Abstract.- The usual maintenance techniques are not useful when dealing with non-stop software systems. It is not possible to stop the system execution to update some of its components. Dynamic software replacement is a mechanism that allows components updating without stopping the whole system. For this purpose, the Software Replaceable Unit (SWRU) concept has been introduced.

In this paper we present a design method for software systems including replaceable components. The executable unit (i.e. the process) is selected as the minimal software unit (SWRU) that can be dynamically replaced. This way, the strong relation between SWRU based software and distributed software is shown. This fact is used to extend the HOOD method in order to design SWRU based systems.

A replacement model is defined to implement the above concepts. The requirements that a SWRU has to fulfil, according to the above model, are identified. Finally, a method and a set of support modules to assist in the detailed design and coding phases are presented.

Key words.- dynamic replacement, maintenance, non-stop systems, distributed systems, design methods.
1. Introduction.

It is not possible to develop software systems that do not need to be modified during their operational life. This process, which is known as maintenance, is the longest phase in the software life cycle, specially when dealing with long-lived systems. In addition, the cost of software maintenance is significantly higher than the development cost. Software maintenance falls into three categories [Sommerville 89]:

- **Perfective**: Changes which improve the system in some way, without improving its functionality.
- **Adaptive**: modifications due to changes in the environment of the program.
- **Corrective**: modifications due to the corrections of system errors.

The usual software maintenance techniques are performed by stopping the whole system, installing the new version, and finally resuming the whole system execution. Under this approach, during a certain period of time the system is not operational and this can cause various types of problems (economical, safety problems, ...).

There are systems that cannot be stopped without large risks. This is the case of the Columbus space platform, whose operational life is estimated at 30 years. Stopping an on-board software system may cause major problems. In this case the classical approach is not valid. The alternative is to develop a system that can replace parts of its code dynamically, while the rest of the system continues working. This approach is forced by the requirements of non-stop systems.

Some work on this area was promoted by the European Space Agency (ESA) for the Columbus project. In this project a new software concept was introduced: the Software Replaceable Unit (SWRU) [Antler 87]. Indeed, all the COLUMBUS on-board software shall be configured as a set of SWRUs [COLUMBUS 90]. An implementation of this concept was studied in the Ada Real Time Test (ARTT) project [Amador 88], and its results were the starting point of the further works reported in this article.

Our research work has followed two directions:

- Study of the influence of the SWRU concept on the architecture of a software system as well as the integration of this concept into an existing software design method.

- The definition of SWRU implementation method for its use in the detailed design and coding phases. It has been performed by defining a Replacement Model, and designing a standard procedure and a set of support modules to implement the model.

All these ideas have been implemented and validated for the special case of a software system designed with the HOOD [HUM 89] & [HRM 89] method and implemented with the Ada language [Ada 83].

Section 2 addresses those questions that are open about dynamic replacement. In section 3, the minimal programming unit that can be replaced is defined. Section 4 identifies the