed arrows feeding into the apply and release functions in the process model.

Example 5.7. Synthesis in EM. Synthesis of an LSD agent definition, such as the Hydrolift pump agent definition

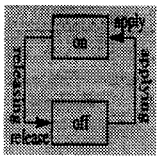
```
agent pump() {
state
  change target
oracle
  brake pressure chan1
handle
  brake pressure chan2
derivate
  k = 100,
  change is (pressure < target) ? k :
            (pressure > target) ? -k : 0
protocol
  target == pressure + change && brake == OFF -> brake = ON,
  change == 0 -> target = chan1*k,
  pressure == target -> chan2 = target/k,
  brake == OFF -> pressure = pressure + change,
  brake == ON && change != 0 -> brake = OFF
}
```

involved identifying

- the observable features associated with the pump in the subject represented as state, oracle and handle declarations,
- the synchronization between observables associated with the pump in the subject represented as derivates, and
- the causal relation between observables associated with the pump in the subject represented as protocol definitions.

Construction of the visualization and animation was found, in general, to follow synthesis of the LSD specification when the subject was novel.

Example 5.8. Synthesis in SD. After synthesizing the behaviour model for the brake, as shown below, based on the software developer's notion of its behaviour in terms of transitions between on and off states



he checked it against the requirements for the brake

> ... The shaft mechanism moves the car towards a destination landing stopping whenever the brake is applied. The brake is applied whenever the car arrives at a landing requested by a user ... The shaft mechanism releases the brake and starts the car moving again. For safety the door is opened and closed by the brake ensuring that the door is only open whilst the brake is on.

to verify the model was consistent with the rest of the MUL SD artefacts.

Example 5.9. Transformation in SD. The software developer followed a set method for transforming the requirements for the brake

> ... The shaft mechanism moves the car towards a destination landing stopping whenever the brake is applied. The brake is applied whenever the car arrives at a landing requested by a user ... The shaft mechanism releases the brake and starts the car moving again. For safety the door is opened and closed by the brake ensuring that the door is only open whilst the brake is on.

into the artefact parts associated with the brake shown in Example 5.3. The resulting mappings were as follows:

- The nouns "brake" and "shaft" were transformed into object Class representations in the structure model.
- The verb phrase "the door is opened and closed by the brake" was transformed into a functional association between the brake and door Object classes in the structure model.
- The verb phrase "The shaft mechanism moves the car towards a destination landing stopping whenever the brake is applied" was transformed into a functional association between the brake and shaft Object classes in the structure model.
- The phrase "The brake is applied whenever the car arrives at a landing requested by a user" was transformed into a functional association between the brake and lift button Object classes in the structure model.
- The verbs "applied" and "released" were transformed into the actions and transitions of the brake Object class represented in the behaviour model.

Essentially, the mapping was from nouns to Object classes and from verbs and verb phrases to actions and functional associations.

Example 5.10. Transformation in EM. The transformation from the definition of the shaft LSD agent

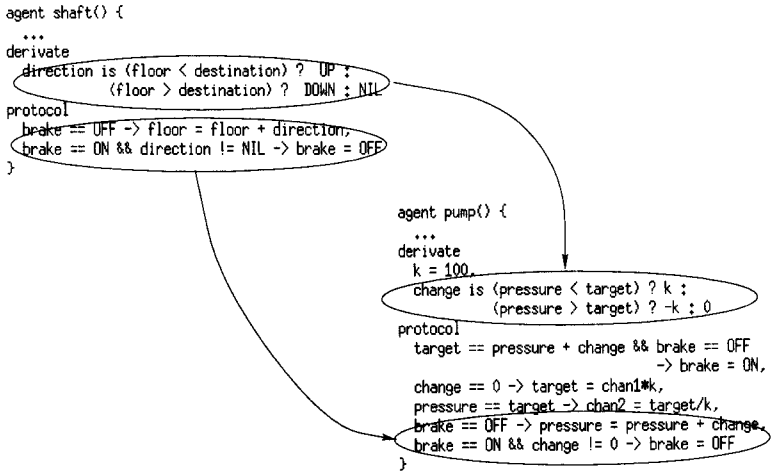
```
agent shaft() {
state
  floor destination direction
oracle
  brake
handle
  brake
derivate
  direction is (floor < destination) ?  UP :
               (floor > destination) ?  DOWN : NIL
protocol
  brake == OFF -> floor = floor + direction,
  brake == ON && direction != NIL -> brake = OFF
}
```

to the definition of the shaft ADM entity

```
entity shaft() {
definition
  direction is (floor < destination) ?  UP :
               (floor > destination) ?  DOWN : NIL
action
  brake == OFF -> floor = floor + direction,
  brake == ON && direction != NIL -> brake = OFF
}
```

is straightforward. However, although the surface-structure of the two definitions are very similar, their meaning is fundamentally different with the LSD representing a system in the world and the ADM definition representing a process in the computer. The difference between LSD and ADM is made clear in the introduction to EM in Chapter 2.

Example 5.11. Transfer in EM. The similarities between the structure of the LSD shaft and pump agent definitions is evidence of the analogical-like transfer between them in the lift project



The modeller kept the essential higher-level structure of the shaft definition while changing the observable names and adding some new definitions. In this way, the modeller preserved the notions of agency, causality and state that were represented in the LSD specification of the shaft agent.

In principle, a primitive LSD specification could have been generated to begin the EM railway project by changing the names of observables and agents in the MUL LSD specification:

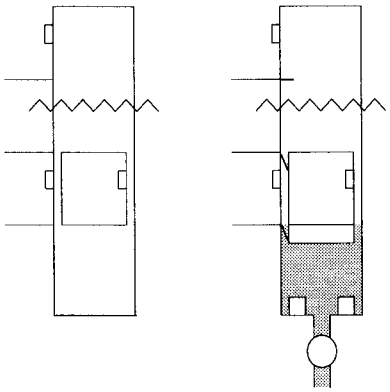
landing	→	**driver1**
car	→	**carriage**
shaft	→	**driver2**
floor	→	**station**
landButton	→	**schedStop**
carButton	→	**unschedStop**
UP	→	**NORTH**
DOWN	→	**SOUTH**

This renaming transforms the MUL landing agent protocol into the protocol for a train driver

```
schedStop{_S} == NORTH && _S == station + 1 && brake == OFF -> brake = ON,
schedStop{_S} == SOUTH && _S == station - 1 && brake == OFF -> brake = ON,
schedStop{_S} != OFF && direction == NIL -> destination = _S,
station == _S -> schedStop{_S} = OFF
```

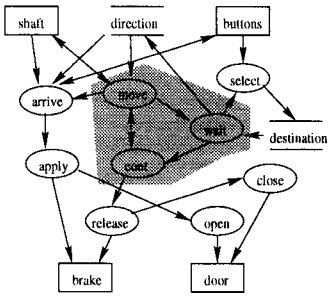
From this new protocol emerged novel timing characteristics. For example, whereas the observable `landButton` could change at any time the observable `schedStop` would only change during timetabling.

Example 5.12. Transfer in PD. The similarities between the sketches of the MUL and Hydrolift

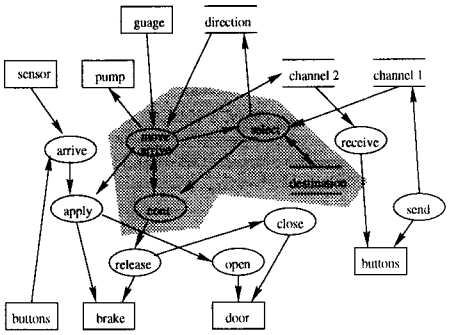


is evidence of the analogical-like transfer that took place during the sketching of the Hydrolift. The designer kept the essential higher-level structure of the MUL sketch while adding new component representations.

Example 5.13. Transfer in SD. There are some quite fundamental differences between the structure of the process model of the shaft and the structure of the process model of the pump indicating that analogical-like transfer would have been of little use in generating the process model of the pump.



Shaft process model.



Pump process model.

The meaning of the process models are denoted more by its higher-level structure. Thus, a change in subject typically requires a major change in the structure of the process model.

Example 5.14. Reduction in EM. The LSD shaft agent definition

```
agent shaft() {
state
  floor destination direction
oracle
  brake
handle
  brake
derivate
  direction is (floor < destination) ?  UP :
               (floor > destination) ?  DOWN : NIL
protocol
  brake == OFF -> floor = floor + direction,
  brake == ON && direction != NIL -> brake = OFF
}
```

can be reduced to a derivate and protocol definitions by removing the structurally higher-level agent construct

```
direction is (floor < destination) ?  UP :
             (floor > destination) ?  DOWN : NIL

brake == OFF -> floor = floor + direction,
brake == ON && direction != NIL -> brake = OFF
```

The protocol definition can be further reduced to statements representing the observations that cue causal change

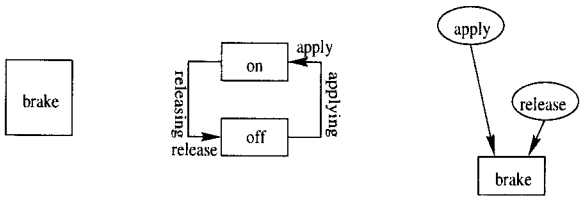
```
brake == OFF                    brake == ON && direction != NIL
```

and the changes themselves

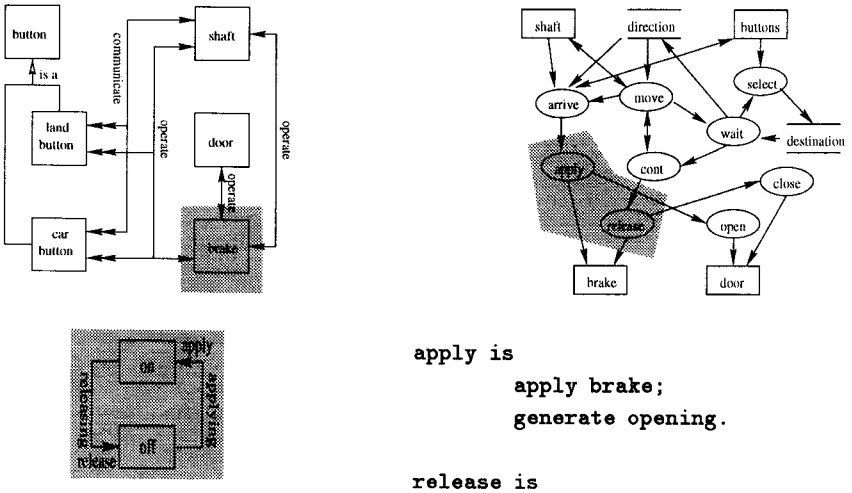
```
floor = floor + direction                    brake = OFF
```

These definitions can be further reduced to observables. The products of each of these reductions arguably preserve the creative property of implicit meaningfulness. Even the most primitive of elements, such as the observable `floor`, has an implicit meaning. Once a definition has been reduced into basic elements those elements are typically reused in the synthesis of new LSD definitions.

Example 5.15. Reduction in SD. It was found inappropriate to reduce structure, behaviour and process models down to basic elements, as shown below for the brake Object class, because most of the meaning of the models was denoted by their structure. So, reducing the structure of these models tended to "reduce" their meaning.



One solution was to keep the whole structure and highlight those parts associated with a particular Object class, as shown below for the MUL brake Object class.



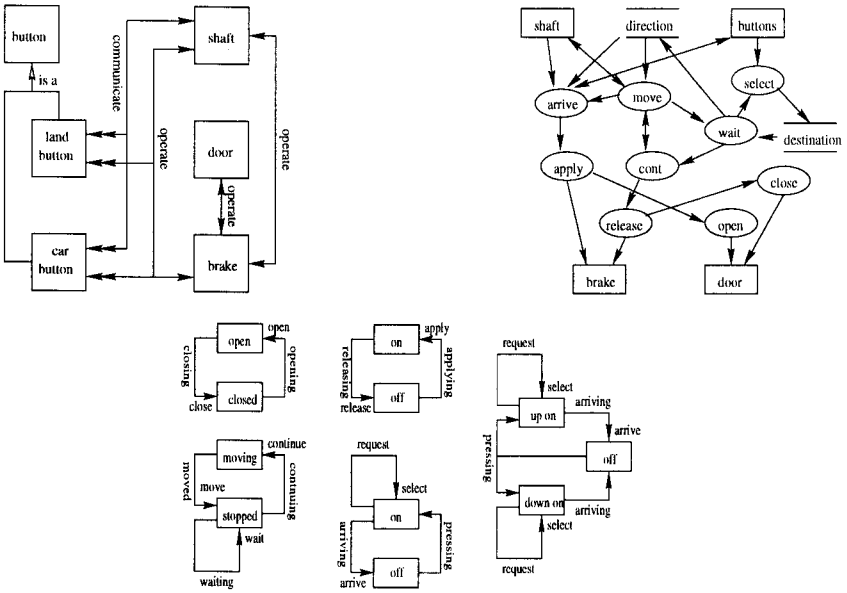

```
apply is
        apply brake;
        generate opening.

release is
        generate closing;
        release brake
```

Another solution is was to define an interface that separated the models into the form of the Object class and its context. For example, the Eiffel-like [Mey88] definition of the door and brake Object classes

```
class DOOR open close
  open is {} body {status' = OPEN}
  close is {} body {status' = CLOSED}
  inv is {status = OPEN and brake.status = ON or status = CLOSED}
end

class BRAKE apply release
  apply is {} body {status' = ON}
  release is {} body {status' = OFF}
end
```

is typical of the way interfaces of Object classes are defined textually. Textual representation suits the abstract nature of the interface.

Example 5.16. Lack of incentive for exploration in SD. The MUL structure, behaviour and process models



are typical of the artefacts constructed during SD in the lift project. These artefacts (along with the MUL statement of requirements and action definitions not shown here but in Appendix C) have the analytical properties of familiarity, unambiguity, explicit meaning, completeness, consistency and convergence. These properties mean there is little incentive to creatively explore the artefacts. Take the definition of the brake as an example:

- the structure and function of the brake is defined explicitly;
- the symbolic language underlying the models of the brake is formal;
- the structural and functional meaning of the brake is unsituated;
- the SD method prescribes the generation of the brake model.

The structure, behaviour and process models of the brake, as well as the models of the other lift components, were generated in the lift project by the software developer transforming the statement of requirements into the artefacts of SD. The lack of incentive for exploration of the resulting artefacts meant that SD was an essentially generative activity.

Example 5.17. Attribute finding in EM. Searches around incongruous parts of artefacts typically results in the discovery of emergent features. One source of conflict within an EM model is the absence of oracle-handle pairs. An oracle-handle pair indicates a link between agents. In the MUL the oracles and handles of the car and shaft agents are

```
floor direction brake destination
```

The modelling of the Hydrolift involved defining the pump, sonar and sensor agents with the following oracles and handles

```
pressure chan1 chan2 direction sensed
```

Each set of observables belong to different domains. The MUL set are high-level concepts associated with use whereas the Hydrolift set are detailed concepts associated with engineering. The need to link the car, shaft, pump, sonar and sensor in the Hydrolift resulted in the creative exploration of alternative interpretations of the MUL observables:

- the floor was interpreted as the pressure of the column of liquid at the base of the shaft;
- the direction was interpreted as the signal from a direction sensor;
- the destination was interpreted as a target pressure.

These emergent features were attributed to the new agents, such as the pump agent

```
agent pump() {
state
  change target
oracle
  brake pressure chan1
handle
  brake pressure chan2
derivate
  k = 100,
  change is (pressure < target) ? k :
            (pressure > target) ? -k : 0
protocol
  target == pressure + change && brake == OFF -> brake = ON,
  change == 0 -> target = chan1*k,
  pressure == target -> chan2 = target/k,
  brake == OFF -> pressure = pressure + change,
  brake == ON && change != 0 -> brake = OFF
}
```

that shows the observables pressure and target pressure instead of floor and destination.

Example 5.18. Conceptual interpretation in EM. Modellers in the lift project interpreted the subjects in terms of the concepts of LSD, DoNaLD and ADM:

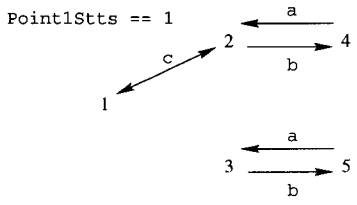
- LSD concepts correspond to common-sense notions, such as agency and causality, which help modellers describe unfamiliar objects and systems.
- DoNaLD concepts have formal meanings defining geometric shapes in a two-dimensional space that allow modellers to create visualizations on the computer and reason about the structure of objects and systems.
- ADM concepts have formal meanings defining processes that allow modellers to create animations on a computer and reason about the behaviour and functionality of objects and systems.

There are other languages in EM that help modellers conceptualize other aspects of the world that were not needed in the lift project, ARCA [Bey86a] for example. Providing a variety of languages, so that the world can be described in different ways and from different perspectives by the modeller, is an important principle in EM [BRS+89].

The railway project shows the use of conceptual interpretation to good effect. The connectivity of railway tracks was rather creatively interpreted as Cayley diagrams represented in the ARCA notation. For example, railway points are defined by the ARCA script

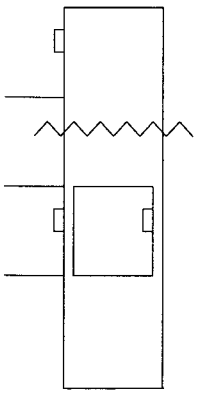
```
a_Point1{4} = 2
b_Point1{2} = 4
a_Point1{5} = 3
b_Point1{3} = 5
c_Point1{1} = if (Point1Stts == 1) 2 else 3
c_Point1{2} = if (Point1Stts == 1) 1 else 2
c_Point1{3} = if (Point1Stts == 1) 3 else 1
```

where `Point1` is the diagram representing the railway point, the lower-case letters denote colours and the numbers denote vertices. The resulting diagram



reflects the connectivity of a railway point that allows trains to switch between railway circuits in both directions.

Example 5.19. Interpretation in PD.
Although the sketch of the MUL, shown below, appears realistic the designer has clearly used conventions for representation, in other words, a kind of "graphical language" [Fer92]. The "vocabulary" and "grammar" of the language have to be known to fully understand the sketch.



Because lift systems have been around for so long designers have a standard repertoire of component "words" with which to interpret systems, as shown in Figure 5.2.

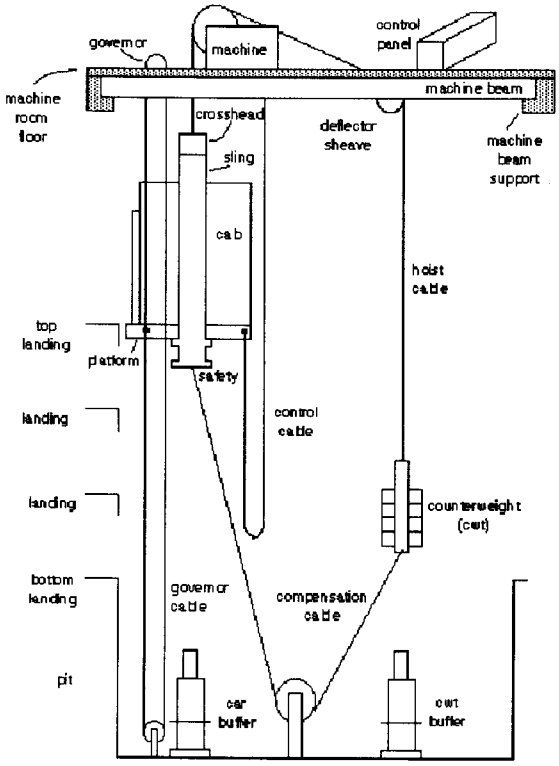
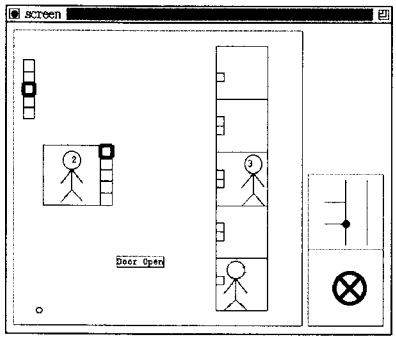


Figure 5.2: Lift system components.

This diagram is taken from [Yos92] which precisely defines each component in terms of its function and structure. In this way lift design becomes a process of fitting together components to give an intended function rather like constructing sentences is a process of fitting words together to give an intended meaning.

Example 5.20. Functional inference in EM. The screen-shot of the Hydrolift visualization and animation



shows a number of features that facilitated functional inference in the lift project:

- mouse point-and-click sensitive buttons corresponding to actual lift buttons;
- updated visual corresponding to the current state of the Hydrolift system;
- stick-men corresponding to lift users, such as the modeller.

These features of the visualization and animation, combined with the facility to enter redefinitions in a dialogue box, provided the modeller with an environment in which to infer the function of the Hydrolift.

By defining the LSD user agent, outlined below, the modeller was effectively modelling himself.

```
agent user(_U) {
  role In {
  state
    floor{_U}
  oracle
    door floor
  handle
    carButton
  }
  role Out {
  state
    floor{_U}
  oracle
    door floor
  handle
    landButton
}
```

It was found that having a representation of oneself helped functional inference in the lift project and other EM projects, including the SBS (helmsman agent) and OXO (player agent).

Example 5.21. Contextual shifting in EM. Perhaps the most significant conceptual contextual shift in the lift project was when the LSD definitions for the pump, sonar and sensor

```
agent pump() {
state
  change target
oracle
  pressure chan1
handle
  pressure chan2
}
```

```
agent sonar() {
oracle
  chan1
handle
  chan2
}
```

```
agent sensor()
state
  floor
oracle
  direction
handle
  sensed
}
```

where added to the MUL LSD specification of a conventional lift system. This shift resulted in the emergence of details about the pump, sonar and sensor that were subsequently incorporated into the Hydrolift LSD specification, visualization and animation.

In the lift project the physical contextual shifts were limited to changing the person interacting with the visualization or animation. This resulted in an objective evaluation of the Hydrolift design:

- the evaluation of the Hydrolift from multiple perspectives of various designers;
- the evaluation of the Hydrolift from multiple perspectives of various users.

Ideally the interaction between computer and it environment would not be restricted to human interaction. In the future it is hoped that the computer can play a more situated role by interacting with actual components.

The OXO project shows the use of conceptual contextual shifting in EM to good effect. Two observations of the game of OXO were modelled:

- the observation of the rules based on the conventional 3-by-3 matrix;
- the interpretation of the board in terms of the conventional 3-by-3 matrix.

The consequence of this approach was that the model of the rules was generic for all games of OXO whilst the model of the interpretation of the board changed for different geometries of the board. Different scripts represented the different player interpretations:

- a conventional board (`geomoxo.e` and `oxo.geom`);
- a three-dimensional board (`geomoxo4.e` and `oxo4.geom`);
- a board over a classical finite projective plane (`geompp7.e` and `pp7.geom`).

Each of these geometries corresponded to a different context for the player.