
**On-line experiment
control system
for the *BaBar* detector at
PEP-II at SLAC**

B. Franek

Rutherford Appleton Laboratory

for the BaBar Computing Group

E-mail: B.Franek@rl.ac.uk

Concepts and Requirements

- Babar consists of a well defined set of *components*. The behaviour of each component can be represented by a simple *Finite State Machine*.
- The *granularity* is defined by individual subdetectors e.g. for Drift Chamber: High Voltage, Gas System...
- Each component is responsible for establishing that it is in correct state for data acquisition ***RUNNABLE***
- *Control of the experiment* is equivalent to *controlling* and *synchronising* the state changes of the FSMs
- The control should therefore be a *hierarchical* finite state machine.
- It should allow to *start with a simple system* that has a minimum number of states such as RUNNING, STOPPED, PAUSED and CONFIGURED
- Then *gradually expand* to a more complex system that will attempt to automatically recover from all errors

Concepts and Requirements

Partitioning

- It allows to ‘split’ the detector into several *concurrent pieces*.
- This then allows *debugging* of various parts of the detector to occur in *parallel*.
- It also allows *multiple calibrations* to be done simultaneously. This functionality is very important for a large detector such as BaBar especially during detector commissioning.
- *Partition* - subset of components
- *Predefined standard partitions* e.g. everything, subdetector (SVT, DCH, DRC, EMC, IFR, TRG)
- *Session* provides a context for creating and dissolving partitions

To design and implement this system we are using :

S M I ++

S M I ++

Object Oriented Framework for designing *Distributed Control Systems*

It provides:

A method supported by a *formal language* to describe the real world to be controled and to design the control system

A set of *tools* to implement and test the control system

S M I State M a n a g e r I n t e r f a c e

Developed by DELPHI and used since 1989 1986-89
(*J.Barlow, B.Franek and M.Jonker*)
in collaboration with the CERN ECP Division
(*A.Defendini, J-P.Matheys, P.Vande Vyvre and A.Vascotto*)

++

Significantly upgraded (*B.Franek and C.Gaspar*) 1996-98
The main tool based on C++ (instead of ADA)

Documentation: [http://delonline.cern.ch/d\\$onl/smixx/doc/www/SMI.HTML](http://delonline.cern.ch/d$onl/smixx/doc/www/SMI.HTML)

Basic Ideas

of

S M I ++

Based on *Object-Oriented decomposition* of the problem
(the real world to be controlled + the control system)

- The problem is viewed as a set of autonomous agents (*objects*) that collaborate to perform a desired behaviour
- Objects behave as *FSM*'s
They exist in discrete *states* (their only visible attribute)
- Objects do things (execute *actions*)
- Two kinds of objects : *associated* and *logical* (control)

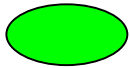


Basic Ideas

of

S M I ++

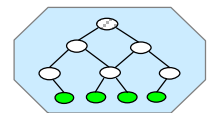
Associated objects are abstractions of the hardware components in the real world (e.g. HV Power supply)



Logical objects are abstractions representing an identifiable entity with a well defined role in the control system (e.g. Subsystem Top Control)



Objects are organised in logically related groups; **domains** (e. g. a Subsystem Control)

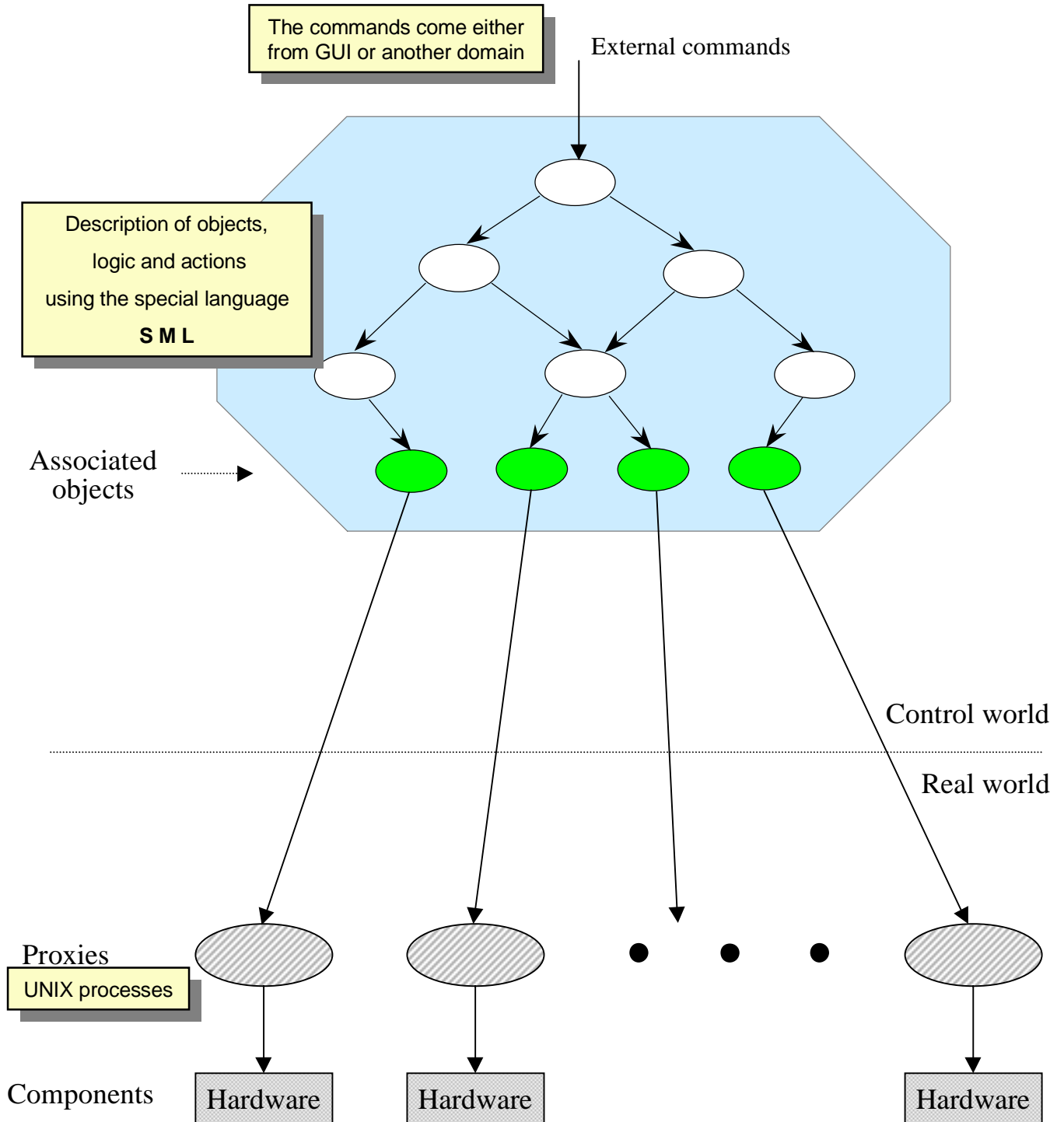


Only few objects in each domain are visible to the other domains

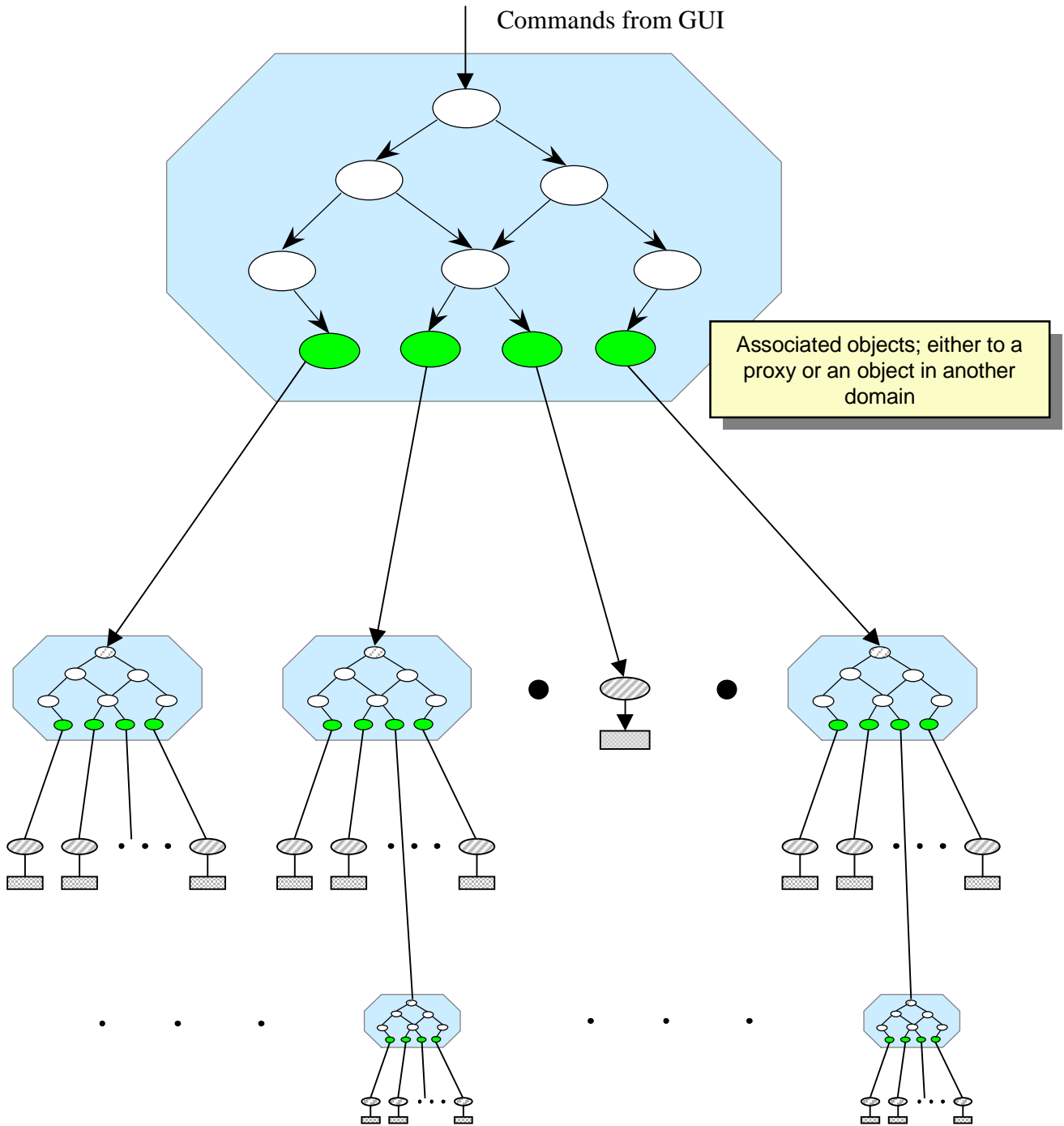
In summary :

The task of creating the control system is equivalent to that of giving a good description of the real world in terms of domains of objects being controlled and the procedures (embedded in logical objects) that operate on them

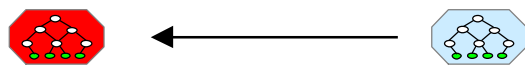
Domain



Hierarchy of domains

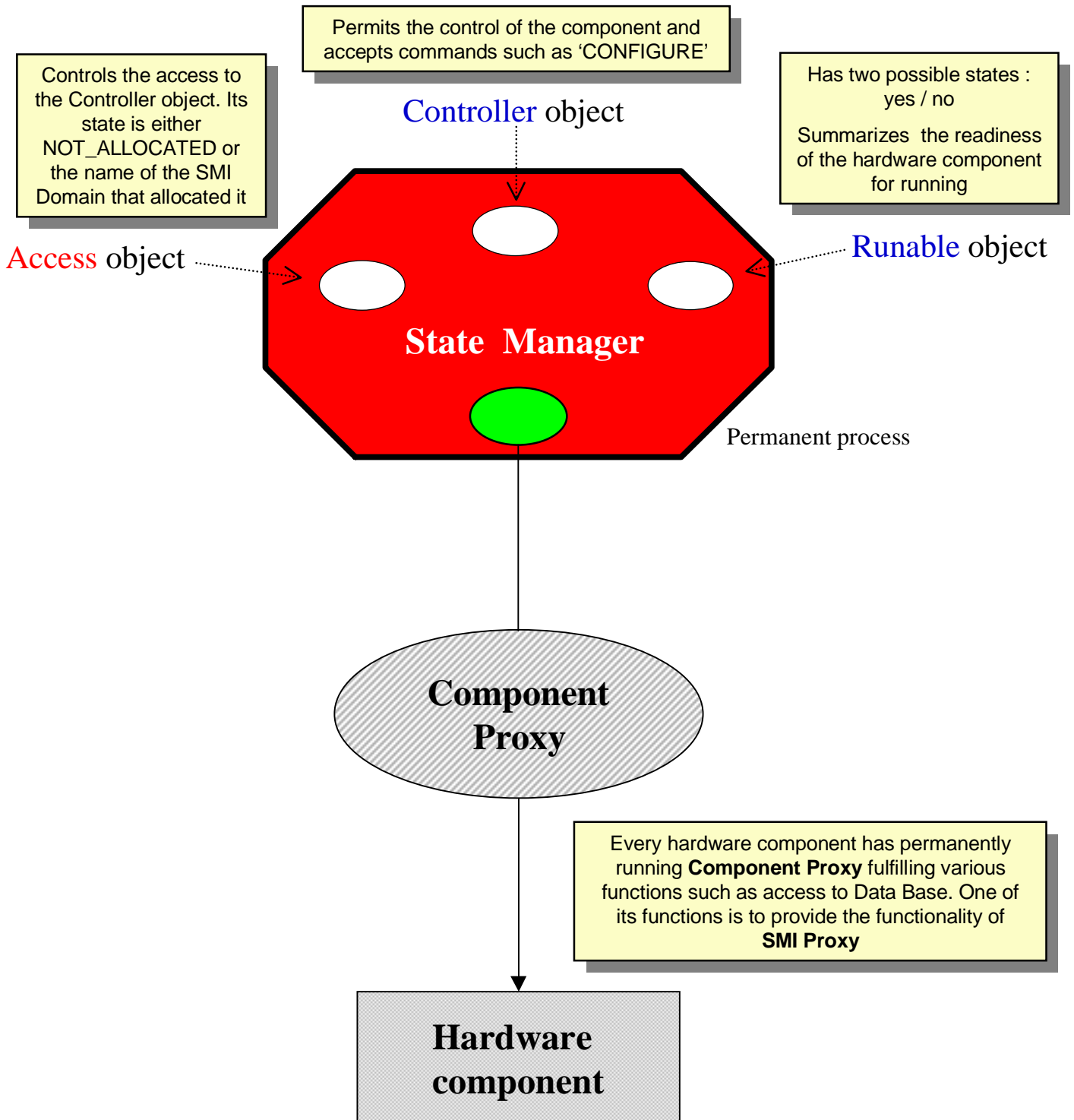


This SML description of the Control System is 'realized' by Logic Engine process called **State Manager**
State Manager / Domain



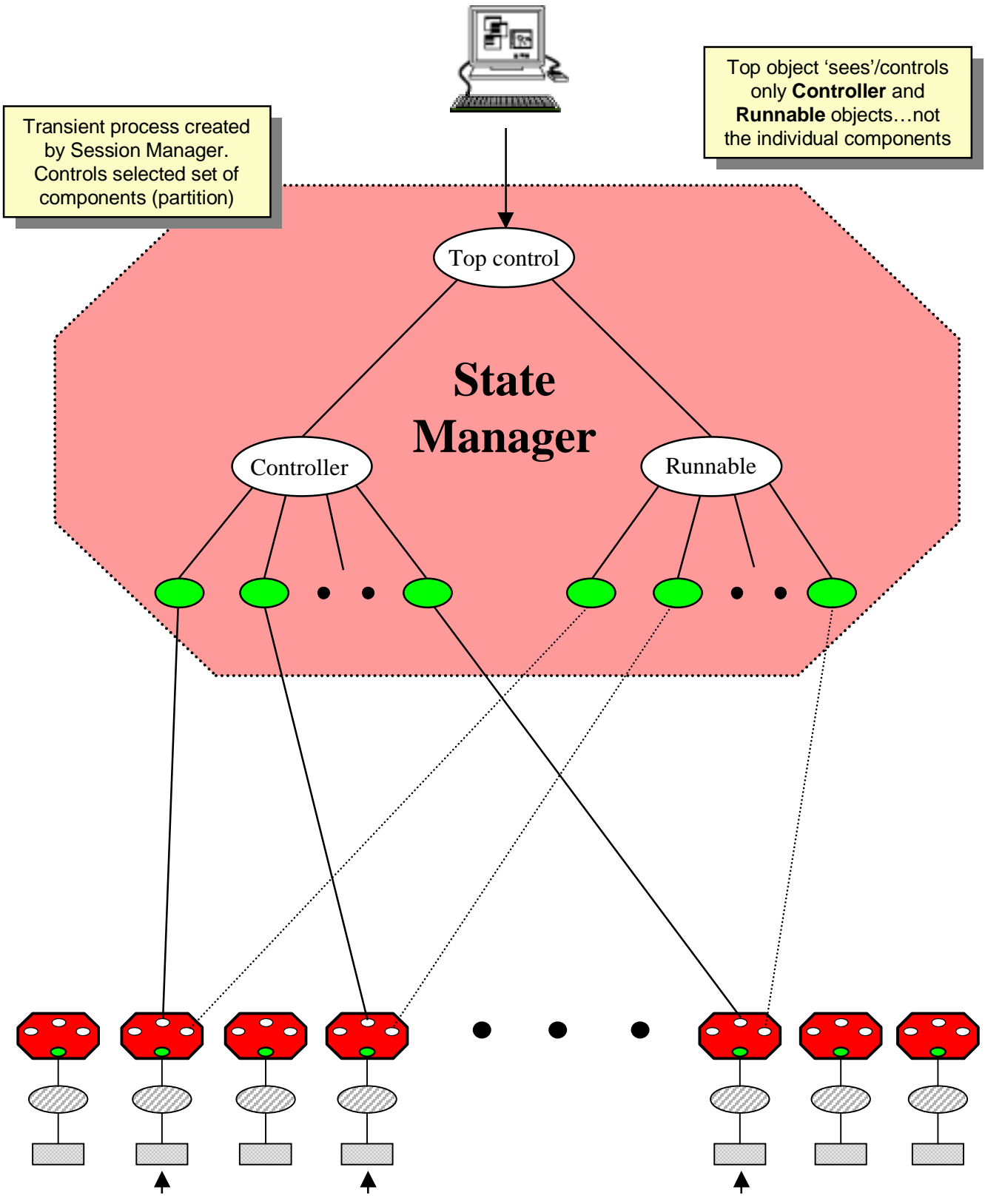
Component Control

BaBar



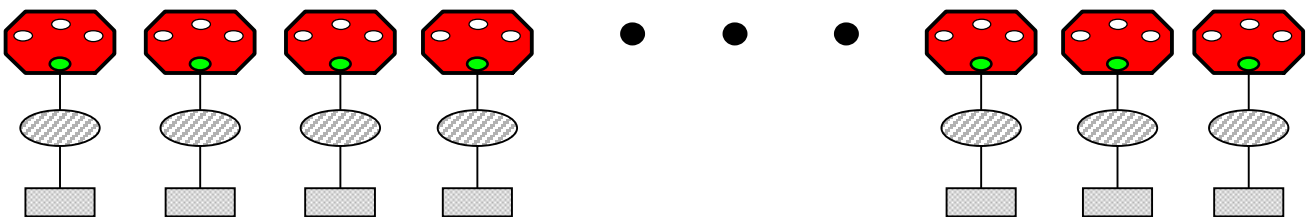
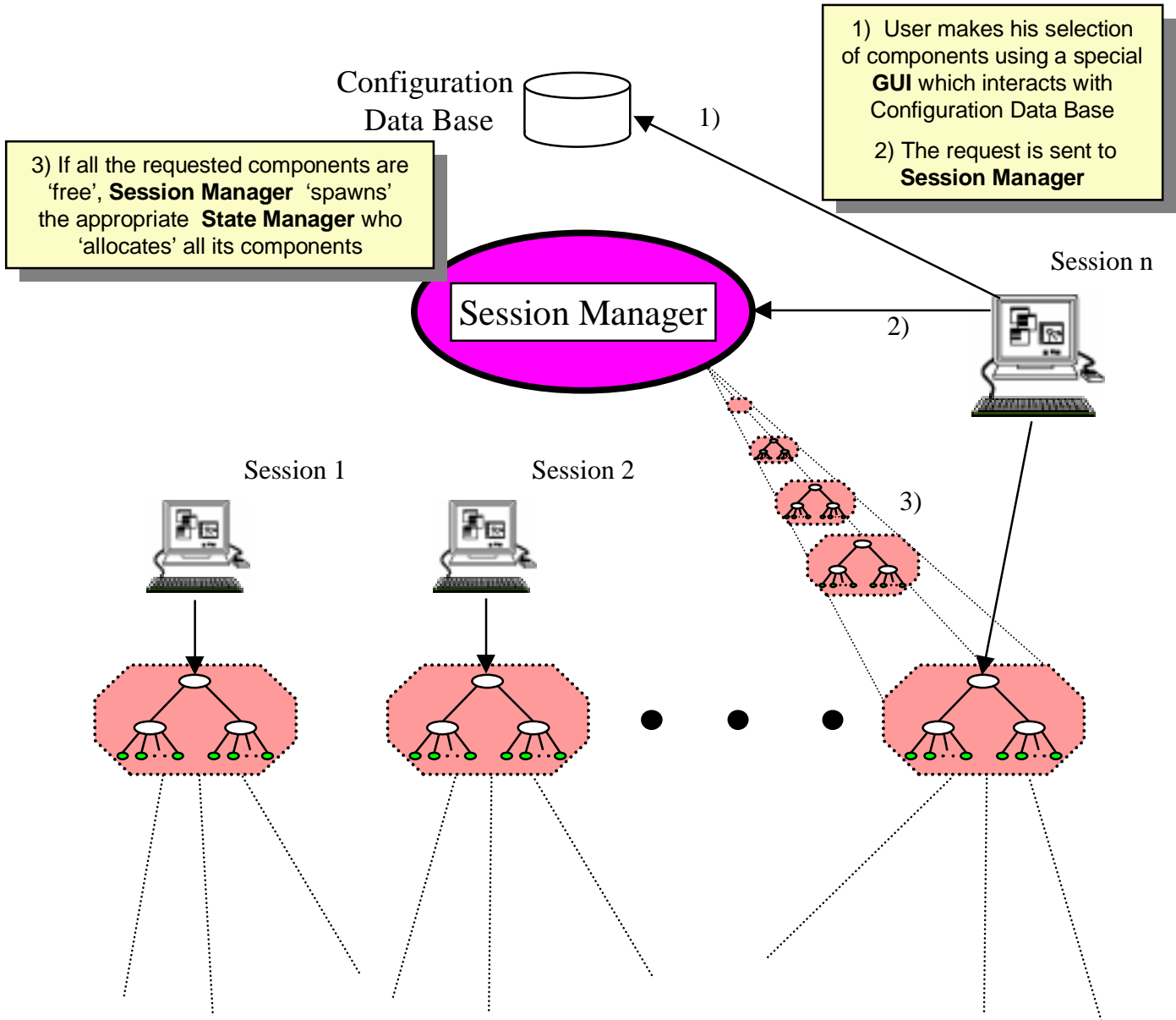
Single Partition Control

BaBar



Session Management

BaBar



Summary

- Strict adherence to *Object Oriented* methodology
- using *SMI++* tool
- Design and implementation of the *Component control* finished
- Design and implementation of the *partition control* in the final stage
- Work in progress on the *session management*