Security through Bracket Methods

Klaus Espenlaub and J. Leslie Keedy

Department of Computer Structures,
University of Ulm, Germany

email: keedy@informatik.uni-ulm.de

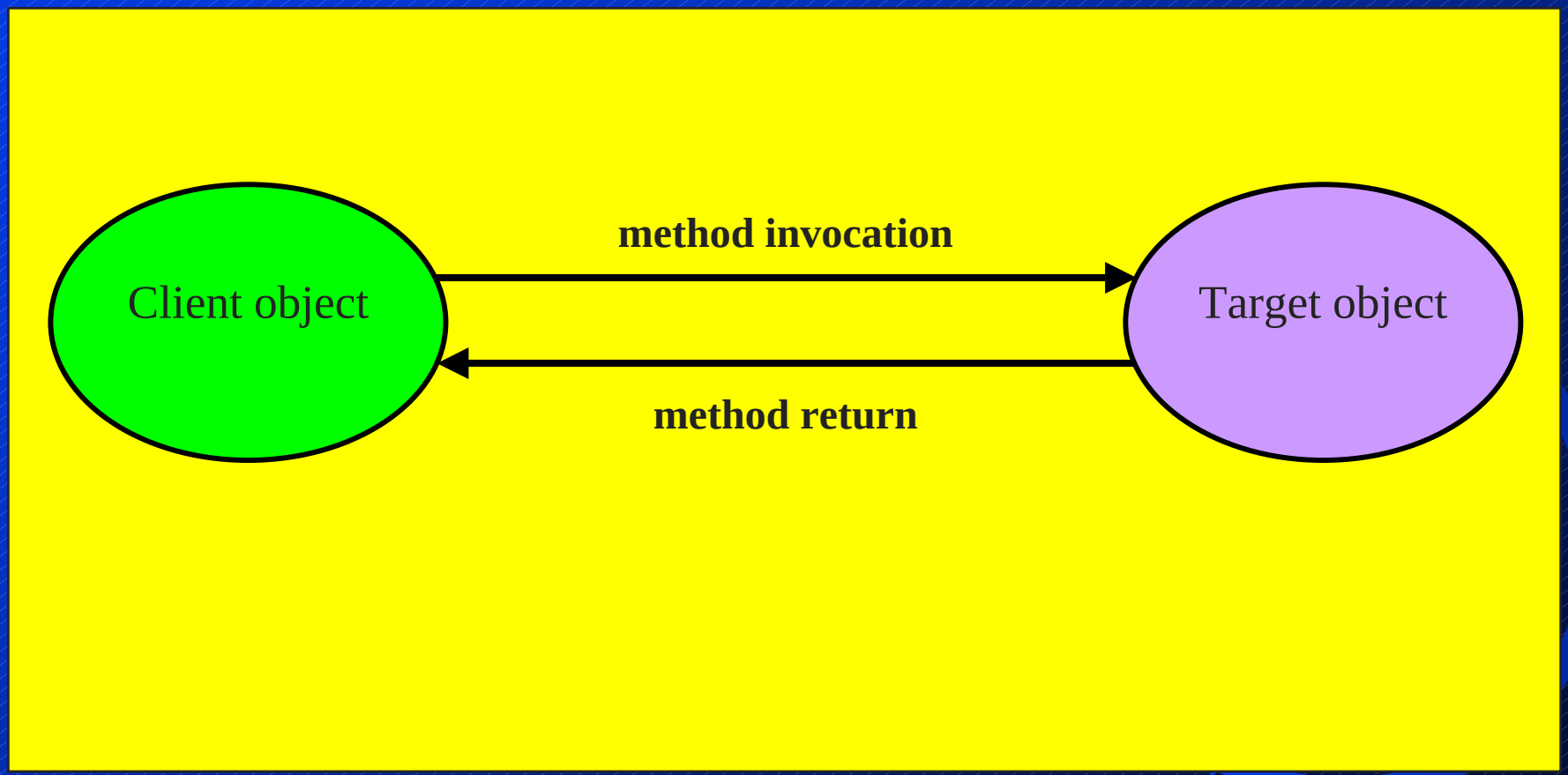
SPEEDOS

Qualifiers and Bracket Methods: The Principle

- A qualifying object (qualifier) is a normal object which also has *bracket methods*.
  - These are methods which are not directly invoked.
  - Instead they are activated when a client object invokes an appropriate method of a target object.
- Thus a bracket method appears to *catch* an invocation of a target method of an object.
- The kernel activates the qualifying objects associated with the target object.

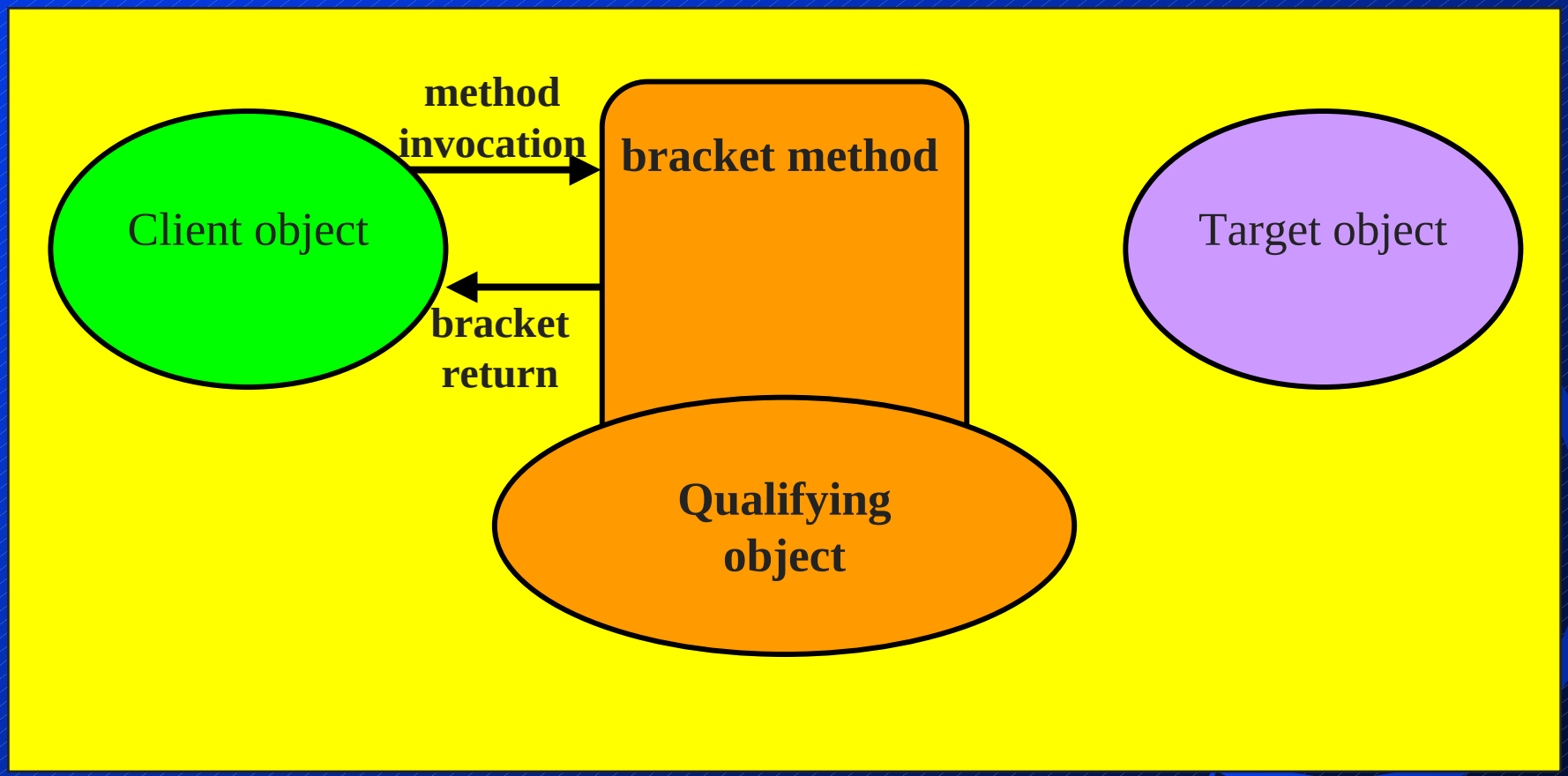
Qualifiers and Bracket Methods: The Principle

- Here is a normal method invocation:



Qualifiers and Bracket Methods: The Principle

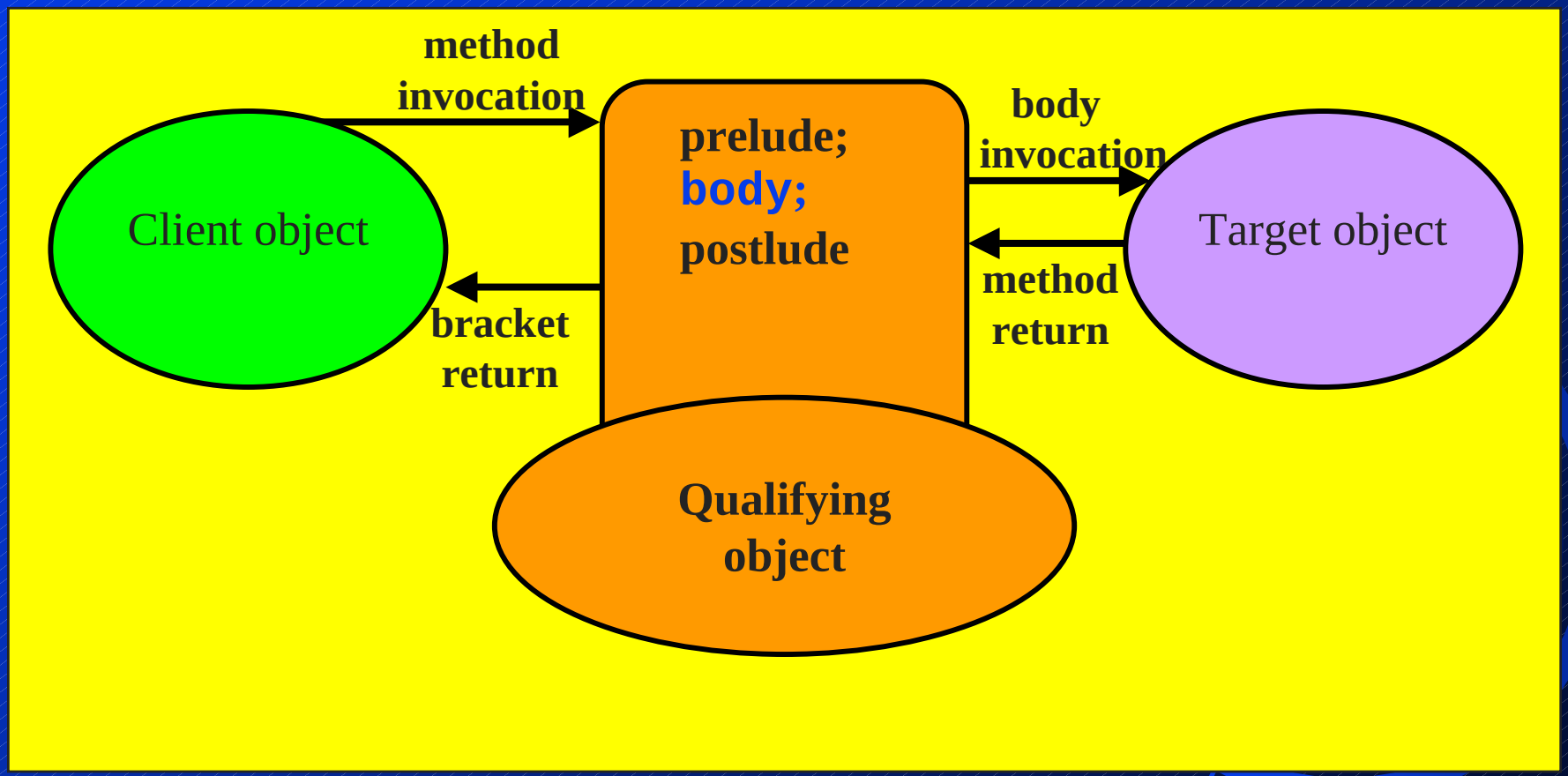
- Now we add a qualifier:



Qualifiers and Bracket Methods: The Principle

- In this form a bracket method *replaces* the method which was invoked by the client.
- But there is a special mechanism (which we designate here by the keyword **body**) to allow the bracket to invoke the target method.
- In this case the bracket method can be viewed as having
  - a *prelude* (code before the target is invoked), and
  - a *postlude* (code after the target is invoked).

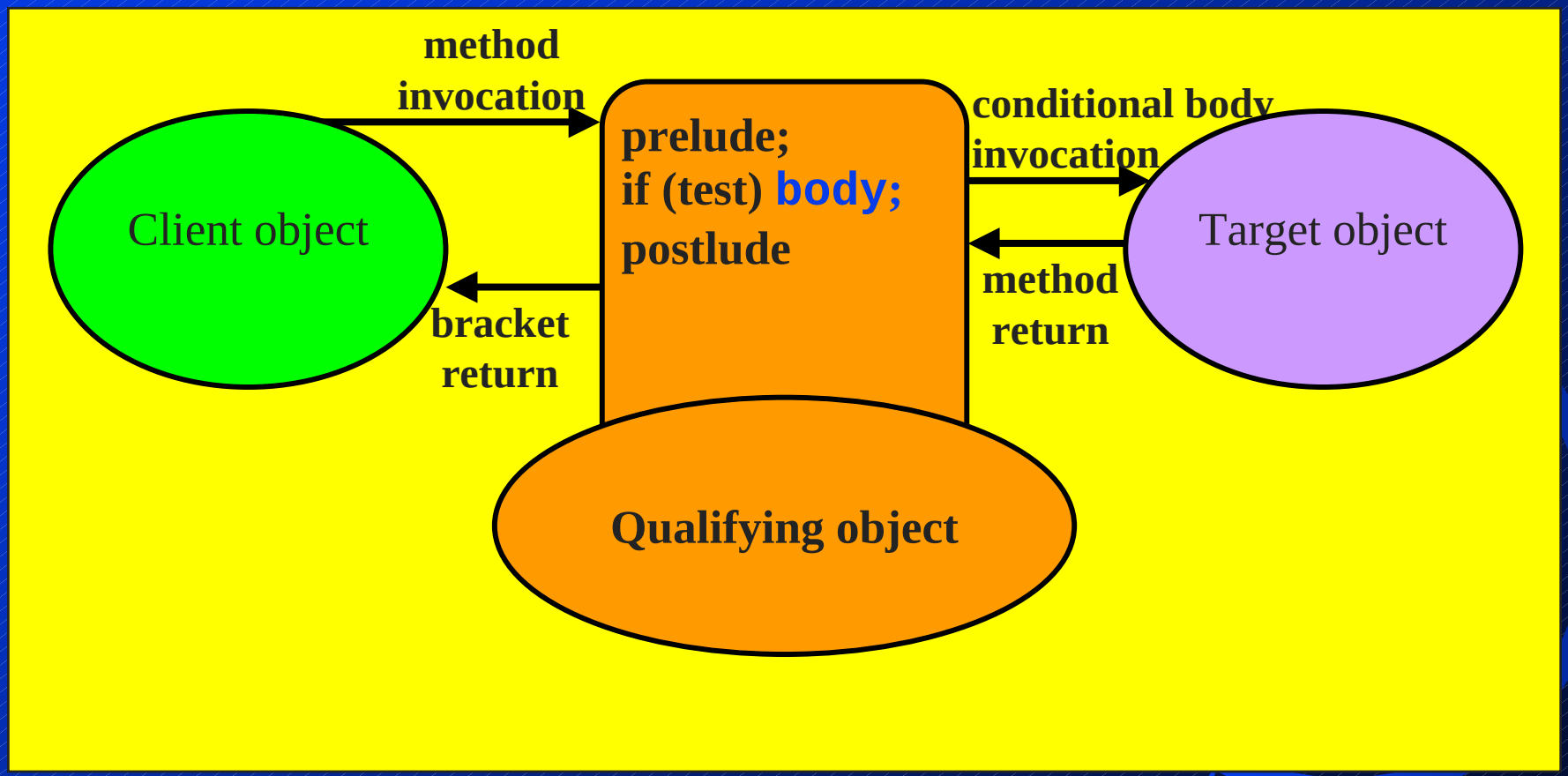
Qualifiers and Bracket Methods: The Principle



Qualifiers and Bracket Methods: The Principle

- The **body** mechanism is in fact a normal statement (a special variant of a method call), implemented in *SPEEDOS* via the kernel.
- Hence it can be executed conditionally.

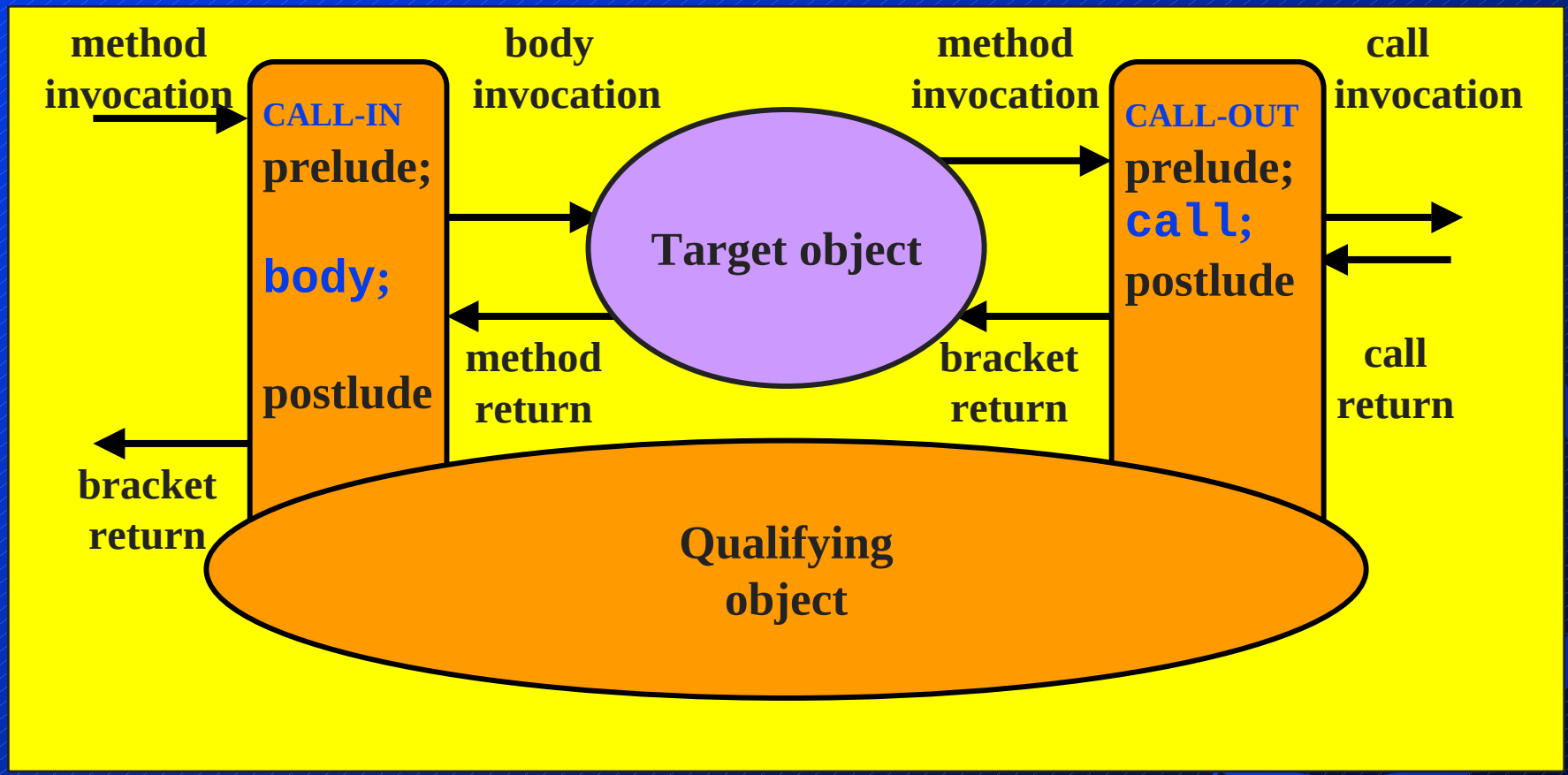
Qualifiers and Bracket Methods: The Principle



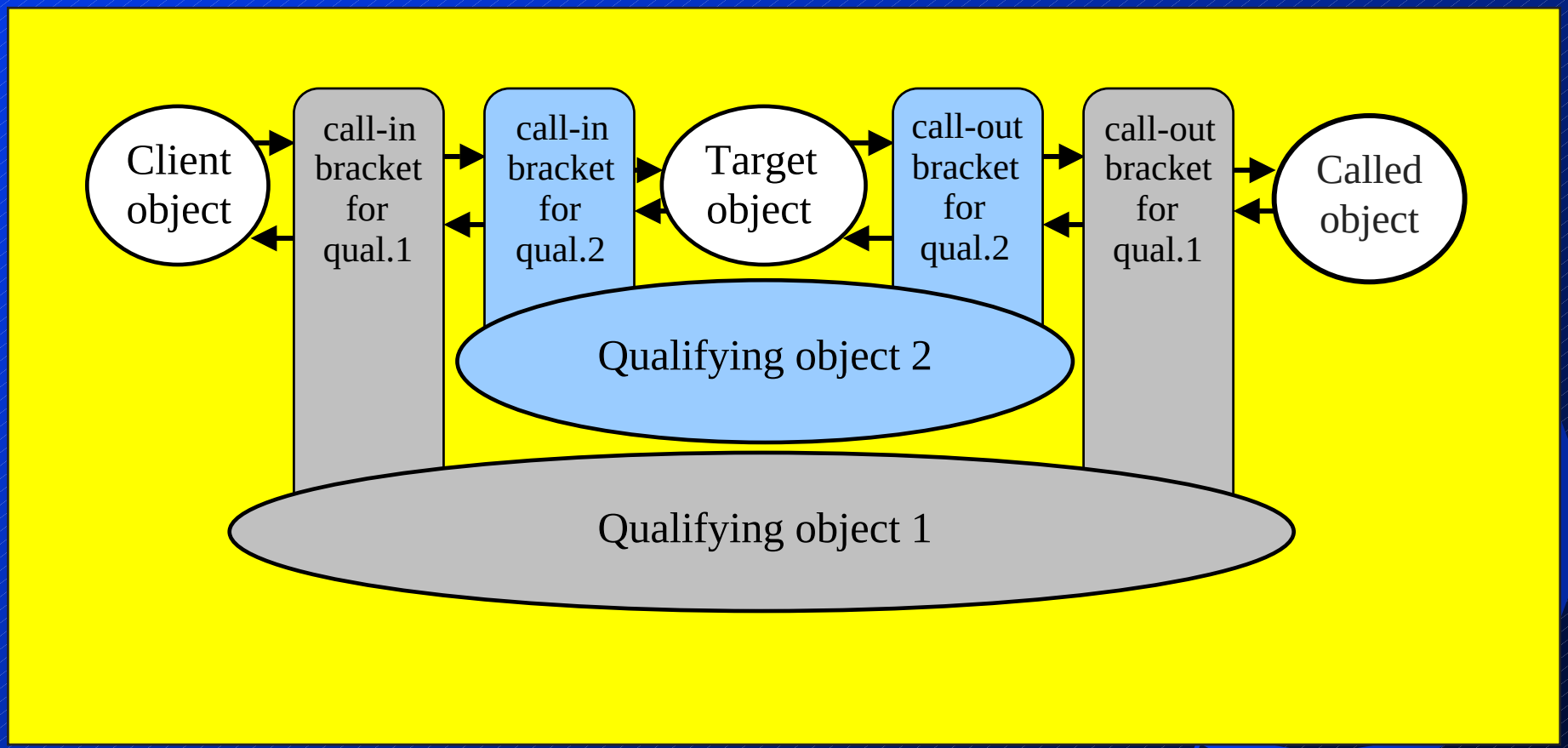
Kinds of Bracket Methods

- There are two kinds of bracket methods:
  - *Call-in* bracket methods are applied to a target object as its methods are invoked. Here the keyword **`body`** is used.
  - *Call-out* bracket methods are applied to a target object as it invokes methods of other objects. Here the keyword **`call`** is used.
- As all objects can be qualified in this way, the same object can have both call-in and call-out methods associated with it:

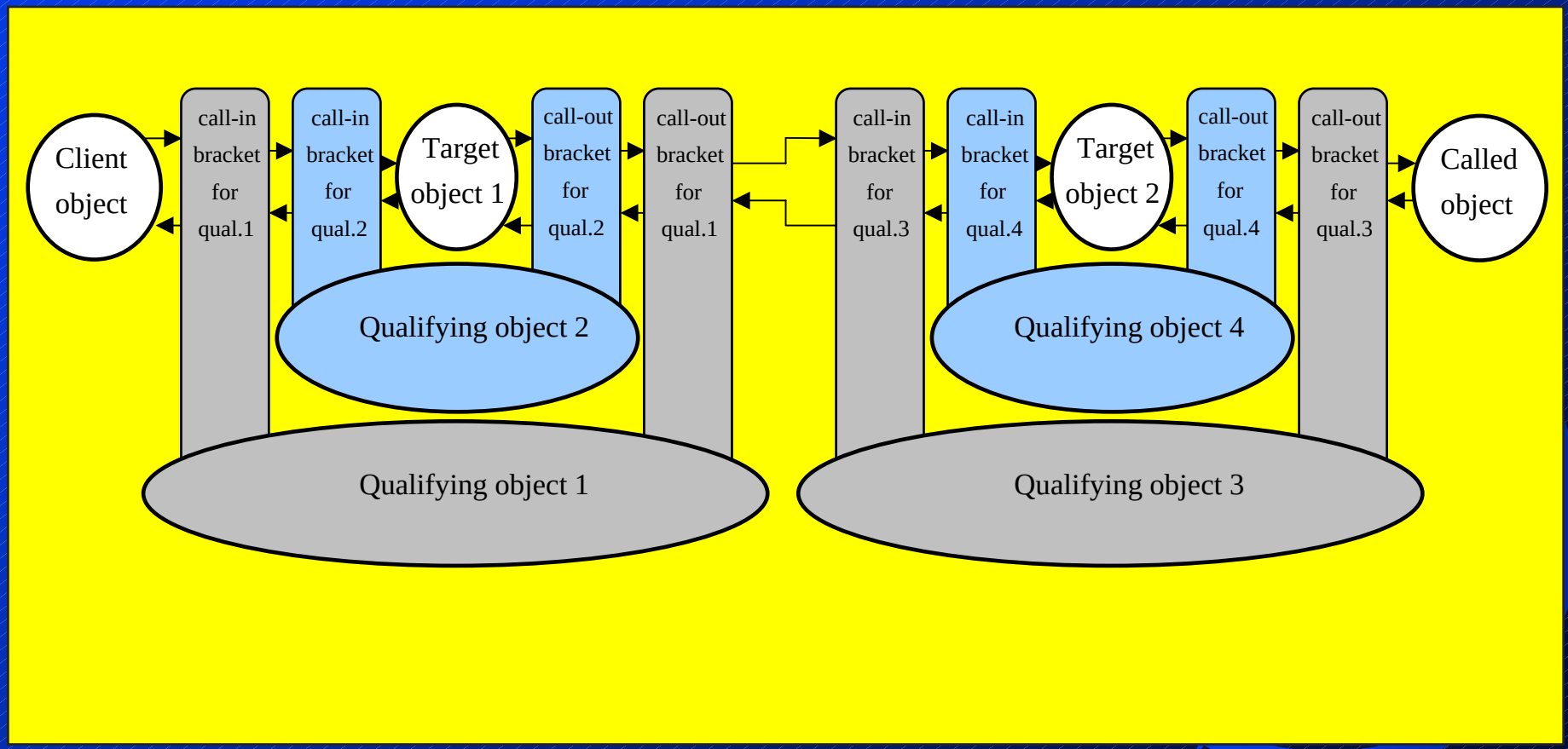
An Object Qualified by a Qualifier with Call-in and Call-out Methods



An Object Qualified by Multiple Qualifiers



Multiple Qualified Objects



Which Methods Can be Qualified?

- In the language Timor, also being developed in Ulm, there is a distinction between an object's
  - reader methods, and its
  - writer methods.
- These can be bracketed separately.
- Methods can be bracketed individually and their parameters inspected or modified in bracket methods.
- In *SPEEDOS* this is left open, but modules can be developed which support all possibilities.

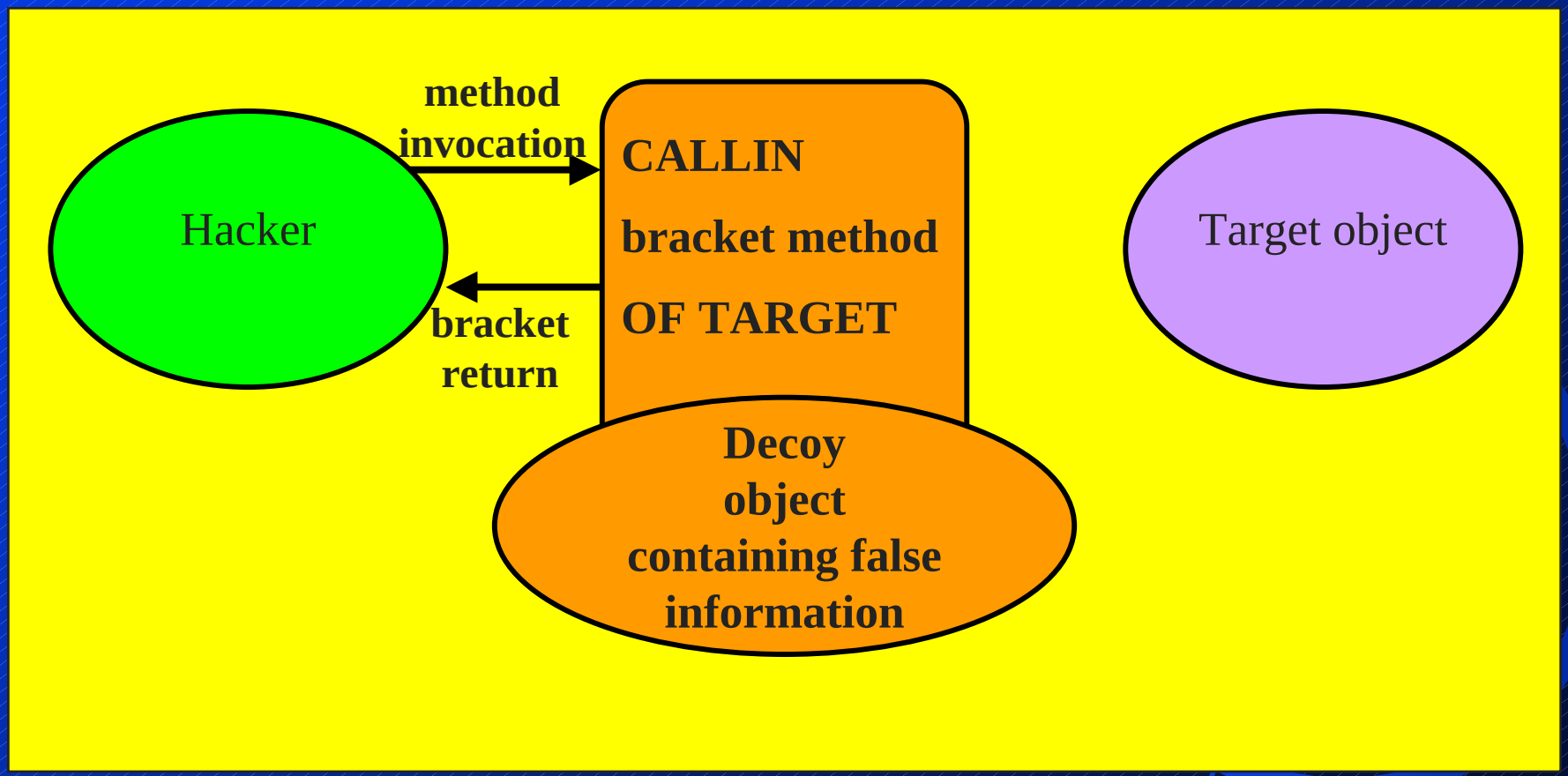
Access to Data

- Qualifiers are separate objects with their own data items (in *SPEEDOS* persistent data).
  - These data items can be accessed by bracket methods.
  - A qualifier can have normal methods which may access (e.g. set up and modify) these data items.
- A bracket method can (in appropriate cases) access parameters being passed to a target, but has no access to the data of their targets.

Some Security Possibilities: Stalking a Hacker

- A system manager who discovers that a hacker is breaking into his system (recall Clifford Stoll and the Bremen hacker) can easily take preventative measures.
- To do this he can set up a qualifier as a *decoy*, which (for example) feeds back false information to the hacker and records information about the hacker's activities.

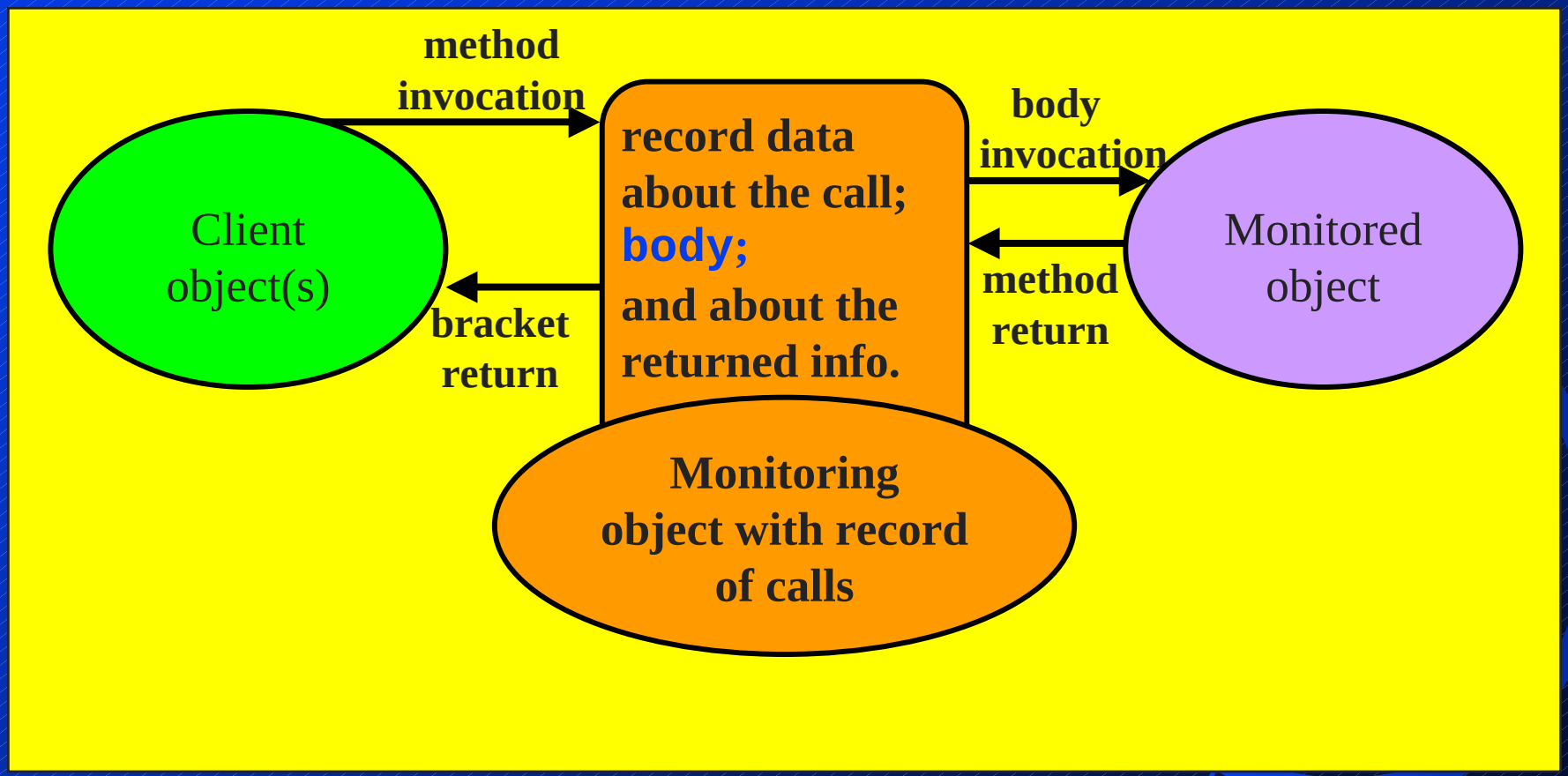
Protecting Information and/or Access from a Hacker



Some Security Possibilities: Monitoring the Use of Objects

- A target module can have an associated qualifier which records information about the clients who invoke the target.
- Neither the client nor the target have to be modified to include the bracket functionality.
- Information collected can be stored in the persistent data of the qualifier.

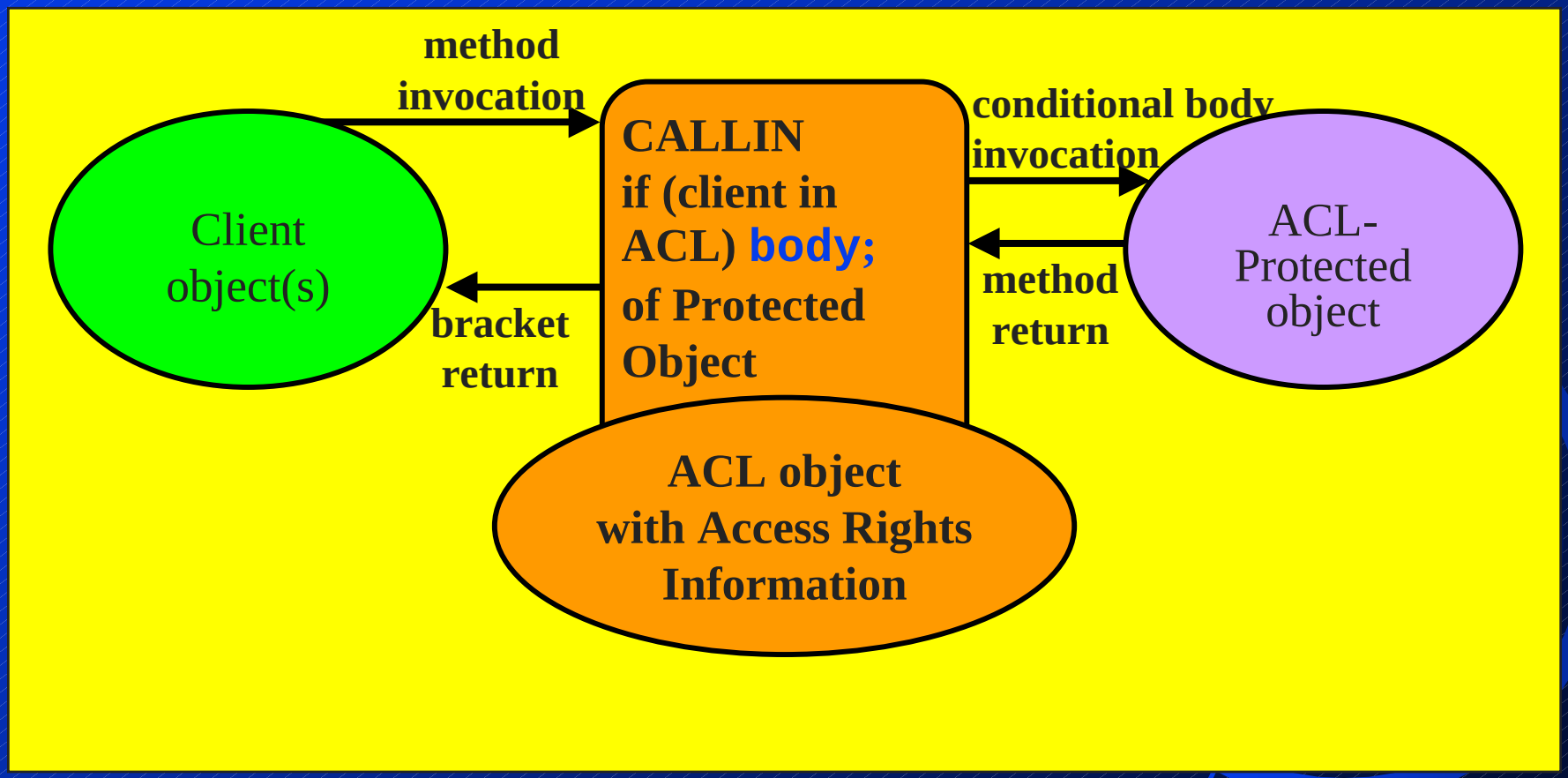
Using Qualifiers to Monitor the Use of Objects



Some Security Possibilities: Installing an Access Control List

- A target module can have an associated qualifier which maintains an ACL in its persistent data.
- Whenever a caller invokes a method of the target, the appropriate bracket method checks the right of the client to call the module.

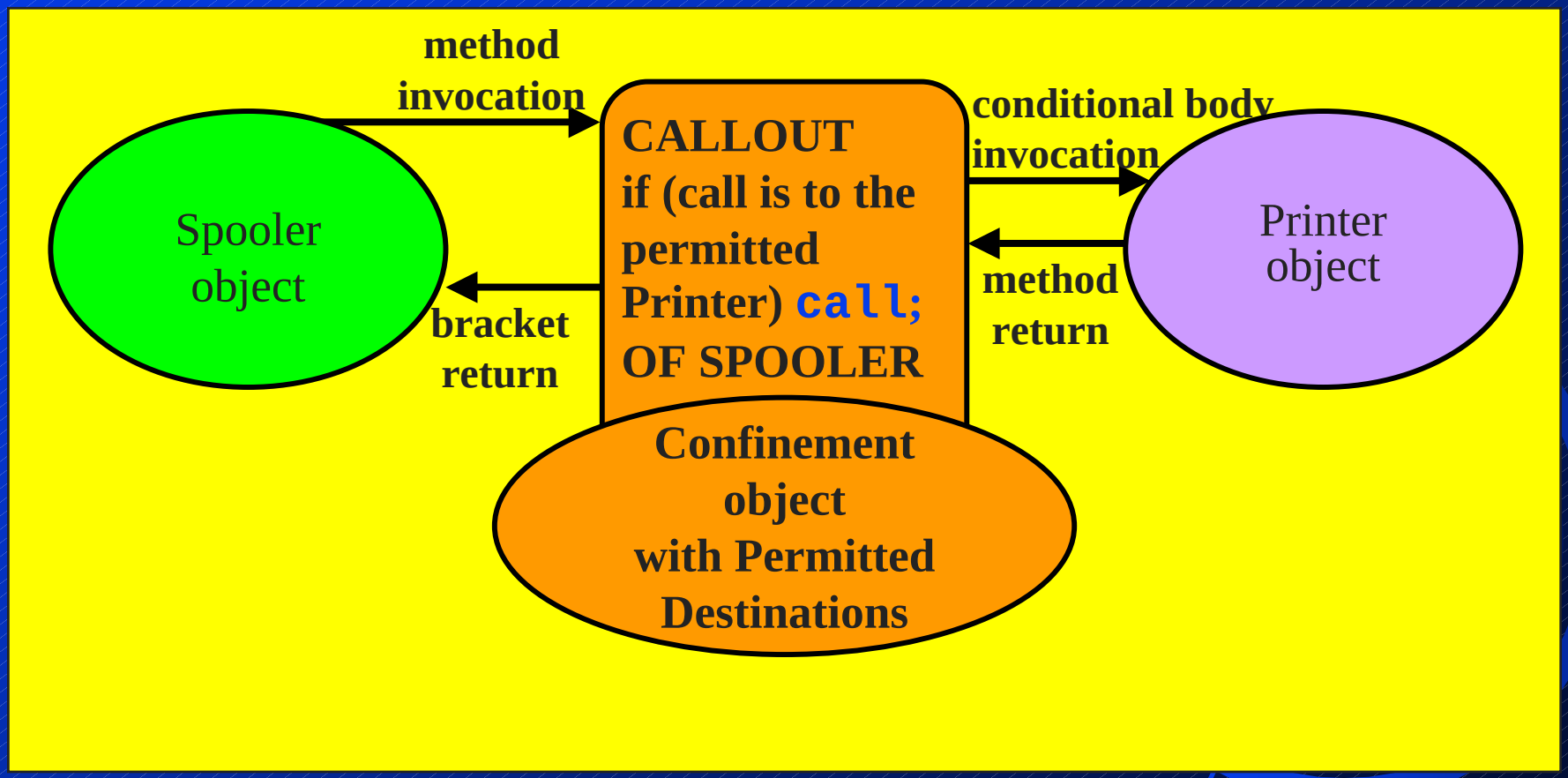
Installing an Access Control List



Some Security Possibilities: Confining an Object

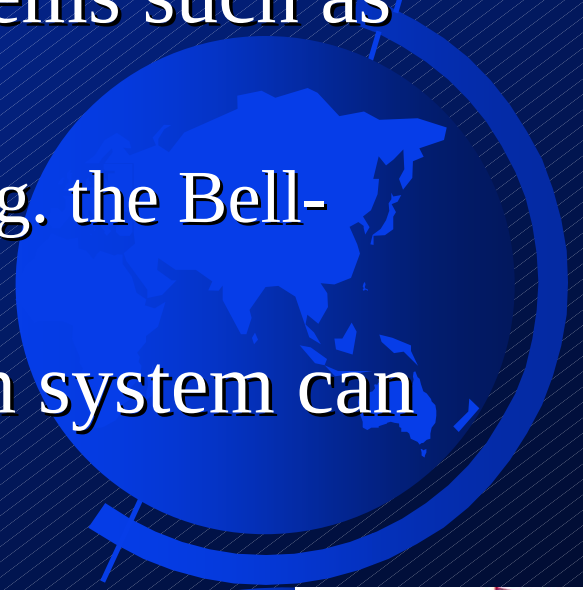
- A module can have an associated *call-out* qualifier which ensures that it only accesses permitted destination modules.
- This can be used for example to confine a spooler to using only a permitted printer.

Confining a Spooler



Conclusion

- Qualifiers with bracket methods provide limitless possibilities for enforcing protection requirements in an appropriately designed system.
- This includes solutions to problems such as
  - capability revocation,
  - various confinement problems (e.g. the Bell-LaPadula security model).
- In fact any rule-based protection system can in principle be implemented.



Security through Bracket Methods

Klaus Espenlaub and J. Leslie Keedy

Department of Computer Structures,
University of Ulm, Germany

email: keedy@informatik.uni-ulm.de

SPEEDOS

Web Information

SPEEDOS:

http://www.speedos-security.org

MONADS:

http://www.monads-security.org

Qualifying Types in Timor:

http://www.timor-programming.org

SPEEDOS