

Re: too much OOP ?

Source: <http://coding.derkeiler.com/Archive/General/comp.object/2008-01/msg00049.html>

- *From:* "Dmitry A. Kazakov" <mailbox@xxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Sat, 5 Jan 2008 21:22:56 +0100
-

On Sat, 5 Jan 2008 10:48:57 -0800 (PST), Daniel Parker wrote:

On Jan 4, 10:44 am, "Dmitry A. Kazakov" <mail...@xxxxxxxxxxxxxxxxxxxxx> wrote:

On Fri, 4 Jan 2008 07:14:02 -0800 (PST), frebe wrote:

Others
quietly
create
their
libraries.
They
might
call
them
"engine",
"middleware",
"framework",
but
show
nothing
alike
that
manic
disorder
the
former
suffer...
(:-))

What
libraries
are
you
suggesting

Re: too much OOP ?

instead
of
a
SQL
database?
Please
show
me
some
example.

There exist
lots of
container
libraries.

Please point out one library
that could be used instead of
a SQL
database for an inventory
management application?

As I said, write the requirements. The
problem of you and the OP is that
you don't care to write the requirements
down. Instead of that you think in
terms of DB records, i.e. an implementation.
The rest follows. In OO one
would first identify and describe the things
and only then say, "hey, that
looks much like a relation!"

You claim that it does exist libraries that can be used instead
of an
SQL database, and do the job better.

Which job? Describe it in the form of requirements.

Here's one. As a business analyst, I want to look at all of the data
in a production trading system. I want to look at the data one way,
then another, as a preliminary step in writing requirements. Getting

Re: too much OOP ?

Re: too much OOP ?

a SQL database account is a pretty good way of achieving that objective.

You cannot say that without specifying what kind of requests are planned. You made three assumptions:

1. For most of requests an RA description is close to optimal;
2. SQL is the best way to spell RA requests.
3. All other components of the system are negligibly (in terms of implementation and maintenance costs), and even more generally, that most of the system functionality is exhaustively described as performing requests.
4. The non-functional requirements allow deployment of an SQL database as the platform, or even as a component (and where is the rest?) [Such as memory constraints, real-time constraints, mobile distributed components, lag and blackout periods, security, safety etc]

Any or all of these four assumptions can be wrong. As I said, I know nothing about inventory management, but in my area of interest relational view does not deliver. This can be stated and proved mathematically. After all the algorithms behind SQL implementations are well known. It is also know when they are optimal and when not... SQL as a language speaks for itself... As for other things, I am dying to see a UI written in SQL...
(;-))

Nevertheless, in case you missed the beginning of this chat, it was not my point, that SQL database should never be used. My point was that designing the system in terms of records updates is low-level.

--

Regards,
Dmitry A. Kazakov
<http://www.dmitry-kazakov.de>

.