|  |  |  |  |
| --- | --- | --- | --- |
|  | **Generic Operations with Explicit Dispatch** | **Data-Directed Style** | **Message-Passing Style** |
| Add New Types (as in polar, rectangular etc. in the case of complex numbers) | 1. Implement all the supported operations including the constructors under the new representation and tag the data with the new type. Make sure that these operation names do not conflict with any of the existing representations 2. Add predicate for the new type (just like rectangular?) 3. Modify every generic selector to test for the new type (in addition to the existing types) and call the newly implemented operation that handles the new type | Implement an ‘install’ package for the new type that contains all the operations, a tag procedure that attaches the new tag (that represents this type) to the data that is supplied to it and a series of ‘put’ calls to add entries to the op-table  (This is an additive process. No existing code needs to be changed.) | Write a new constructor that returns a procedure that dispatches based on the operation name supplied  (This is an additive process.) |
| Add New Operations (as in real-part, imag-part, magnitude, angle etc. in the case of complex numbers) | 1. Implement the new operation under each existing representation while avoiding name-conflicts between representations 2. Implement a new generic procedure that, based on the data type dispatches to one of the newly implemented operations above. | 1. The new operation needs to be implemented in each the existing representations 2. A ‘put’ call needs to be added to each package so that the new operation is added to the operations table.   (This is also additive.) | All existing constructors need to be changed so that their dispatch procedures check for the newly added operation also. And the implementation for the new operation should be added to the new conditional check clause |

Clearly, data-directed style and the message-passing style are both better options than generic operations with explicit dispatch. Much less work needs to be done in these approaches.

If new types need to be added often, the message-passing style is more suitable.

If new operations are added often, both data-directed style and message-passing style are equally good.