What is wrong with contacts management in OpenERP v7?

(as of April 2013, 4 months after the release, hopefully we may have fixed it when you will read that later)

v 1.1

by Raphaël Valyi

reminder: conceptual change in v7

now res.partner and res.partner.address (contacts or addresses) records all live in the res.partner table.

Only spec published about this was:

http://v6.openerp.com/node/1169/2012/06

Terms that we will use:

- **company**: what was a res.partner is 6.1 (customer or supplier commercial entity),
- **contacts**: addresses or contacts of these commercial entities.

reminder: conceptual change in v7

to simplify, we won't talk about physical persons for which there is little problem in v7 and focus on the issues with companies.

the main idea of putting everything in the same table was:

- to make B2C easier
- simplify menus and view definitions
- be able to use both companies and contacts with the same field/key when we can: ex to send a mail, select the contact of an invoice etc...

so what's the problem?

several many2one fields that were previously pointing to a company in 6.1 now accept a company OR a contact in 7.

For instance: partner_id on:

- invoice
- purchase order
- stock picking
- crm claim
- project task...

an exhaustive list is being made here https://docs.google.com/spreadsheet/ccc?key=0AlrjP6ETn3tJdC1CUEw2bGQ1RkhDR0lmS2dGT1E4eWc&usp=sharing

What you did to you?

1) The problem of the contact record information

when you invoice a contact (on purpose or because OpenERP does it for you as when you use the CRM), the account module code will try to read fiscal position, partner payment term, credit limit and other such fields all from the contact.

This information is not expected to be maintained on every contacts of a company of course!

This problem also happens when writing data.
This problem happens in many more modules/localizations etc...

1) proposed solution by OpenERP SA

automatically hard copy these data from the company to its contacts records.

PROBLEMS:

- data duplication is never a good thing
- even properties (like invoice account) are duplicated, that means even DUPLICATING these property records...
- this is totally incompatible with extensions like base_contact where a single contact could belong to SEVERAL companies
- OpenERP SA claims that it's expected you have to fill pricelist and some other fields right on every contacts so you have an idea what to expect "it's user fault"...

You F*~k
my wife ?!

2) Reporting is dead

wanted to compare

- total invoiced by customer?
- total purchased by supplier?
- compare supplier product prices?
- do your "intrastat" reports to declare where you sold your goods?

PROBLEMS:

All these reports that were doing **group_by** and **join** over partner_id as a company will now do it on mere contacts instead. **Unless you add a second company key** and change all these reports definitions, you cannot have such reports in V7.

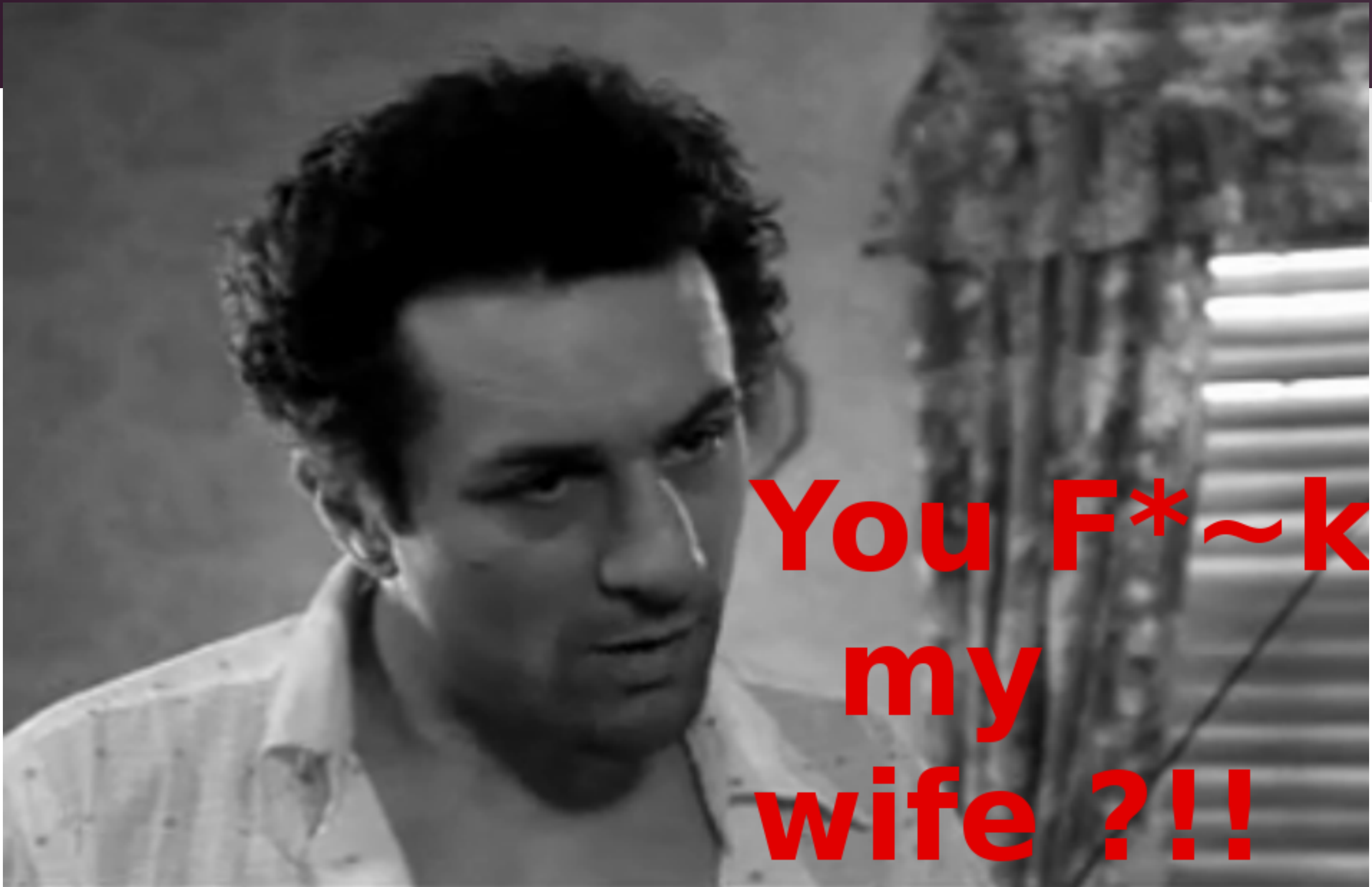
2) proposed solution by OpenERP SA

unf*ck just the invoice reporting by effectively adding a new "commercial_entity_id" key pointing again to the the companies, in a new module.

PROBLEMS:

- As for the other objects, dream on.
- modules needing such a key may add their own and hack reports definitions/filters in overlapping incompatible ways.
- hell we already had such a key before and it was called partner_id!

What?



You F*~k
my
wife ?!!

3) the SQL cardinality nightmare

everywhere we have some foreign key pointing to res. partner, it may be wrong! Fear not...

3) a) domains nightmare

search with ('partner_id', '=', some_id)

some_id may now be the contact on a company while the code was expecting to find a company

3) a) domains nightmare

search with ('partner_id', '=', some_id)

for instance the code may create a new invoice if no invoice is found for record with partner_id == some_id.

PROBLEMS:

- In fact an invoice may already exist for that company but not with the same contact as some_id...
- To fix that, we would need a **NEW KEY to the company** and then use a child_of operator.
- shocking: we already had that company key and it was called partner_id

3) a) domains nightmare

PROBLEMS

- that forces us to change the code in all official and community addons to change such domains and use child_of instead
- and to use the parent company of the contact if some_id is a contact.
- That also means slower code when the code had only some_id without browsing the object, like in an on_change function for instance.

Just tell me…

3) b) one2many and many2many nightmare

a code doing

```
for record in partner.related_records_ids:
    do_something_critical(...)
```

PROBLEM:

well, now some records that would have been included in the loop in 6.1 will unexpectedly be missing in 7 if

- their partner_id was set to a different contact of the same company
- OR if they point to the company while the partner is a contact of it.

3) b) one2many and many2many nightmare

unf*cking that would imply **getting partner parent company (ideally using a field pointing to it)** and iterating over each of the records related to each of its contacts and also the ones related to the company...

Not really making the code any simpler...

And it also means changing all the official and community addons code.

shocking: as for a field pointing to the company of a contact, we had it already, it was called partner_id.

Just tell me…

3) c) many2one nightmare

suppose you deal with Return Material Authorization (RMA) in OpenERP user interface.

You create a new return picking. You now want to relate that picking to a possibly existing claim ticket of that partner or else create a new one.

Typically, the ticket_id field will have a domain such as ('partner_id', '=', partner_id) in order to filter only the possible tickets to relate.

3) c) many2one nightmare

PROBLEM:

a claim ticket may already exist for the company but not the contact you selected in your picking in partner_id.

To fix it we would need a **stored key to the company** and filter with a 'child_of'

shocking: we already had such a key called partner_id

OpenERP SA's answer to that is generally: the contact is the right "granularity"! Oh is it? Tell me, for what granularity was the module built before when partner_id was a company?

3) d) unexpected: more complex access rules

imagine a rule like:

sales users will see only the opportunities and orders of their own portfolio?

PROBLEM:

Well now you will need to make sure that every time a new contact of customer is created, it will be related to the right salesman or else write much more complex rules.

That is, it's more complex to set up or else they will miss documents unexpectedly

CONCLUSION 1

1. **ERP documents CANNOT make it without an SQL key pointing to the company**. Despite one can usually infer the company from the contact, only the contact isn't enough in practise.
2. to be able to filter with that key before saving a record, it needs to be set with an on_change, that is solution invades even view definitions.
3. it cannot be a fields.related because it would be set only when document is saved and isn't compatible with the case where a contact could belong to several companies.

CONCLUSION 2

1. They aren't telling you the truth when they deny documents need a key to the company
2. They aren't telling you the truth if they finally admit some documents like invoice may in fact need such key, but may be not all kind of documents.
3. **We already had that key: it was called partner_id**
4. During 8 years, hundreds of developers used partner_id as a key to the company, this is exactly what hundreds of modules are expecting.

CONCLUSION 4

1. It's not acceptable that we should live test in production all modules designed for companies which now can receive contacts randomly.
2. It's not acceptable we should change all the semantic of partner_id everywhere when bugs will be discovered and make it a pointer to a contact and then progressively adding a new key again pointing to the company again (that is **swaping the semantic**) ??!??!??
3. how many **years** will it take to fix all these hidden functional bugs when it usually takes months to get a trivial regression fix on the official branches even when a patch is provided?

CONCLUSION 5

what kind of chaos will we have with hundreds of modules:

1. adding their own new keys to a company in overlapping incompatible on_changes (incompatibles view definitions)
2. missing the change and doing buggy things
3. rejecting the change and assuming partner_id is still a company and that the contact is carried by another key

?

CONCLUSION 6

A SOLUTION EXISTS!

1. It consists in automatically adding a new field **contact_id** to objects having a partner_id. Then partner_id is also automatically hidden in forms and only contact_id is shown.
2. User cannot see any visual difference with current v7
3. When contact_id is set, an **on_change** properly sets partner_id to the right company or physical person id. So partner_id is exactly what the code has always been made for.
4. optionally or if user belongs to "advanced contact group", the user may edit both contact_id and partner_id fields so that he can pick a company that isn't the usual company of the contact, when a contact belongs to several companies.

CONCLUSION 7

BUT

over the last month, OpenERP SA repeatedly rejected that idea, even after 130 messages from the most experienced OpenERP integrators in the world ALL rejecting OpenERP SA model and ALL agreeing on having two keys partner_id and contact_id instead.

https://bugs.launchpad.net/openobject-addons/+bug/1160365

CONCLUSION 8

Yeah, I think they actually f*~k her!
But eventually we work out an unf*ck solution.

See these links:

- goo.gl/aYG3S
- https://docs.google.com/a/akretion.com.br/document/d/1CvPz-BZnZ-waQZoFpdIM6aNjjcbdLadGQqTZFL3lw7A/edit#heading=h.19sozwzfx45i
- https://bugs.launchpad.net/openobject-addons/+bug/1160365/comments/27

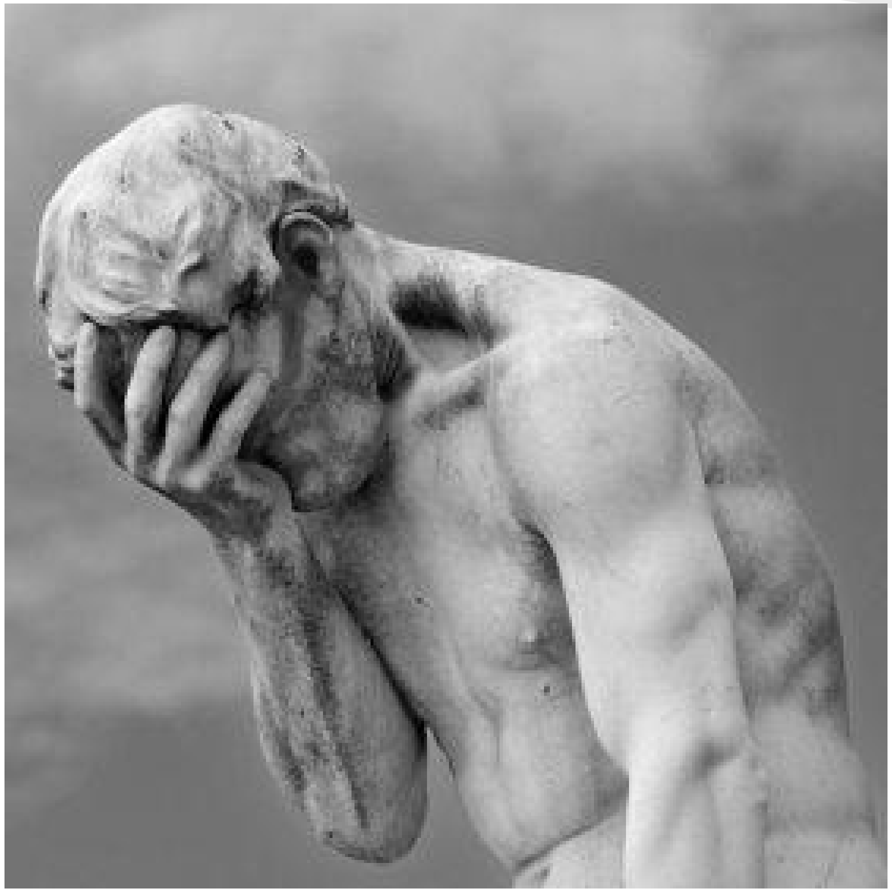
CONCLUSION 9

"Wait, after 130+ posts, we got the message, in fact we might need a second field on an **invoice**": 'commercial_entity_id' well, this would give us:

invoice object	contact	company
v6.1	address_invoice_id	**partner_id**
v7	**partner_id**	commercial_entity_id

* #sorrysap

ô rage, ô desespoir…



I tell you, "we simplified everything!"

and oh, fear not! the same code is going to make it!

our code, your code... Fear not, feel the presence...

EPILOGUE

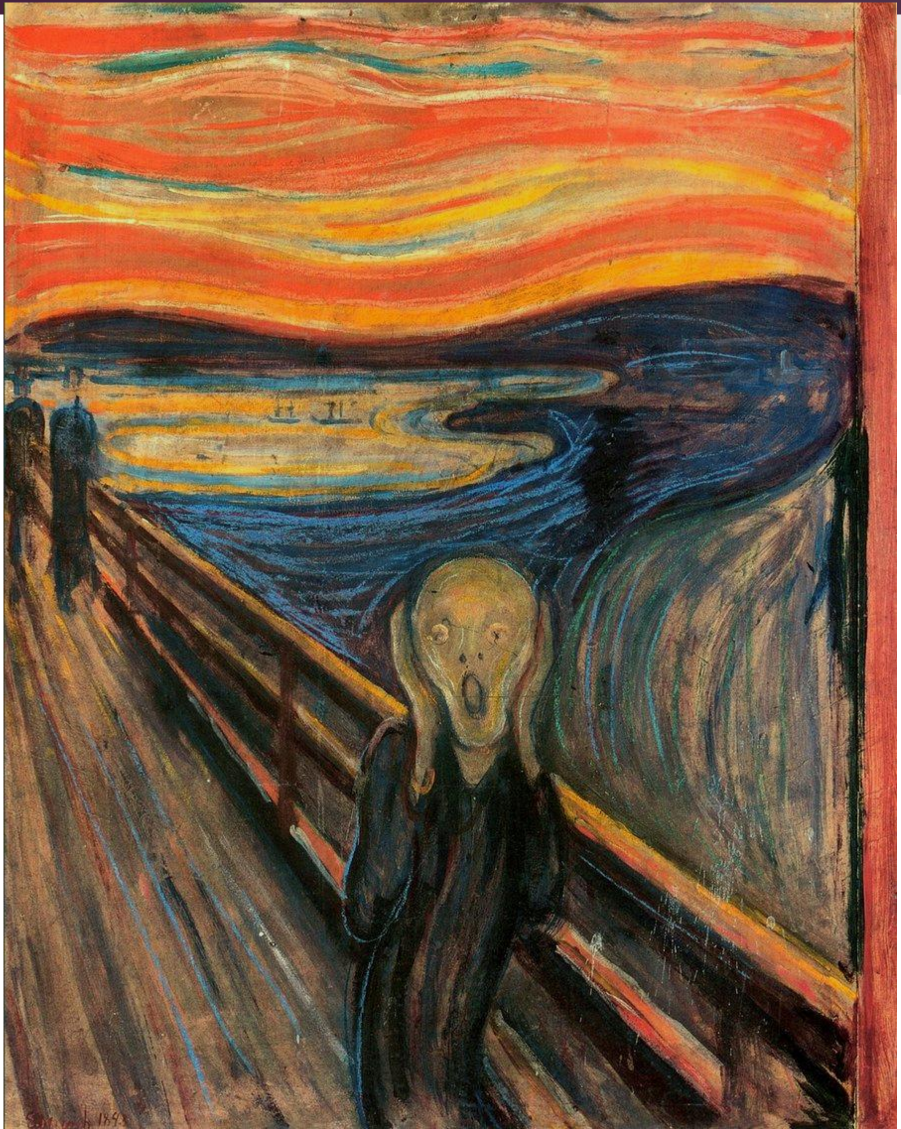
It took us ~130 posts to convince them the invoice needed two fields, may be they also understand in a few years we also need that second field to the company on:

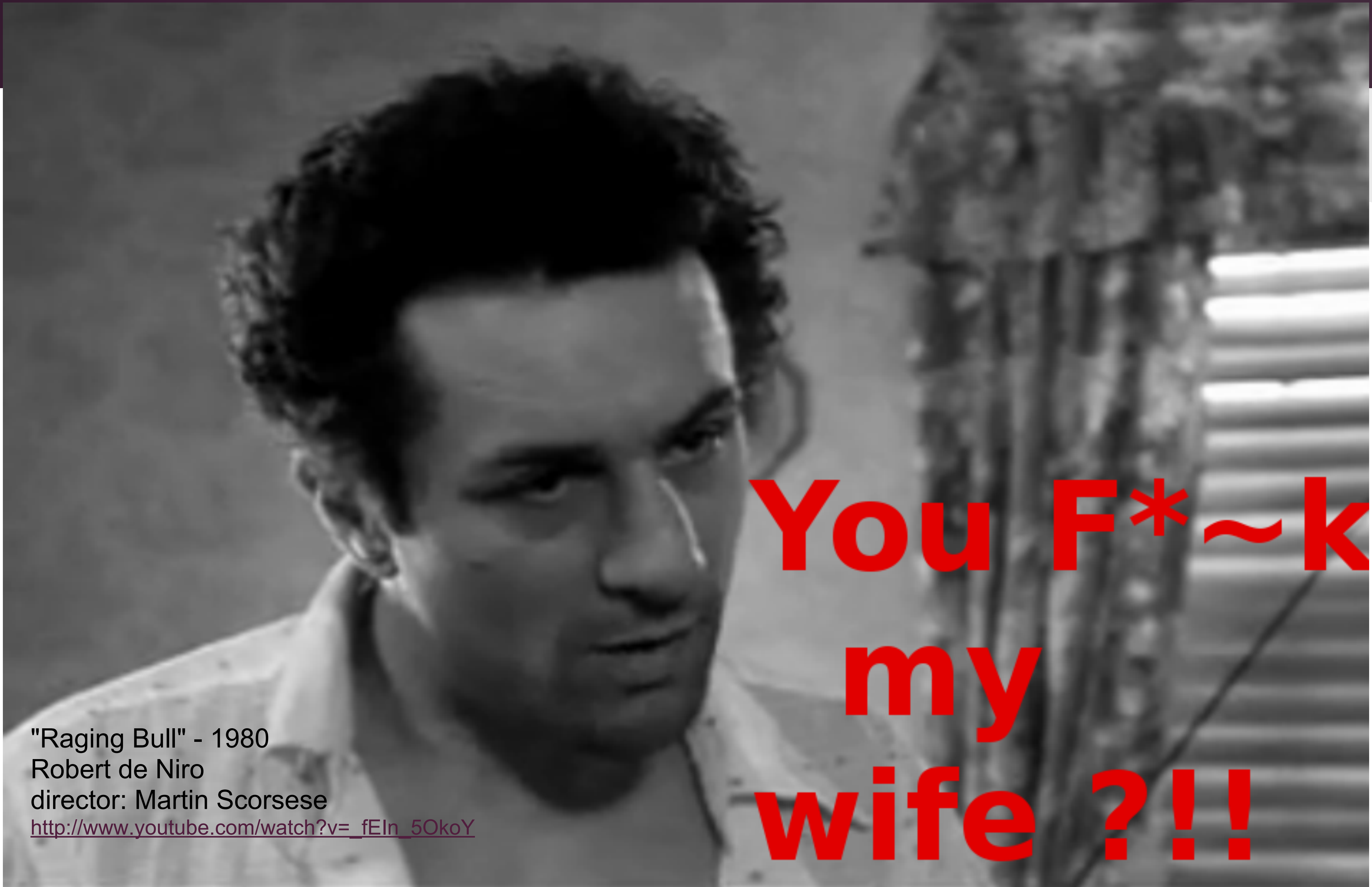
- purchase orders
- picking
- analytic accounts
- CRM opportunities
- CRM claims
- project tasks
- purchase requisitions...

So we may enjoy the field inversion for lot's of business objects and all official and third party code using them...

But hey, don't worry, we will make a script...

And who said our offshore forces cannot do in a few months, what hundreds of experts did in 8 years...





You F*~k my wife ?!!

"Raging Bull" - 1980
Robert de Niro
director: Martin Scorsese
http://www.youtube.com/watch?v=_fEIn_5OkoY