SUMMARY OF DISCUSSIONS

1. Requirements for meteorological models

The computer requirements sought by scientists for meteorological models in the late eighties, early nineties were discussed. There was a considerable difference between what was considered desirable for meteorological purposes and what was considered feasible within this timescale.

1.1 Computing speed

It was considered that ideally a peak performance of 100 Gigaflops would be required in the early nineties, but realistically a CPU performance of 10-30 Gigaflops was sought.

1.2 Memory

100 Megawords was considered the absolute minimum necessity for readily accessible memory, that is, memory more readily accessible than disk. However, it was agreed that the possibility of running several models at once was desirable, and therefore 500 Megawords was considered a more realistic working target.

1.3 Algorithms

The possibility of splitting algorithms across processors was discussed. It was agreed that a fine balance would need to be struck between the advantage

gained by the splitting of algorithms and the disadvantage of the supplementary coding and processing resulting from the additional I/O and control necessitated by such splitting. None of the group had any plans to create or investigate new algorithms.

1.4 Parallelism

The general opinion was that only a rather low level of parallelism would be available within this timescale, that is, total of 8-16 processors as a maximum. This was not seen as a cause for concern (see 2.1). It was also felt that algorithms used with hundreds rather than tens of processors would need intensive investigation.

1.5 Languages

It was generally agreed that it would be difficult to change from Fortran as the language of meteorological programming. However, it was strongly felt that new techniques could, and should, be grafted onto the original language. A good vector extension language would be a very useful addition, as would multitasking constructs, which, it was suggested, might be taken from ADA.

2. Consequences of requirements

2.1 Number of processors

It became apparent during the discussions that the smaller the number of processors the better, the optimum remaining a uniprocessor running at the required speed, because of all the additional overheads caused by

multiprocessing. The aim should therefore be to combine as few high powered elements as possible to achieve the operating speed required. It was believed, however, that economic considerations will eventually force the change to large numbers of small processors, since they are likely to prove much cheaper to acquire. These small processors will become very specialised to the point where, eventually, the algorithm is built into the operating system. This level of specialisation would not, however, be appropriate to meteorological applications, since many of the algorithms are subject to constant development and change.

2.2 Memory layout

It was noted that problems caused by the demands of multitasking upon memory resources had already been experienced: large areas of memory shared in read-only mode, memory bank conflicts, accidental memory overwrites etc. In general, it was felt that the more processors there were involved,the more problems were likely to arise.

2.3 Problem decomposition

Currently used algorithms can all be decomposed fairly satisfactorily, however memory management becomes a problem, without the introduction of hierarchies. There has not yet been enough development of software to protect one process from another and debugging poses ever more complications. It was noted that the number of processes in problem decomposition was linked to the availability of memory resources: the more memory available, the higher the level of decomposition possible, which would in turn probably lead to a higher

proportion of multitasking. However, once the number of processes exceeds the number of processors, the advantage begins to be lost and unbalanced tasks may become a problem.

2.4 Number of processes

The optimum number of processes depends upon the memory layout of the machines available. It was impossible to come to any conclusions in this regard without having additional information from individual manufacturers.

3. The tools required to assist in the programming of multitasking machines

3.1 Languages

There are some good languages available at present. Meteorology, however, needs a language which is not only efficient but which can also be easily maintained by a number of individuals. Moreover, code produced should be easily transportable because of meteorology's international nature. The language should allow flexibility for the development of new models, yet at the same time be capable of efficiently running an operational model.

3.2 Memory protection

One of the most difficult problems to be overcome with multitasking is the protection and efficient use of shared memory. It was considered essential that manufacturers devise a method of comprehensive built-in hardware protection of memory.

3.3 Debugging facilities

Multitasking makes debugging extremely difficult and the greater the number of tasks running in parallel, the more complicated becomes the location of problems and errors. Users considered the development of sophisticated and comprehensive debugging facilities to be essential to the production of efficient multitasking applications.

4. Future work

Finally, the group considered which of these areas needed further investigation. More needs to be learnt about problem decomposition: at what point exactly is the advantage gained by decomposition lost through synchronisation, I/O and other overheads? What is the best method of achieving task control? The tools required to aid multitasking programmers - Fortran extensions, debugging tools, etc., need closer investigation and users should attempt to make their requirements known to manufacturers. Finally, more needs to be known about the kind of memory structures which will be offered by manufacturers, so that this can be taken into account when considering degrees of parallism.