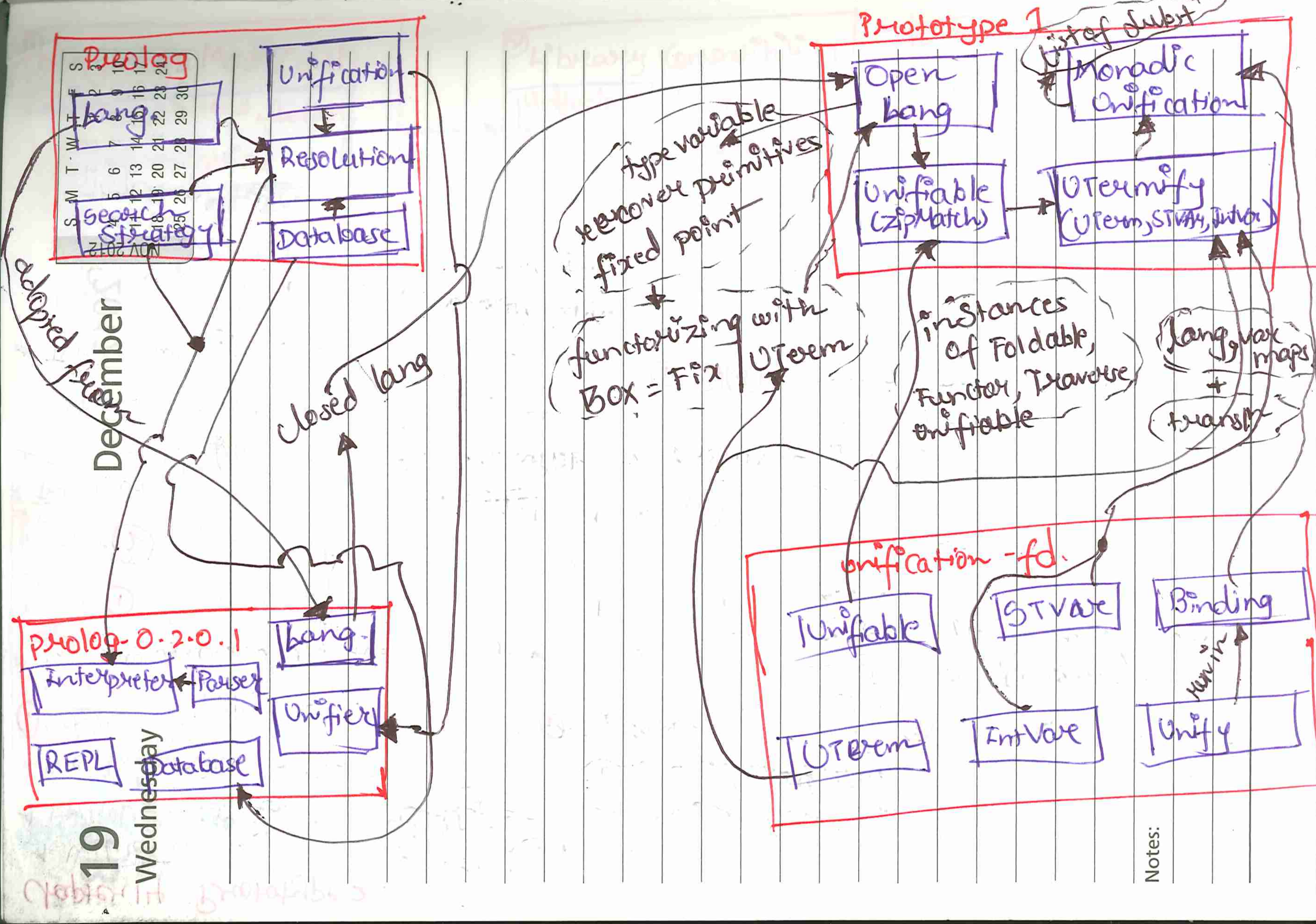
Chapter 13 Prototype 1

* [illegible] Prolog-like (open) language in Haskell adopted from prolog-0.2.0.1 with monadic unification and few practical features
* Start with prolog what if (is, does, doesn't, ----), how to bring it into Haskell
* related work → papers / libraries / implementation } → prolog-0.2.0.1 → prolog like language; unification-fd → first-order unification over general structured types
* prolog-0.2.0.1 → base lang → opened, fix, flanned, made unifiable, functorized
* unification-fd → make compatible to library → unify monadically → substitutions
  - → define prolog like unification procedure → unify monadically
* recap: recursive abstract syntax lang → extend lang } compatibility, other benefits → open → unifiable } compatibility → unify → monadic unify → substitutions

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

JAN 2013

Notes:



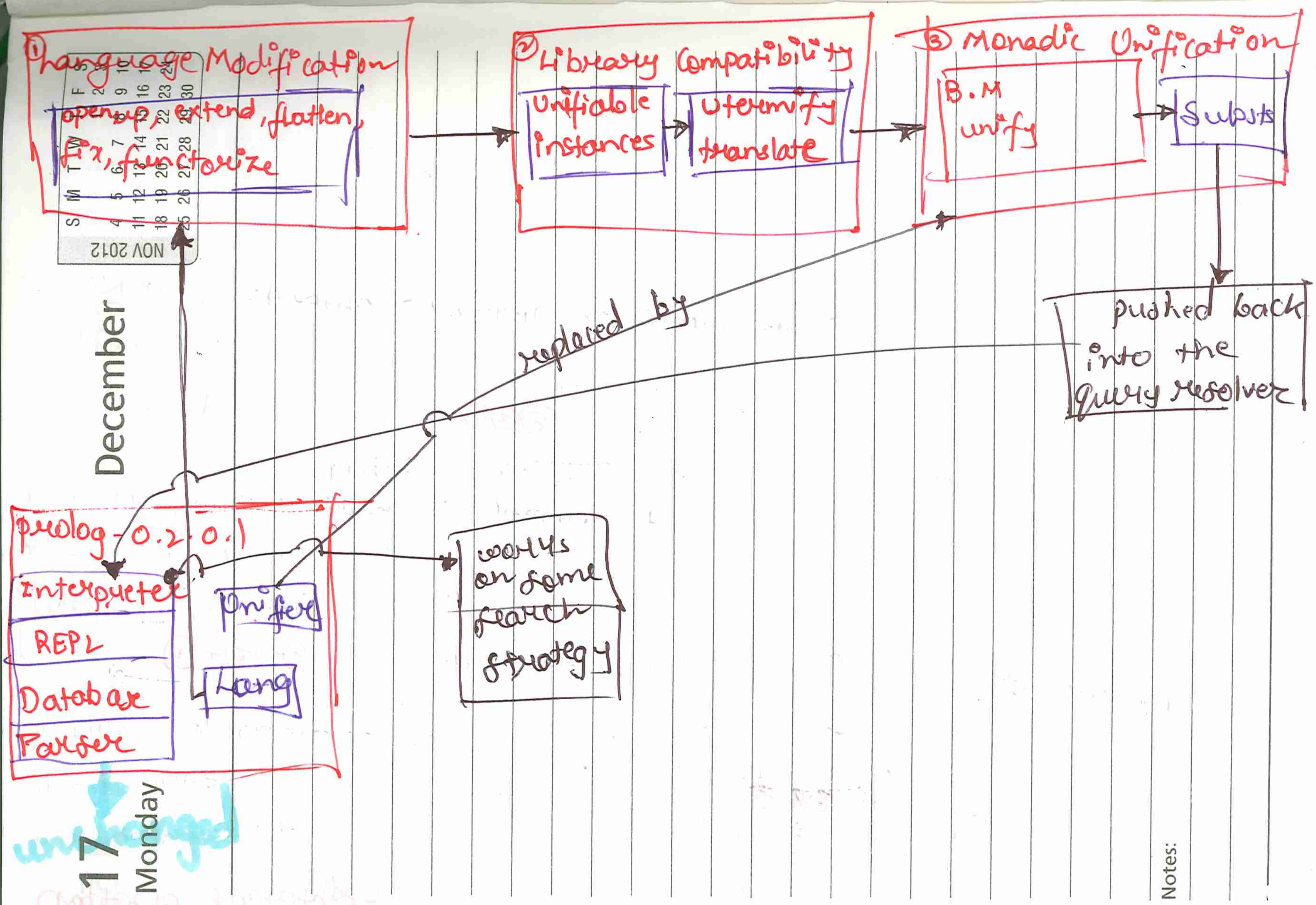
Notes:

Chapter 14 Prototype 2.

* Unification is just a part of Prolog processing.
* Unification + search strategy == query resolution in prolog,
* The language modification process + library compatibility + monadic unification

① {flatten, fix, functorize} ② {unifiable + uttermify} ③ {unify in binding monad to get substitutions}

* In prolog-0.2.0.1, the interpreter is the equivalent of the query resolution
* Apply ① to the lang. expressions when the query comes in.
* Apply ② to the above.
* The built-in unification is replaced by ③
* The results from above are translated back to the original form and pushed into the interpreter
* This prototype shows the modularity of ① ② ③; meaning upon appropriate language modification, monadic unification can be applied and the results can be injected back into the original domain. and it fits into the bigger picture.

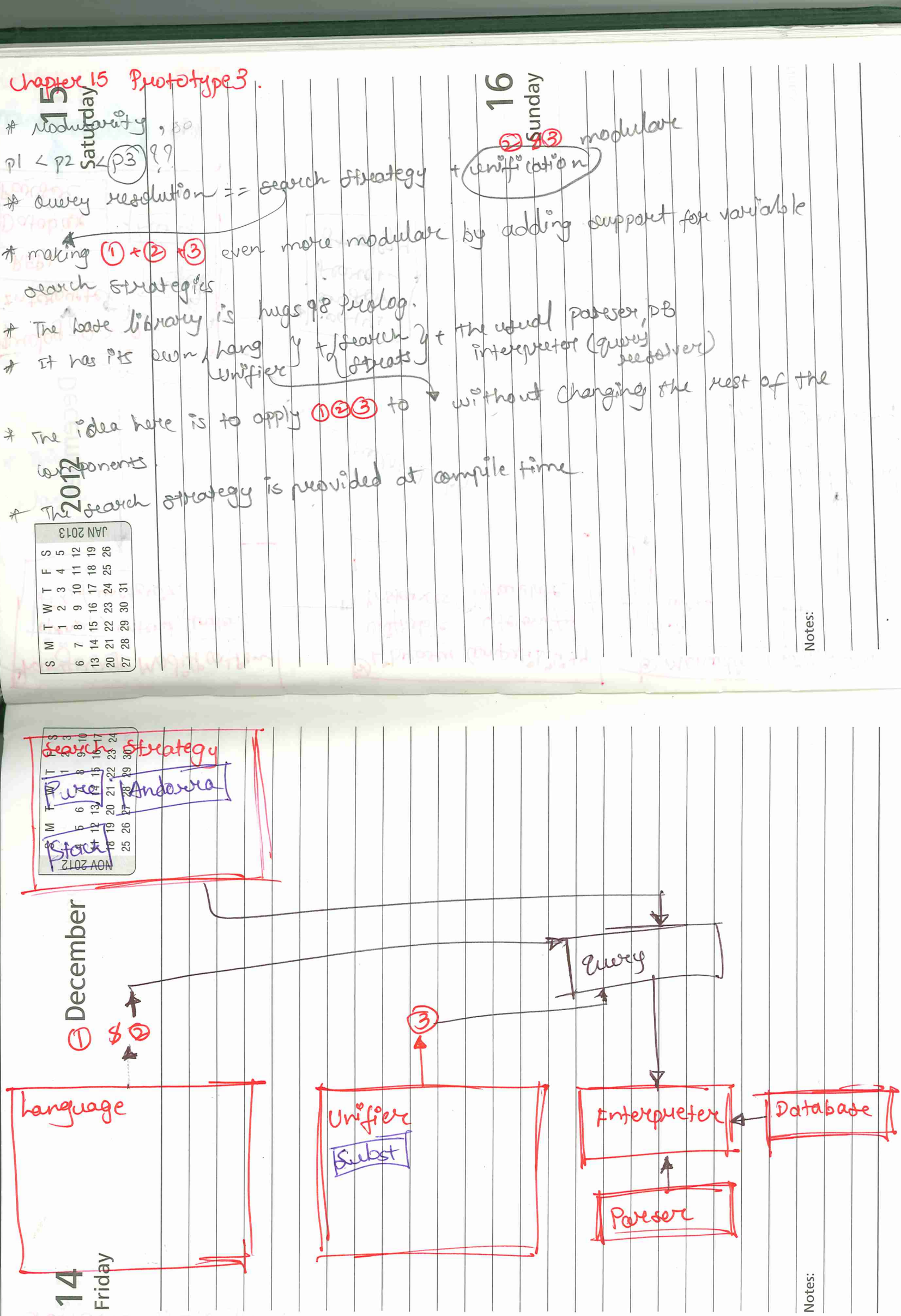


Chapter 15 Prototype 3.

15 Saturday

* Modularity, so
p1 < p2 < p3 ??
* Query resolution == search strategy + unification ② ③ modular
* making ① + ② + ③ even more modular by adding support for variable search strategies
* The base library is huge as Prolog.
* It has its own { hang unifier } + { search strats } + the usual parser, DB interpreter (query resolver)
* The idea here is to apply ① ② ③ to ... without changing the rest of the components.
* The search strategy is provided at compile time.

16 Sunday



14 Friday

Notes: