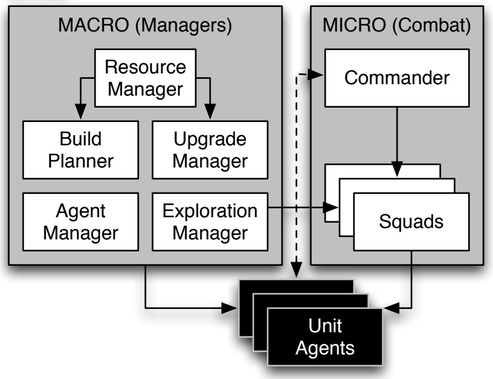
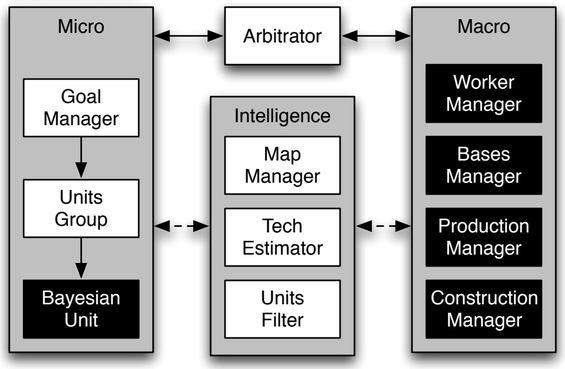
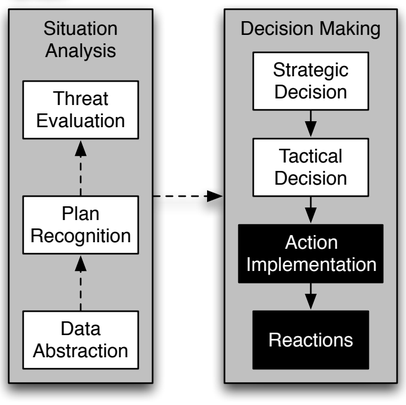
BTHAI:



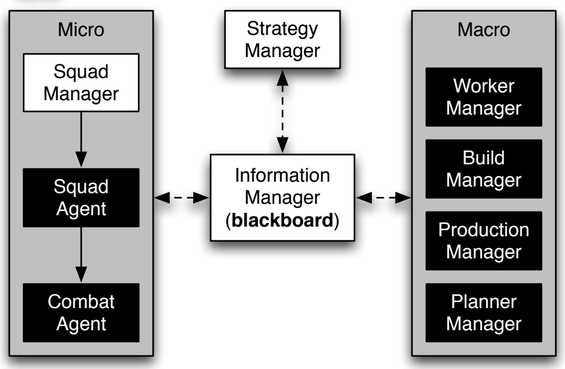
BroodwarBotQ:



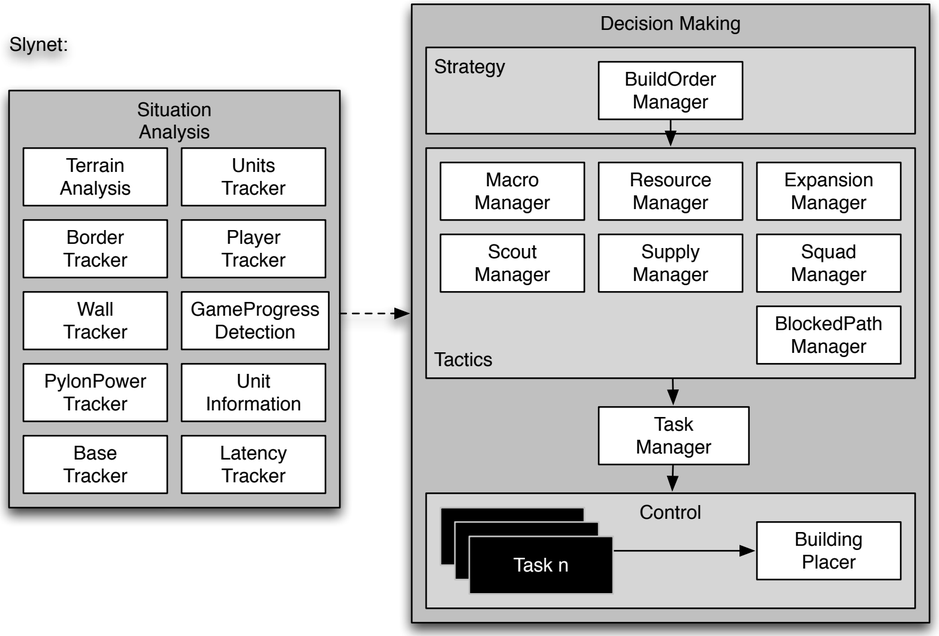
SPAR:



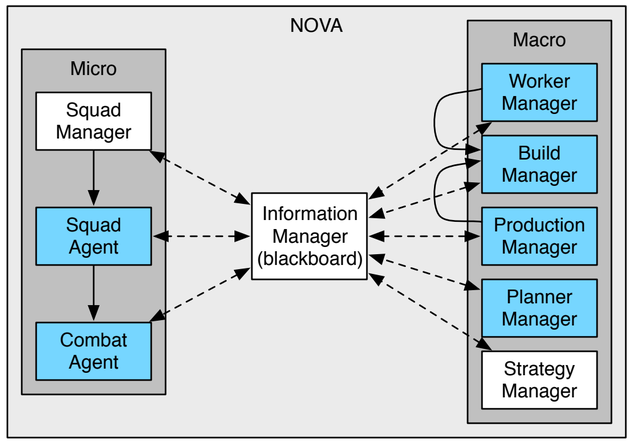
Nova:



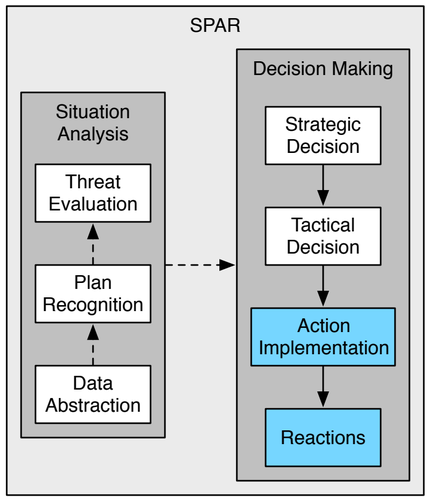
Slynet:



I marked in blue those modules that can issue commands to units

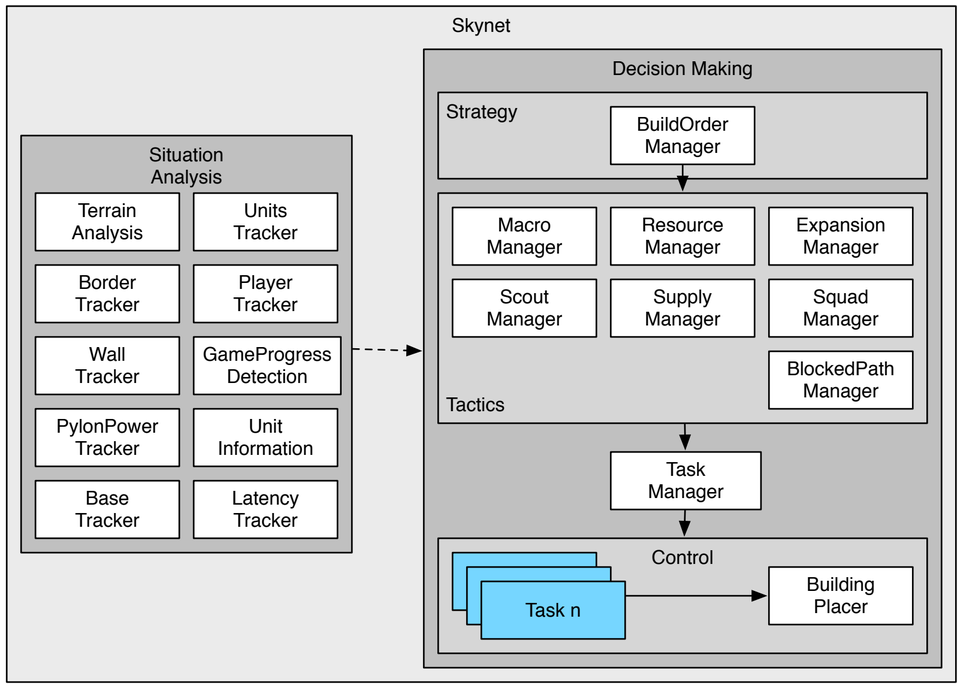


Nova coordination happens at 2 levels:
- all information communication in the blackboard (except for some direct requests to the build manager to find proper building locations)
- MICRO has a hierarchical control, where the squad manager controls squad agents, which control combat agents (individual units)



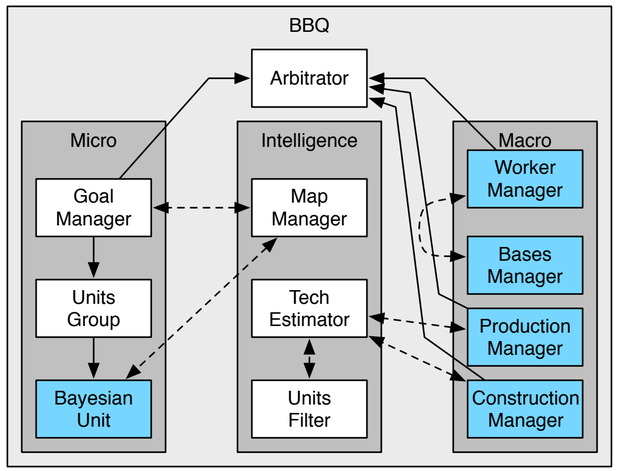
Very different from Nova or BBQ, in that it's split by abstraction lever, rather than by "task". The decision making module works like this:
- Strategic decision: decides army and building composition
- Tactical decision: decides which "abstract actions" to execute (attack, defend, build)
- Action implementation: executes abstract actions
- Reactions: FSMs for each unit with reactive behavior

Situation Analysis is similar to the "Intelligence" module of BBQ.

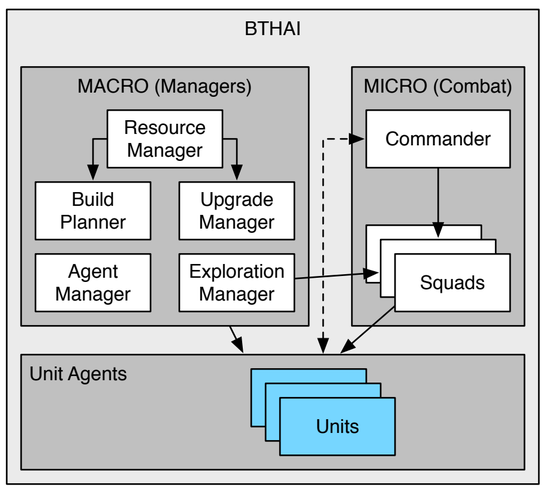


Skynet has a VERY finegrained division of tasks, which both divides everything by abstraction, as well as by task. At a high level, it looks like SPAR: a collection of situation analysis modules that produce information for the decision making module.

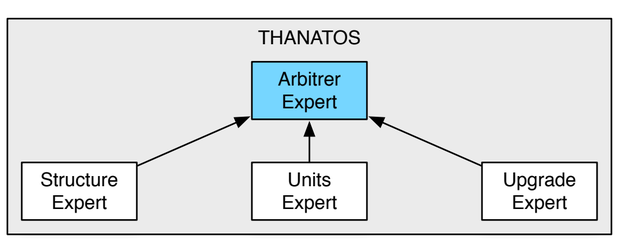
- The highest level is the BuildOrder, which is a compeltely hardcoded set of orders, to an extremely fine grained level of detail. Expliciting the build-order, when to expand, when to train units, when to research, etc. HERE is where resources are arbitrated amongst different tasks.
- At the tactics level, there are a collection of modules that take the orders form the strategic layer, and turn them into specific "tasks" (which are like the abstract actions in SPAR). There is a task manager that queues up tasks, and gives priority to them (in function of a specified set of rules).
- At the control level, each different kind of task (there are 11 basic types) has its own implementation that sends orders to units and handles the reactive control.



BBQ coordinations happens at 2 levels:
- MACRO modules are coordinated through a bidding process
- MICRO modules have hierarchical control (goal manager assigns goals to unit groups, which control bayesian units)



MACRO: build planner and upgrade planner have predefined orders read from file (not reactive)
MICRO: commander have predefined list of squads to fill, and sends them to attack or defend using a rule-based system
Resources are not arbitrated: first-come first-serve
The most interesting thing of this bot is the micro control of squads using potential fields

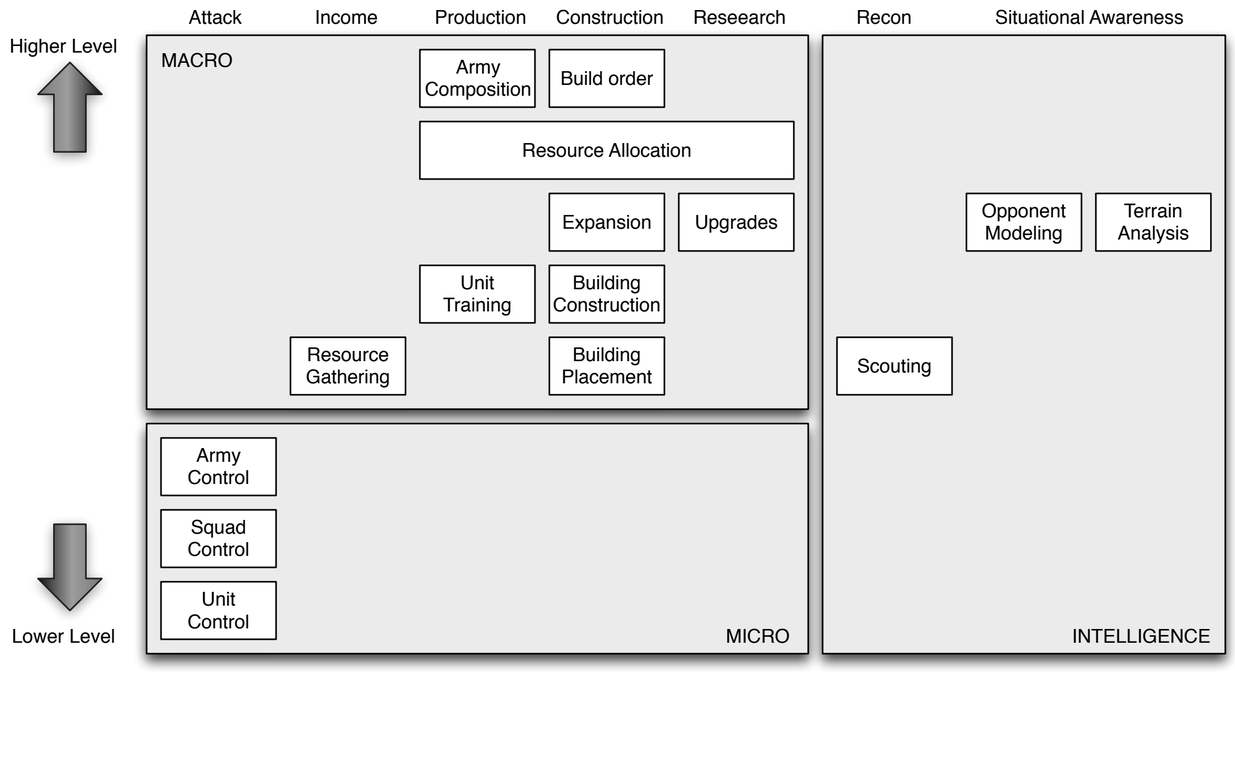


Thanatos coordination happens through a central arbitrer, that learns which requests to execute by using Q-learning.

RESOURCE ARBITRATION:

- When modules share resources (gas/mineral/buildings), an "arbitrer" or coordinator is needed:
  - NOVA has the "planner manager", that distributes tasks amongst buildings, and the "production manager", that distributes resources amongst tasks
  - BBQ has the bidding mechanism for the same purpose
  - THANATOS has the arbitrer module for exact the same purpose.
  - SPAR doesn't have this problem, since it has a hierarchical organization, and there are no completing modules for units or shared resources (resources are arbitrated at the highest strategic level)
  - SKYNET is like SPAR, resource allocation for different tasks is predefined in the build order.
  - BTHAI does not arbitrate, it uses a first-come first-serve rule

Most bots have the 3 big grey boxes differentiated, but then NOVA, BBQ and THANATOS divide the white boxes **vertically** (i.e. by "topic"), whereas SPAR divides them **horizontally** (i.e. by level of abstraction)



	Subtasks	Nova	Thanatos	BTHAI	BroodwarBotQ	SPAR
Game		Starcraft	Stratagus	Starcraft	Starcraft	Starcraft
Micro	Individual Unit Control	Combat Agent		Unit Agent(s)	BayesianUnit	Reactions
	Group Formation	Squad Agent			UnitsGroup	Actions Implementation
	Overall Unit Control	Squad Manager		Commander	GoalManager	Tactical Decisions
	Target Selection	Squad Manager		Commander	???	Tactical Decisions
	Target Reactive Selection	Squad Agent		???	???	Reactions
Macro	Resource Gathering	Worker Manager		WorkerAgent	WorkerManager	Tactical Decisions (high level) + Actions Implementation (low level)
	Repair					
	Building Placement	Build Manager			BuildingPlacer	Actions Implementation
	Resource Spending	Production Manager	Arbitrer Expert		ProductionM+ConstructionM	Strategy Decisions
	Assign Tasks to Buildings	Planner Manager		StructureAgent	ProductionManager	Tactical Decisions
	Build Order	Strategy Manager	Structure Expert	BuildPlanner	Standard (Rules)	Strategy Decisions
	Which units to train?		Units Expert	Commander	Rules + Intelligence (adaptive through ML)	
	What to research?		Upgrade Expert	UpgradeManager		
	Expansion			BuildPlanner	???	Tactical Decisions
	Popiulation Control	Build Manager			???	Strategy Decisions
Intelligence	Scouting	Information Manager		Exploration Manager	Intelligence	Tactical Decisions (high level) + Actions Implementation (low level)
	Opponent Modeling			PFManager		Plan Recognition + Threat Evaluation
	Terrain Analysis	BWTA		BWTA	BWTA+heuristics	BWTA
Integration		Blackboard for Macro, Hierarchical for Micro	Arbitrer learns (Q-learning) which Expert to use each time		Blackboard for Macro, Hierarchical for Micro	Hierarchical
Notes			Focuses only on Macro, uses the built-in Stratagus AI for Micro State representation is the list of executed actions by the agent itself. Thus, no reactivity, nor oponent modeling.	Hard-coded build-orders, squad compositions and upgrade-orders. Resource spending is a first-come-first serve. Most interesting aspect is the use of potential fields for unit control		SPAR is completely different from NOVA, BBQ or BTHAI. It's organized hierarchically, rather than as a blackboard distributed system. Higher modules generate "abstract actions", that the lower level modules implement.